DOCTORAL THESIS

# HOLOSCOPIC 3D PERCEPTION FOR AUTONOMOUS VEHICLES

*A Thesis submitted to Brunel University*
*in accordance with the requirements*
*for award of the degree of Doctor of Philosophy*

*in*

*Department of Electronic and Computer Engineering*

Chuqi Cao

January 19, 2021

# Declaration of Authorship

I, Chuqi Cao, declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ............................................................. DATE: ..................................

(Signature of student)

# Abstract

Autonomous mobile platforms are going to be huge part of the future transportation and autonomous navigation is the critical part of autonomous platforms. An autonomous mobile platform navigates the vehicle by perceiving the environment through the sensors mount on the vehicle, and acting on the data it receives from these sensors by making sense of the environmental and surroundings. As a result, an autonomous mobile platform consists of localisation aka positioning and path planning. Both of them require very accurate sensor measurements. In terms of accuracy, sensor can generally be divided into two groups (a) High accuracy sensors like the state-of-the-art in LiDAR and vision sensors e.g. mobile-eye sensor. (b) Low accuracy sensors whereas GPS (accurate within 2-10 metres) sensor and IMU (suffering from drifts) could be fused to improve the other method of positioning. These are expensive process due to offline map creation. To deal with low accuracy sensors, researchers normally use very complex models, which again run into performance reliability and consistency issue.

Furthermore, it is common believe, that when navigating autonomously, perception and situation cognisance is an important component to navigate safely and there have been a huge research on AI enabled perception such as Mobile Eye and Tesla car which uses 2D cameras for its perception. In this research, an innovative method is proposed to use rich vision sensor holoscopic 3D camera for environment perception with artificial intelligent algorithms to observe road objects and learn their 3D behavioural for reliable detection and recognition. The sensor provides rich information - 3D cubic visual information about the environment including the very valuable “depth information” to imitate third coordinate of real world. To learn the objects, different AI algorithms are studied and in particular deep learning model is proposed that provides a reasonable good result. To evaluate the innovative holoscopic 3D sensor, we applied to face recognition challenge under different face expression where 2D images are considered to fail. However the holoscopic 3D sensor outperform and delivered outstanding performance by recognising faces under different expression by only training on the neutral face using a simple AI algorithm. Then we design and develop holoscopic perception database of 200000 frames for autonomous car. The experimental result has shown a promising result that AI algorithm, particularly deep learning algorithm learns effectively from holoscopic 3D content compared to traditional 2D images even those DL models which were designed for visual features yet holoscopic 3D images contain motion data which shall be exploited.

# Acknowledgements

I would like to express my very great appreciation to my supervising team Dr. Rafiq Swash, Dr. Hongying Meng, Dr. Tatiana Kalganova and without their guidance and support this research may not be possible.

I would like to thank Brunel University and all ECE staff to provide me the platform and support to carry out my research.

My special thanks are extended to my family, friends and research colleagues who support me generously during my studies.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Motorised vehicles with wheels, such as a car, motorcycle, truck, or bus have had a significant impact on our lives since their introduction in the global market in early 20th century [1]. The benefits include on-demand transportation, mobility, independence, and convenience, societal well-being from leisure and travel opportunities [2]. It was reported in 2018 that the world motor vehicle production statistics were over 95 million, including over 70 million cars, according to International Organisation of Motor Vehicle Manufacturers (OICA) [3]. The numbers are increasing rapidly, especially in the newly industrialised countries like China and India.

To drive safely and handle emergency situations, a driver is required to have the following skills [4]: Making good decisions based on factors such as road and traffic conditions, evasive manoeuvring, skid control, steering and braking techniques, understanding vehicle dynamics.

Distractions caused by things like talking on a phone [5], listening to loud music [6] and medical conditions like Seizure disorders and Alzheimer's disease [7] can compromise a driver's mental skills. This can potentially result in an accident which in turn result in injuries and fatalities. In the US alone, 37,133 people died in motor vehicle crashes in 2017 [8]. According to National Highway Traffic Safety Administration (NHTSA) [9], 94 percent of serious crashes are caused by human error, which in addition to destructions include things like lack of knowledge, failure to follow traffic rules and driver's incapacity (DUI or fatigue) [10].

To help make vehicles safer, and with technological advances in computing hardware and algorithms, in last few decades, researchers have been actively working on developing both assistive technologies and autonomous car solutions. While former provides support to human drivers in the form of a Pedestrian Automatic Emergency Braking (PAEB) system, Adaptive Speed Control (ASC), Cruise Control (CC), Forward Collision Warning (FCW) and Lane Departure Warning (LDW), the latter is used to minimise the need for a human driver [10].

## 1.1 Autonomous Mobile Platform

Autonomous means having the power for self government. Autonomous car means a car that can guide itself to a specific target without or with minimum human intervention. It does this by using intelligent algorithms that decide on vehicle's key functions such as steering, throttle, brake and lights, considering perception of the environment around it through the fusion of sensor observations. An autonomous car is also known as a driverless car, robot car, self-driving car or autonomous vehicle [11].

The sensors such as camera and laser range finders are mounted on an autonomous car in strategic points. There observations are used to monitor the vehicle's movement and the movement of other vehicles, pedestrians, and potential obstacles around it, much more thoroughly than an average driver as shown in Figure 1.1 [12], especially, in the blind spots. As a result, the intelligent algorithms responsible for decision making and control of the vehicle, can safely react and also predict potential hazards. They can take action quicker than humans, and would not become prey to human errors, suggesting the potential for at least a 40% deadly crash-rate reduction, assuming automated malfunctions are minimal, and everything else including the levels of long-distance, night-time and poor-weather driving remains constant [13]. This results in a more safer means of transportation substantially reducing the current motor vehicle accidents and associate fatalities.

Other key benefits include:

(1) Efficient use of existing highway and road systems by reducing traffic accidents, decreasing the quantity of cars on the road, reducing the overall traffic.

(2) Continuously operating vehicles with maximum fuel efficiencies. Autonomous vehicles are designed to optimise efficiency in acceleration and braking, therefore they will also help improve fuel efficiency and reduce carbon emissions. The estimation of the potential of autonomous vehicles in the reduction of greenhouse gas emissions is expected to be by approximately 40-60% in total [14].

(3) Saving valuable travel time for commuting [15].

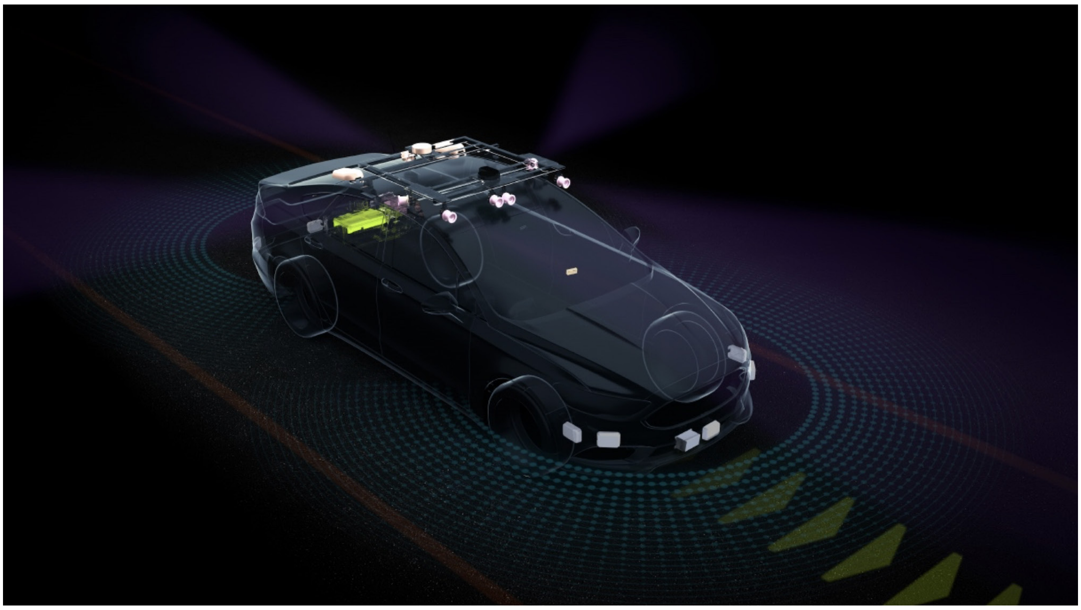


Figure 1.1: Autonomous car – environment detection
[12]

Autonomous driving consists of key technologies for the following four general areas:

(1) Environment perception: It consists of detecting environment features and classifying objects around the vehicle using fusion of information from sensor such as LiDAR, Radars, cameras and ultrasonic. Figure 1.2 [16] illustrates typical classification of objects. It also includes lane marker detection, road surface detection.

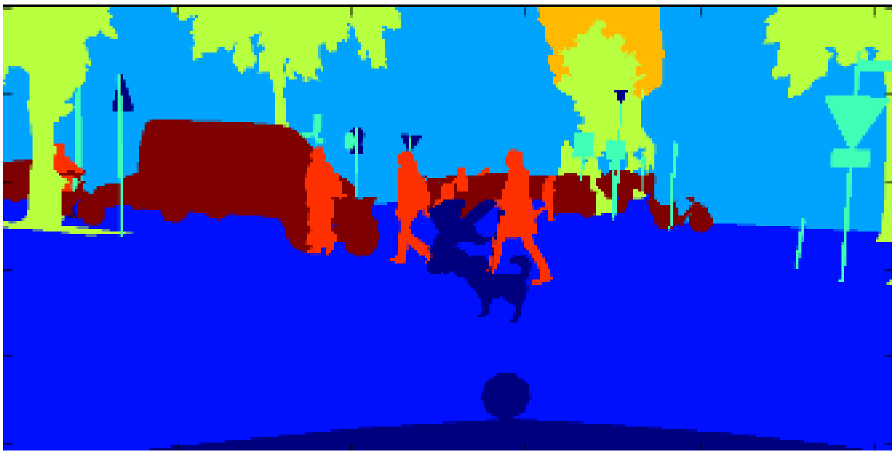


Figure 1.2: Autonomous car - classification of objects
[16]

(2) Planning: Given a desired location, path planning consists of deciding trace that the vehicle should follow in order to reach the desired destination without colliding with obstacles [17]. This is usually done both at global and local level. While the global planned path gives end-to-end trace to destination, local path planning, which

considers a finite distance forward from vehicles location is continuously performed to deal with obstacles on the move [18].

(3) Localisation: This consists of determining the location of the vehicle in a global coordinate frame. Since GPS accuracy is between 2-10 meters due to its bias, drift, dropouts, and multi-path problems, researchers use approaches such as Simultaneous Localisation and Mapping (SLAM) to localise the vehicle. Using SLAM, the vehicle can start at a known or unknown location with no knowledge of the location of features in the environment. As the it moves through the environment, it makes relative observations of the features. Using these observations, SLAM incrementally builds a map of the region explored and uses this to localise the vehicle's position [19] [20] [21].

(4) Control: This is the brain of a autonomous driving system. Based on sensor feedback, it computes and provides the input to the vehicle hardware (brake, throttle and steering) to achieve the desired motion. This is mainly achieved by using Proportional-Integral-Derivative (PID) controller or Model Predictive Controller (MPC).

While the popular understanding is that autonomous vehicles should be able to navigate from point A to point B, by planning the path in between and then navigating it keeping itself localised at all times, both of which are computationally very expensive with later requiring high-end sensors like Light Detecting and Ranging (LiDAR) sensors, and Global Positioning Systems (GPS) along with Inertial Measurement Unites (IMU), which are very expensive to acquire.

However, people often use cars to travel to certain common destinations like home (point A) to work place (point B) following a particular road network. Also, there are cheaper sensors like holoscopic 3D imaging (H3D) technology, which endeavours to provide a new method of creating and representing 3D images. The principle of this technique uses a naturally occurring process as in the "fly eye" for capturing and displaying 3D images [22]. It is a unique method for creating a true volume spatial optical model of the object scene in the form of a planar intensity distribution through a micro-lens array [23]. It uses natural light and a single aperture camera setup with a micro-lens array in the capture process. Moreover, it offers a full parallax object scene as in the real-world without the need for calibration that is required by other currently available 3D imaging techniques.

With advances in machine learning and growing availability of super-fast processing capabilities such as GPU processors and inspired by work done on autonomous driving by NVIDIA [24], using the H3D camera, rich content of perception will support key point required for autonomous control of the vehicle, like environment recognition and objects detection. This will lead to a more affordable automation solution that can be realised using just the latest vision technology.

## 1.2 The PhD Aim and Objectives

The main aim of this research is to develop AI "self-learning" platform based on holoscopic 3D systems for effective perception of environment for surrounding understanding and recognising road objects reliably.

List of main objectives of the research are as follows:

(1) Carry out a literature review on different imaging systems, e.g. 2D and 3D imaging systems, machine learning knowledge and autonomous mobile platforms

(2) Design and prototyping holoscopic 3D camera for autonomous mobile platform

(3) Holoscopic 3D image analysis and feature extraction

(4) Develop AI algorithms for perception to detect and recognise road objects for situation awareness

(5) Carry out experiment and evaluation of the AI perception system

(6) Disseminate and publish the research findings in international conferences and journals

## 1.3 Research Contributions

The research targets to contribute effective perception platform with self-learning AI platform for autonomous vehicles and in addition to this:

(1) The use of a new sensor; in particular, holoscopic 3D (H3D) camera for environment perception. H3D is capable of acquiring rich and spatial data.

(2) The use of machine learning e.g. deep learning along with some innovative data pre-processing methods for H3D database benchmark, to classify H3D based content. This is accepted as a conference paper.

(3) The use of deep learning along with some innovative data pre-processing methods for H3D scene / driving database, to classify / detect H3D based content for autonomous

control of the vehicle. This is prepared as a conference paper and ready to be submitted.

The research contribution will be measure through international conference and journal papers.

## 1.4 Publications

So far, the published and accepted international conference papers from the PhD research findings are listed below.

Published paper: C. Cao, M. R. Swash and H. Meng, ”Reliable Holoscopic 3D Face Recognition,” 2020 7th International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India, 2020, pp. 696-701, doi: 10.1109/SPIN48934.2020.9071409.

Accepted paper: C. Cao, M. R. Swash and H. Meng, “Semantic 3D Scene Classification Based on Holoscopic 3D Camera for Autonomous Vehicles,” 2020 16th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD 2020), Xi’an, China, 19-21 December 2020.

## 1.5 The Research Scope

Although a autonomous vehicle consists of many sensor technologies and algorithms through which vehicle controls are managed, in this thesis a subset of it is covered to provide a proof of concept.

(1) Short road segment: Although the distance between points A and B could be very long, in this research, for simplicity, a comprehensive holosocpic 3D dataset will be produced for capturing all road objects and then the holoscopic 3D dataset requires preparation including annotation for AI training.

(2) Only consider obstacles on the road: Since the aim of research is to achieve perception required for situation awareness, common objects on the road, e.g. cars, people and buses, will be considered.

## 1.6 Thesis Contents

Since people often travel between destinations, this thesis will mainly focus on developing machine learning algorithms and approaches that will learn driver behaviour for these short journeys with help of H3D sensor technology. The thesis is divided into chapters, each chapter focusing on a key aspect of the research. Chapter 2 of literature review presents the review of sensor and imaging technologies used in automation and details the proposed H3D imaging technology, along with techniques and approaches for image processing, scene recognition and object detection. These are vital to the final proposed model for control of vehicle navigation using only images. Then the review of machine learning and neural networks are provided. Chapter 3 presents the design of the proposed camera, process of its calibration and assembly on an RC car and a real car. In Chapter 4, it first reviews different face recognition methods and then bring together the proposed camera system discussed in Chapter 3 and proposed image processing method, and use them for an experiment of face recognition using a neural network. Chapter 5 includes two experiments of proposed H3D perception platform application on both RC car and real car, using deep learning neural networks along with the performance and evaluation. Finally the conclusion of all the work done in this thesis is provided in Chapter 6, and detailed the tasks planned for future research.

# Chapter 2

# Literature Review

## 2.1 Introduction

This chapter of literature review is divided into four parts. Before detailing the mechanics and use of H3D sensor, there is a review on some of the existing sensor technologies used for automation in both industry and research in Section 2.2. Then the review of the progress of different imaging technologies starts by explaining 2D imaging system, and continue to 3D imaging technology with the details of the H3D imaging technology in Section 2.3. Section 2.4 aims to provide the essential in pre-processing that will be required by our proposed deep learning model. It starts by introducing computer vision, then reviews the three main areas in computer vision that we propose to use for pre-processing the H3D data, which are image features, scene recognition and object detection. Section 2.5 provides an review of the machine learning approach called deep learning that we propose to use in this thesis. It starts by describing what machine learning is, what is it used for and its types, then details one of the key approaches in machine learning called artificial neural networks (ANN). Having reviewed ANNs key components, it then moves on to explaining what deep learning is. There are different types of deep learning neural networks used for different application. Since this thesis is dealing with visual data, the details of deep learning approach that deals with visual data called convolutional neural networks (CNN) are provided. Finally, the summary of the chapter is provided in Section 2.6.

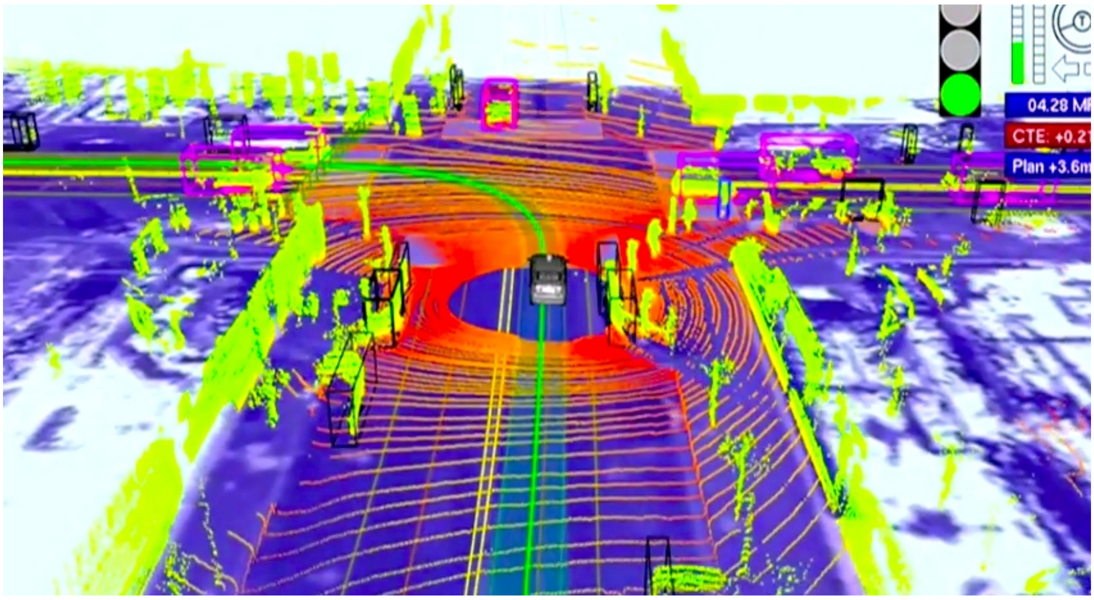


Figure 2.1: Google's driverless car using LiDAR
[26]

## 2.2 Environment Perception Sensors

There are different types of sensors used to perceive the environment. In general they can be categorised as laser-based (LiDAR), radio wave based (Radar), and vision-based (2D/3D Images) sensors.

### 2.2.1 LiDAR

LiDAR stands for Light Detection and Ranging. It is an active remote sensing method that determines ranges (i.e., distances), which using the product of the speed of light and the time required for an emitted laser to travel to a target object. Light moves at a constant and known speed so the LiDAR instrument can calculate the distance between itself and the target with high accuracy. Pulse ranging can be used to measure the time elapsed since the laser is emitted from the sensor and the object is intercepted, where the travel time of the laser pulse from the sensor to the target object is recorded. At present, most LiDAR systems used are based on the principle of pulse ranging [25].

Main application of LiDAR in automation include generating very high-resolution 3D maps of the surface of the environment and object detection, Figure 2.1 [26] shows how Google's self-driving car uses LiDAR to create 3D image of its surroundings.

Example of LiDAR used in automation is shown in Figure 2.2 [27]. The Velodyne LiDAR VLS-128 is applied on a Renovo AWare powered Voyage Pacifica. Figure 2.3 shows the Velodyne LiDAR VLS-128, which is a sensor specifically made for autonomous driving

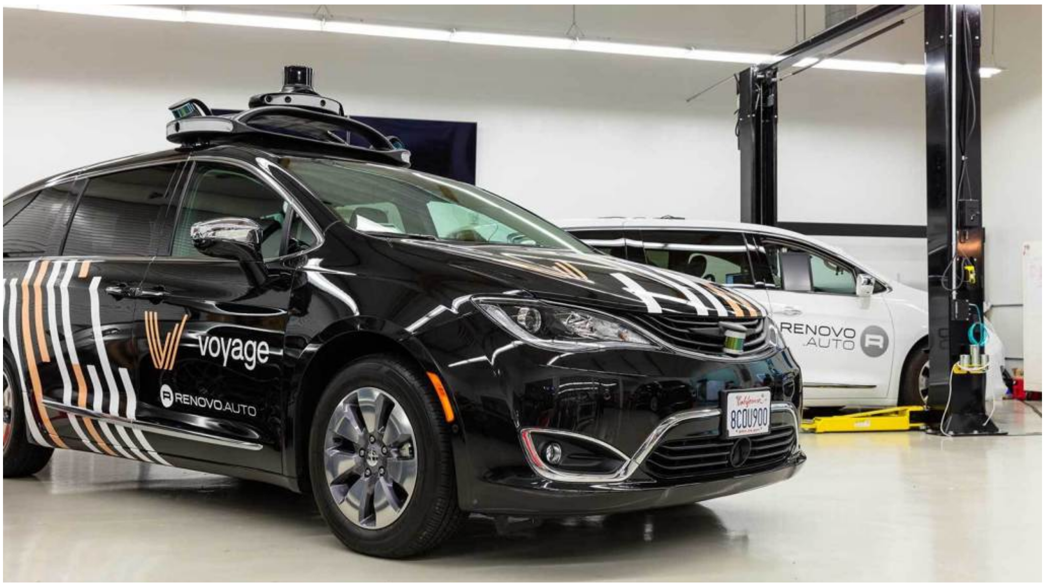


Figure 2.2: Velodyne LiDAR VLS-128 on a Renovo AWare powered Voyage Pacifica [27]

and advanced vehicle safety at highway speeds. It delivers real-time 3D data up to 0.1-degree vertical and horizontal resolution with up to 300-meter range and 360°surround view [27].

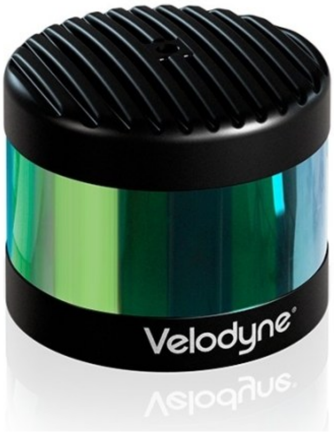


Figure 2.3: Velodyne LiDAR VLS-128 [28]

### 2.2.2 RADAR

RADAR technology uses radio waves to determine the range, angle, or velocity of objects. Automotive radar employs millimetre-wave frequencies for long-range object and obstacle detection, as well as for tracking the velocity and direction of the various obstacles in the environment around the vehicle such as pedestrians, other vehicles, guardrails, etc. [29]. Radar sensors provide reliable information on straight lanes, but have a narrow field of view and reduced angular resolution, so they fail in curves due to their restricted field of view. However Radar sensors use the Doppler effect to directly provide velocity information. [30]

In the late 50's, first experiments in the field of automotive radar have already taken place. Some intensive radar developments started at microwave frequencies in the 70's. The activities of the last decades were mostly focused on developments at 17 GHz, 24 GHz, 35 GHz, 49 GHz, 60 GHz, and 77 GHz. Even from the early stage in automotive radar, the idea of collision avoidance has been the key driver of all these investigations, which has been the motivation for many engineers to develop smart vehicular radar units. A lot of technical knowledge has been gained in the field of microwaves and in radar signal processing during this long period. Accompanied by the remarkable progress in semiconductor microwave sources and in available computing power of micro controllers and digital signal processing units, the commercialisation of automotive radar became feasible in the 90's [31].

There are three principle categories of radar systems are typically applied in automotive active safety systems [29]:

(1) Short-range radar (SRR) for collision proximity warning and safety, and to support limited parking assist features

(2) Medium-range radar (MRR) to watch the corners of the vehicle, perform blind spot detection, observe other-vehicle lane crossover and avoid side/corner collisions.

(3) Long-range radar (LRR) for forward-looking sensors, adaptive cruise control (ACC) and early collision detection functions.

Main application of Radar in autonomous application is illustrated in Figure 2.4 [29].

### 2.2.3 Vision

Visual perception is the ability to interpret the surrounding environment using light in the visible spectrum reflected by the objects in the environment. With visual perception, cars can use sensors such as cameras to detect vehicles in front of them, identify them as potentially dangerous, and know that they can continuously track their movements. This capability extends to the 360-degree field of view around the vehicle, enabling the car to detect and track all moving and static objects while driving [32].

There are essentially two approaches for obstacle detection, active methods and passive methods. Active methods use sensors such as laser scanners, time-of-flight, structured light or ultrasound to search for obstacles. In contrast, passive methods use camera images to

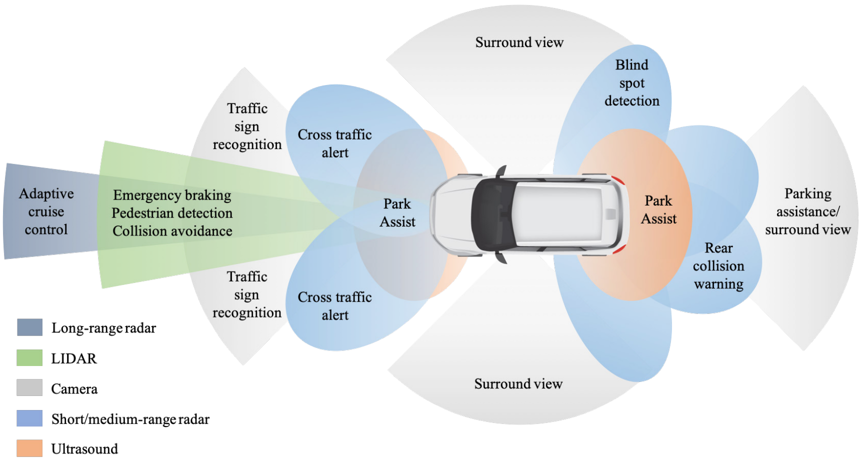


Figure 2.4: Autonomous vehicle RADAR sensing architecture [29]

detect obstacles based on passive measurements of the scene, for example, a wide field of view can be covered using fish-eye cameras. The using of cameras take advantages of that they work over diverse conditions of weather and lighting to offer a high resolution, at the same time cameras are worthwhile in economy [33]. Successful examples of autonomous vehicles such as the Stanford Shelley [34], AnnieWAY [35] or the Google's self-driving car [36], are all using LiDAR to detect objects and camera for traffic sign recognition and delineation, affecting the overall mission plan. These autonomous vehicles can work properly with accurate and reliable data equipped with multiple sensors, using different physical properties (light, ultrasound, radio frequency, etc.) [37].

In current status, camera systems used for autonomous vehicles are usually forward-facing, rear and 360° [38]. Forward-facing camera systems are systems for medium to high ranges, such as in the area between 100 and 275 yards. These cameras use the algorithms to automatically detect objects (e.g. traffic signs and signals), classify them (e.g. pedestrians and cyclists, motor vehicles, side strips, bridge abutments, and road margins) and determine the distance from them. Rear and 360° cameras support the driver with a better representation of the environment outside the vehicle. The input signals from four to six cameras are required for the three-dimensional image to be realistic, and it is necessary to specialise the 'image stitching' to avoid loss of image information or generation of ghost images. Today's automobile rear and 360° video systems usually have a centralised architecture. This means that a central control unit processes the raw data of four to six cameras.

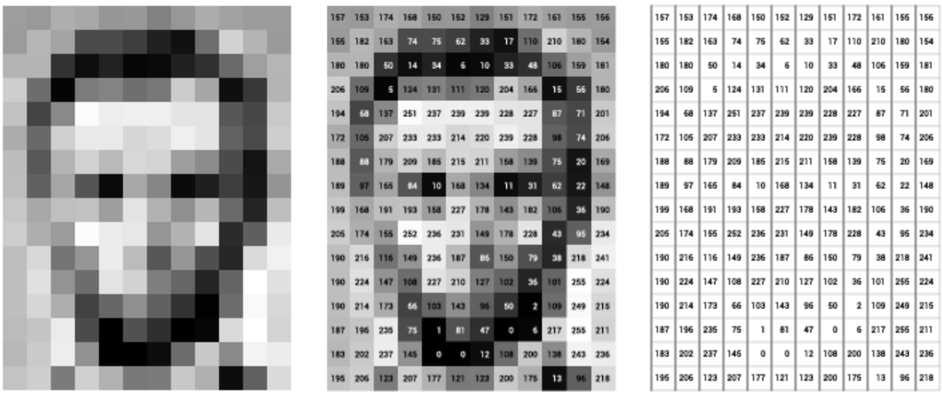
Research in using cameras on autonomous vehicles usually use monocular camera [33] [39], stereo camera [40] [41], surround-view camera or multi-camera [42] [43], for example, mounting four cameras on the Nissan Quasquai Around View Monitor to provide full omni-directional view around the car [44].

## 2.3 Imaging Systems and Technologies

In this section, it starts the review of the progress of different imaging technologies by explaining 2D imaging system, and continue to 3D imaging technology with the details of the H3D imaging technology.

### 2.3.1 2D Imaging System

A standard 2D image is flat with the information of length and width. It does not provide any depth information. Figure 2.5 illustrates grey scale 2D image which is basically a matrix of pixel values between 0-255 [45]. For colour 2D images, each pixel is represented with three values for red, green and blue, therefore each pixel of the image has three channels and is represented as a 1x3 vector, as shown in Figure 2.6 [45].



| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 157 | 153 | 174 | 168 | 150 | 152 | 129 | 151 | 172 | 161 | 155 | 156 |
| 155 | 182 | 163 | 74 | 75 | 62 | 33 | 17 | 110 | 210 | 180 | 154 |
| 180 | 180 | 50 | 14 | 34 | 6 | 10 | 33 | 48 | 106 | 159 | 181 |
| 206 | 109 | 5 | 124 | 131 | 111 | 120 | 204 | 166 | 15 | 56 | 180 |
| 194 | 68 | 137 | 251 | 237 | 239 | 239 | 228 | 227 | 87 | 71 | 201 |
| 172 | 105 | 207 | 233 | 233 | 214 | 220 | 239 | 228 | 98 | 74 | 206 |
| 188 | 88 | 179 | 209 | 185 | 215 | 211 | 158 | 139 | 75 | 20 | 169 |
| 189 | 97 | 165 | 84 | 10 | 168 | 134 | 11 | 31 | 62 | 22 | 148 |
| 199 | 168 | 191 | 193 | 158 | 227 | 178 | 143 | 182 | 106 | 36 | 190 |
| 205 | 174 | 155 | 252 | 236 | 231 | 149 | 178 | 228 | 43 | 95 | 234 |
| 190 | 216 | 116 | 149 | 236 | 187 | 86 | 150 | 79 | 38 | 218 | 241 |
| 190 | 224 | 147 | 108 | 227 | 210 | 127 | 102 | 36 | 101 | 255 | 224 |
| 190 | 214 | 173 | 66 | 103 | 143 | 96 | 50 | 2 | 109 | 249 | 215 |
| 187 | 196 | 235 | 75 | 1 | 81 | 47 | 0 | 6 | 217 | 255 | 211 |
| 183 | 202 | 237 | 145 | 0 | 0 | 12 | 108 | 200 | 138 | 243 | 236 |
| 195 | 206 | 123 | 207 | 177 | 121 | 123 | 200 | 175 | 13 | 96 | 218 |

Figure 2.5: 2D greyscale image pixel value
[45]

In computer vision, many applications involve scenes that are two-dimensional essentially. An object is defined as an arrangement of parts whose properties (e.g., grey levels, textures, sizes, shapes) and relations (e.g., relative position, relative size, etc.) satisfy given constraints. Therefor, we should look for a collection of image parts that correspond to the object parts and satisfy the appropriate constraints to recognise an object in the image [46]. Typical limitations of 2D computer vision technology include parallax, depth of focus, ambient light, and variations in contrast.

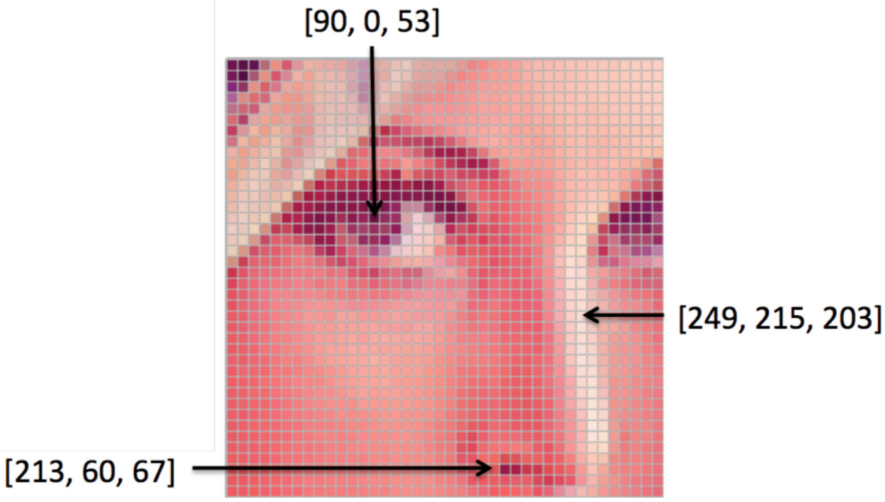


Figure 2.6: Colour 2D image pixel value
[45]

### 2.3.2 Stereoscopic 3D Imaging Technology

Stereoscopic 3D imaging technology is first created by Sir Charles Wheatstone in 1838 [47]. It is also known as binocular vision, which present two offset images separately to the left and right eye of the viewer. These two-dimensional images are then combined in the brain to give the perception of 3D depth. This is a human eye technique which require the observer to wear special equipment to transmit the two images to the corresponding eye. This technique is distinguished from 3D displays that display an image in three full dimensions, allowing the observer to increase information about the three-dimensional objects being displayed by head and eye movements.

Two/stereo image capturing is used by the capture system of this technology. The stereoscopic image capture system is usually set up either by a side-by-side camera rig or mirror camera rig as shown in Figure 2.7. An example of stereoscope image is illustrated in Figure 2.8 [48].

There are two categories of 3D viewer technology, active and passive. Active viewers have electronics which interact with a display. Passive viewers filter constant streams of binocular input to the appropriate eye.

Active shutter 3D systems generally use liquid crystal shutter glasses [49]. These glasses are controlled by a timing signal that allows the glasses to alternately block one eye, and then the other, in synchronisation with the refresh rate of the screen. Active shutter 3D systems are used to present 3D films in some theatres, and they can be used to present 3D images on CRT, plasma, LCD, projectors and other types of video displays [49].

In passive viewers, there are generally anaglyph system and polarisation system. Anaglyph

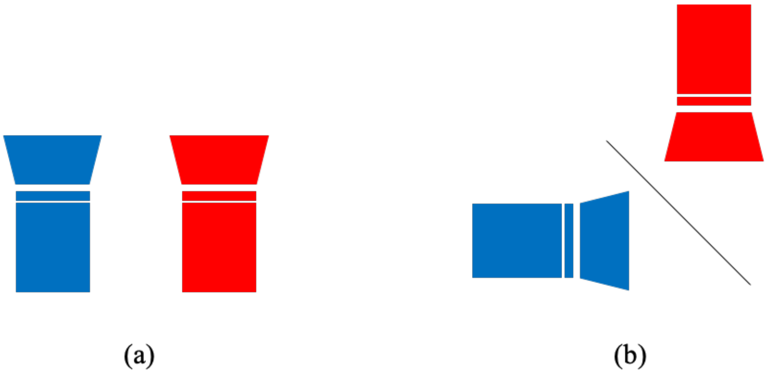


Figure 2.7: Camera rigs of stereoscopic image capture system: (a) side-by-side camera rig (b) a mirror camera rig

Figure 2.8: Still stereoscopic frame from Avatar movie (2009) [48]

system uses two different colour coded images to separate the views with coloured glasses for left eye and right eye respectively. When the anaglyph image is viewed through anaglyph glasses, each of the two images reaches one eye, revealing an integrated stereoscopic image [50]. Polarisation system projects the two separate views simultaneously and the observers wears the special polarised glasses to perceive the left and right images. The polarised glasses are used to filter views to deliver the correct view to the eyes [51]. As each filter passes only similarly polarised light and blocks the light polarised in the opposite direction, each eye sees a different image. This is used to produce a three-dimensional effect by projecting the same scene into both eyes, but depicted from slightly different perspectives.

Figure 2.9 displays the three types of stereoscopic 3D glasses: anaglyph, shutter and polarised [52].

Figure 2.9: Different stereoscopic 3D glasses: (a) anaglyph glasses, (b) shutter glasses, (c) polarised glasses
[52]

### 2.3.3 ToF 3D Imaging Technology

A Time-of-Flight (ToF) camera is a device which is capable of measuring distance between each pixel and it's correspondent in the 3D world. The distance measurement is performed by emitting a light wave which is reflected by objects in the scene and returns to the camera sensor [53]. It is a range imaging camera system that resolves distance based on the known speed of light, measuring the time-of-flight of a light signal between the camera and the subject for each point of the image. The time-of-flight camera is a class of scannerless LIDAR, in which the entire scene is captured with each laser or light pulse, as opposed to point-by-point with a laser beam such as in scanning LIDAR systems [54]. Figure 2.10 describes Time-of-Flight measurement principle with pulsed light [54].

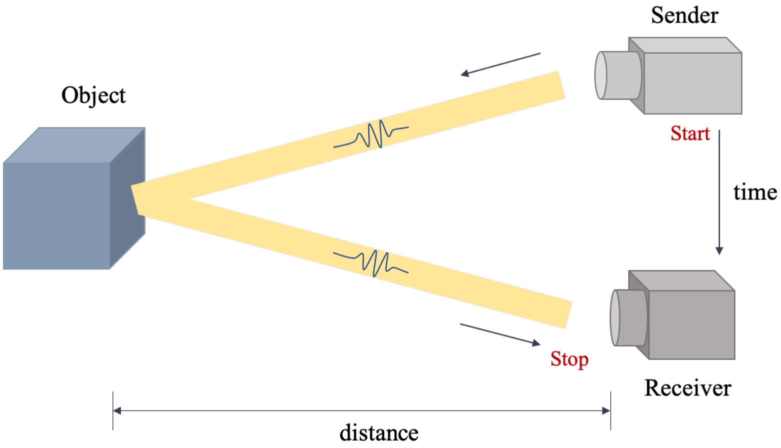


Figure 2.10: Time-of-Flight measurement principle with pulsed light
[54]

A varieties of time-of-flight camera technologies have been developed, including RF-modulated light sources with phase detectors, range gated imagers and direct Time-of-Flight imagers. The range gated imagers have a built-in shutter in the image sensor that opens and closes at the same rate as the light pulses are sent out. Because part of each returning pulse is blocked by the shutter according to its time of arrival, the amount of light received relates to the distance the pulse has travelled. This principle was invented by Antonio Medina in 1992 [55]. The direct Time-of-Flight imagers measure the direct time-of-flight required for a single laser pulse to leave the camera and reflect back onto the focal plane array. The 3D images captured using this methodology image complete spatial and temporal data, recording full 3D scenes with single laser pulse, also known as ”trigger mode”. This allows rapid acquisition and rapid real-time processing of scene information.

The common way to implement an indirect ToF camera is to illuminate the scene with modulated infrared light and utilize photonic mixing device (PMD) pixels to detect the reflections. Each pixel measures a value that indicates the correlation between the received signal and a reference signal. The so called four-phase algorithm is used to determine the distance and amplitude for each pixel [56]. Four measurements with different phase-shifted versions of the transmitted signal as reference signal are used to calculate the phase difference and the amplitude for each pixel. The distance of ToF pixels can be easily determined using the phase difference, the speed of light and the modulation frequency. Integrated pixels in the selected ToF camera implement a suppression of background illumination (SBI) circuitry [57]. This circuit prevents the pixels from being saturated during exposure to an unmodulated light source with a spectral component in the same range as the ToF working frequency, such as sunlight. However, pixels will still experience noise from ambient light. Therefore in applications with bright ambient light, the illumination power has to be selected correspondingly [58]. Figure 2.11 shows the 3D imaging principle based on PMD time-of-flight camera [54].

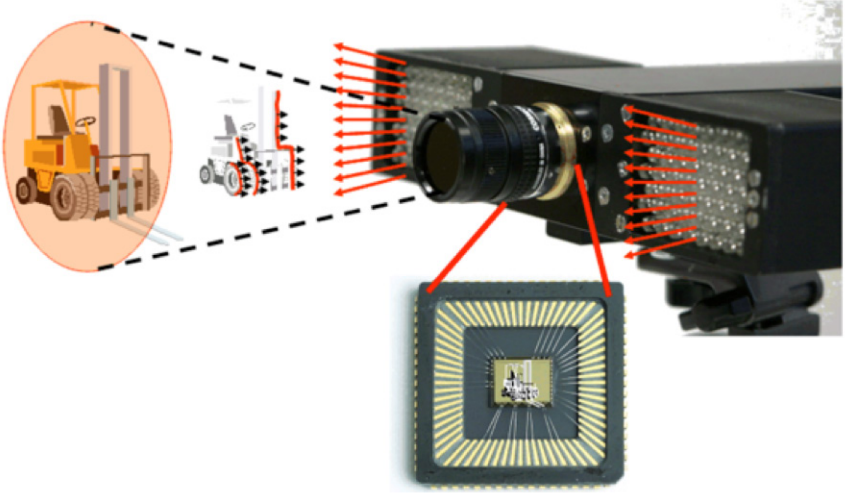


Figure 2.11: 3D imaging principle based on PMD time-of-flight camera [54]

In the field of robotics, a mobile robot implemented with ToF camera can build up a map of their surroundings very quickly, enabling it to avoid obstacles or follow a leading person. Because the distance calculation is simple, very little computational power is used. For automotive applications, ToF cameras can be used in assistance and safety functions for advanced automotive applications such as active pedestrian safety, pre-crash detection and indoor applications like out-of-position (OOP) detection [59] [60].

Figure 2.12: Kinect camera used for Xbox
[61]

There are many brands who developed different devices using Time-of-Flight camera such as Kinect for Xbox One by Microsoft and Fotonic by Panasonic. Figure 2.12 shows the Kinect camera used for Xbox [61] and Figure 2.13 shows the Fotonic camera [62].

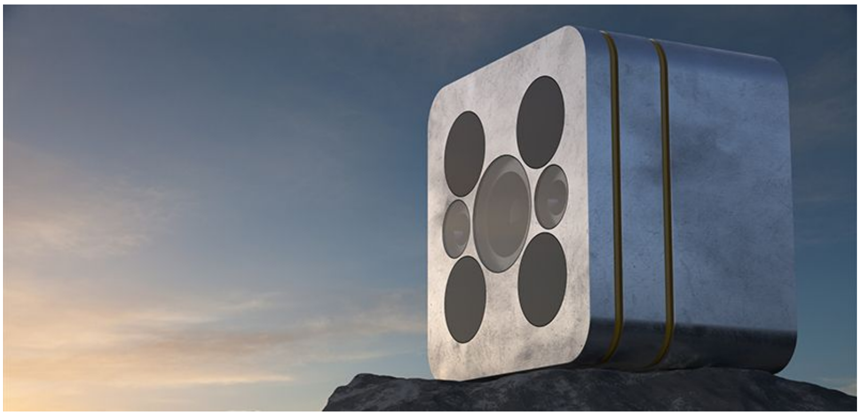


Figure 2.13: The Fotonic camera
[62]

## 2.3.4 Holoscopic 3D Imaging System

3D holoscopic imaging (also referred to as integral imaging) was first introduced by Gabriel Lippmann in 1908 [63] when he proposed integral photography as a technique for recording and reproducing 3D contents.

Holoscopic 3D (H3D) technology is a true 3D imaging technology that is based on the "fly's eye" technique that uses coherent repetition of light to build a true spatial 3D scene

[63]. It is a part of a three-dimensional imaging system. It displays 3D images within a wide viewing zone with continuous parallax and allowing full colour images within a wide viewing zone. This is a technique that uses unique optical components to create and represent a true volume spatial optical model of the object scene in the form of a planar intensity distribution. 3D holoscopic image is recorded by using a regularly spaced array of small lenslets closely packed together in contact with a recording device. The capture device is shown in Figure 2.14 (a) [63] [64].

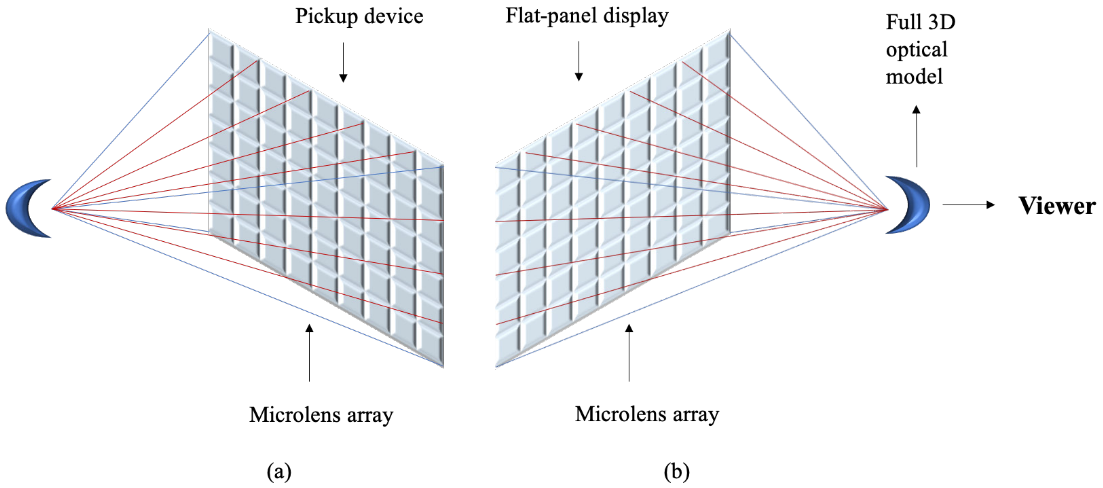


Figure 2.14: Holoscopic 3D capture and replay device (a) Holoscopic capture device (b) Holoscopic replay device
[63] [64]

Each lenslet views the scene at a slightly different angle to its neighbour and therefore a scene is captured from many view points and parallax information is recorded. The replay device of the 3D holoscopic images is consisted by placing a micro-lens array on the top of the recoded planar intensity distributions that is illuminated by diffuse white light from the rear. The object will be constructed in space by the intersection of ray bundles emanating from each of the lenslets as shown in Figure 2.14(b) [64]. In replay, the reconstructed image is pseudoscopic (inverted in depth). In the last two decades, optical and digital techniques to convert the pseudoscopic image to an orthoscopic image have been proposed [65] [66] [67] [68] [69].

However, in (a), there is a disadvantage of the camera setup that does not possess depth control. As a result, the reconstructed object appears in its original location in space. And therefore, this camera setup only can produce 3D virtual images or 3D real pseudocsopic images. Moreover, objects that very far away from a micro-lens array will suffer from poor spatial sampling on sensor pixels. Thus, this type of camera would be better suited to close imaging applications. What's more, standard methods of micro-lens manufacturing such as UV and hot embossing cause shrinkage and replication errors which can yield errors in the pitch of ± 20$\mu$m/20mm (35$\mu$m/35mm), which is particularly a large percentage error with small pitch for those micro-lens arrays [64].

To repair these two problems mentioned above, a camera setup with an objective lens and a relay lens added is shown in Figure 2.15 [64]. In order to provide depth control, an objective lens is added which allows the image plane to be close the micro-lens array, the spatial sampling of the 3D holoscopic image is depends on the number of lenses. Therefore, by reducing the size of the lenses, it is able to obtain higher resolution images.

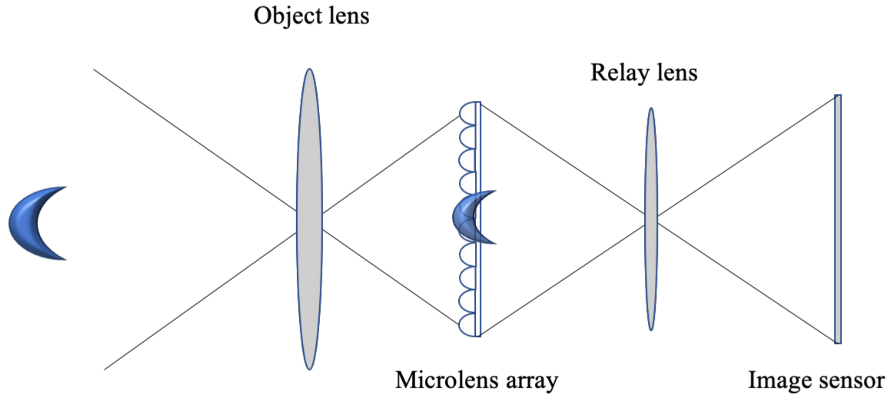


Figure 2.15: 3D holoscopic camera setup with objective and relay lenses [64]

There is a compromise exists between the number of lenses and number of viewpoint images or pixels under each lens. These pixels define discrete locations in the aperture of the objective lens from where the object can be viewed. Therefore, making the lenses smaller as fewer pixels can view it can reduce the angular information about points in the object [64].

The 3D holoscopic live images are recorded as displayed in Figure 2.16. It consists regular block pixel patterns because of the structure of the micro-lens array used in the capture process. Each block pixel pattern is called an elemental image (EI) or a micro-image. In image capture process, the 2D information such as intensity and directional can be stored in the holoscopic 3D camera, therefore all the 3D spatial information is contained in a 2D format. Each lens in micro-lens array obtains the 3D object information from slightly different directions, which records in each EI. The number of EIs is related to the number of lenses that record the 3D object from different perspectives. The recording contains the depth information embedded in a unique way. Along the outline path of the image, an image profile analysis is used to determine the intensity values and then along the horizontal line an auto-correlation is conducted [70].

To achieve above 95% fill factor, there is a square aperture fitted on the front of the camera lens. It gives rise to a regular structure in the intensity distribution by using the square aperture at the front of the camera lens and the regular structure of the square micro-lenses array in the square grid (recording micro-lens array) as shown in Figure 2.16.

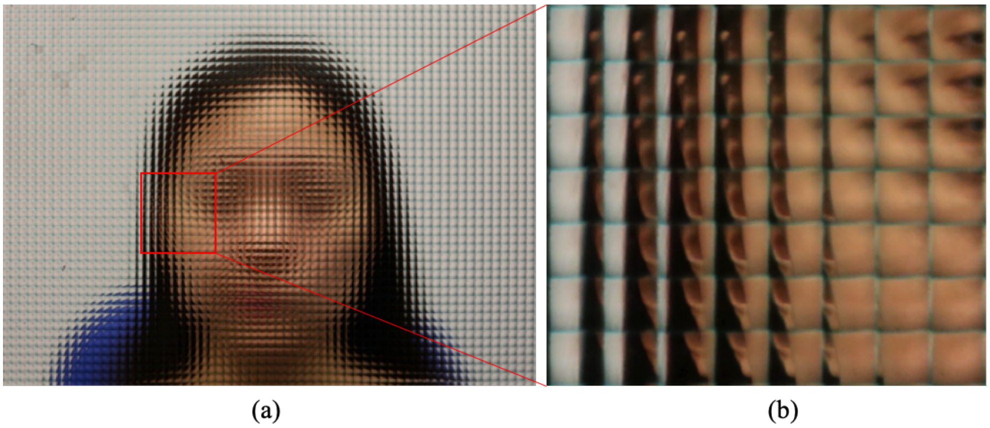


Figure 2.16: Holoscopic 3D image's block patterns (a) Recorded 3D holoscopic image (b) Magnified part

In order to transform the 3D holoscopic images consisted of block patterns to the form of 2D images, and for the following process, the collected data need to be resampled [70]. This resampled image is called the viewpoint image, and the resampling is to extract the holoscopic 3D image from one particular view direction of the scene using mathematical technique. The process of resampling is to obtain all the pixels data at the same position under different micro-lens, for example, to obtain one specific pixel from all micro-lens images at a time, and resulting in a one view direction of the recorded scene. Resampling generates a new image with different sizes and resolutions in the same image. To extract VPIs (viewpoint images) from the H3DI (holoscopic 3D images), the computational reconstruction algorithm generates VPIs independently by superimposing the pixels from all EIs (elemental images), the principle is shown in Figure 2.17 [70].

Figure 2.17 explains the principle of extraction for assumed nine VPIs from H3DI by periodically extracting pixels from the captured EI for simplicity. From Figure 2.17 (a) to (b), assume there are only $3 \times 3$ pixels under each micro-lens from $3 \times 3$ EIs. One pixel from the same position under different micro-lenses is extracted and placed in an order to form one VPI, and Figure 2.17 (c) illustrates the extraction of nine VPIs from $5 \times 3$ EIs (i.e. $m \times n$ number of VPIs).

The H3DI is defined as: $H3DI = [H3DI(m,n)]_{m=1,2,...,M,n=1,2,...,N}$, where $m$ and $n$ are the horizontal and vertical positions of the H3DI pixels respectively. The EI is the recording image under the recording micro-lens, the H3DI uses quadratic EIs that are uniformly

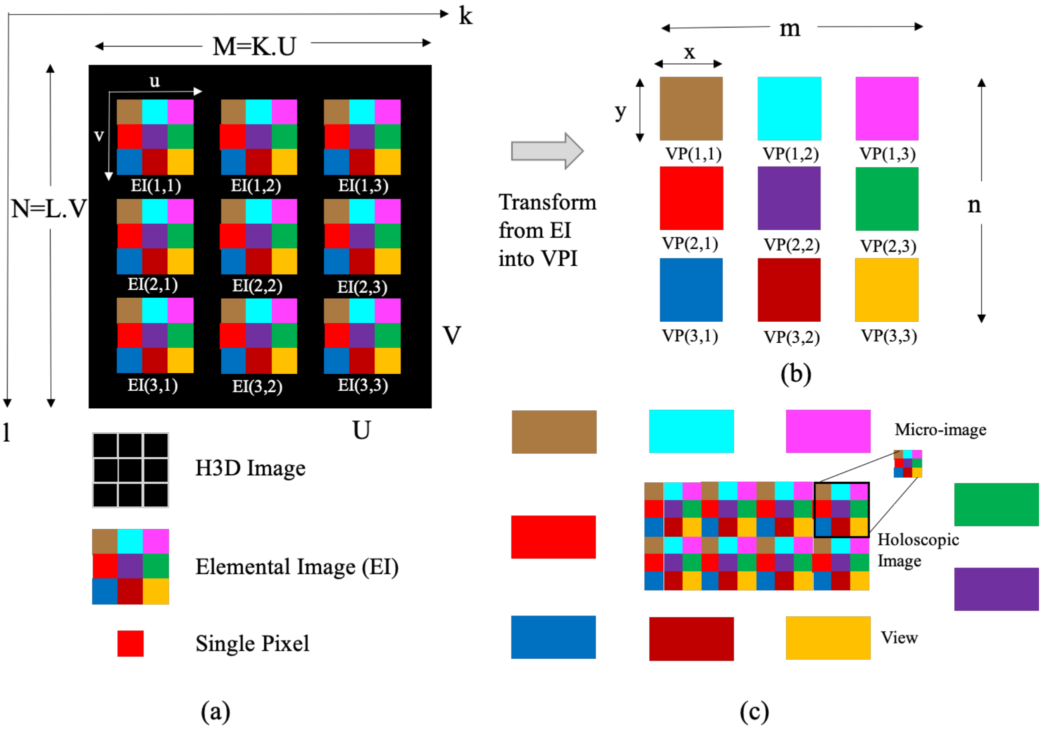


Figure 2.17: The principle of transforming EI to VPI [70]

positioned, with each EI at row $k$ and column $l$, and defined as [70]:

$$EI = [EI_{k,l}(u,v)]_{k=1,\ldots,K,l=1,\ldots,L} = H3D[k.U+u, l.V+v]_{k=1,\ldots,K,l=1,\ldots,L} \tag{2.1}$$

where $u = 1,2,\ldots,U$ and $v = 1,2,\ldots,V$ are the horizontal and vertical pixel positions within the EI. Hence, each $K \times L$ EI has a resolution of $U \times V$ pixels and has its $u,v$ coordinate system. The VPIs are formed as subsets of pixels in the H3DI and share the same relative horizontal and vertical offset to each EI centre, so the VPI is defined as [70]:

$$VPI_{u,v}(k,l) = [EI(u,v)]_{k=1,\ldots,K,l=1,\ldots,L} = H3DI(k.U+u, l.V+v) \tag{2.2}$$

The total number of EIs is $K \times L$, and a $VPI(x,y)$ is evaluated by the summation of $EI_{k,l}$, which is the intensity of the $k_th$ column and $l_th$ row of EI, thus [70]:

$$VPI(x,y) = VPI_{u,v}(k,l) = \sum_{k=1}^{K-1}\sum_{l=1}^{L-1} EI_{k,l}(x-kS, y-lS) \tag{2.3}$$

in Equation 2.3, $x$ and $y$ are the index of pixels in the x and y directions, and $S$ is an integer number of shifted pixels (pixel by pixel) in the overlapping EIs, the minimum step between the shifting distances becomes one pixel.

## 2.4 Computer Vision

Computer vision is a field includes methods for acquisition, processing, analyzing, and understanding images, which are usually high-dimensional data from the real world with the purpose of producing numerical or symbolic information, for example, in the forms of decisions [71]. The subject in the development process of this field is to replicate the abilities of human vision by electronically perceiving and understanding an image [72]. Human beings perceive 75% of the amount of information from the objective world through visual perception, so it is at most important that robot have visual perception of the environment they are operating on. By using models constructed with the help of statistics, geometry, physics, and learning theory, this image understanding can be considered as the disentangling of symbolic information from image data [73].

For our purpose, since we are dealing with large volume of visual data, the most important is to understand visual data and be able to use key features for classifying objects, summarising it and detected changes in it.

### 2.4.1 Image Features

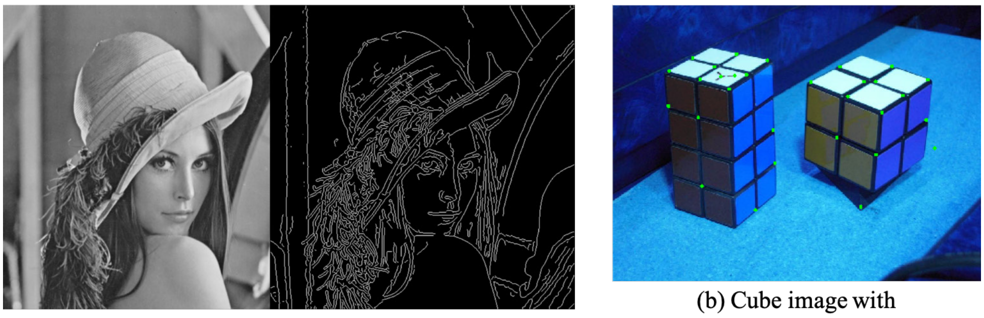
The various features of an image are colour, texture, shape or domain specific features. In pattern recognition and in image processing, feature extraction is an important step in the construction of any pattern classification and aims at the extraction of the relevant information that characterizes each class. Feature extraction is a special form of dimensionality reduction. The main goal of feature extraction is to obtain the most relevant information from the original data and represent that information in a lower dimensionality space.

In this process, relevant features are extracted from objects/ alphabets to form feature vectors. These feature vectors are then used by classifiers to recognize the input unit with target output unit. A good feature set contains discriminating information, which can distinguish one object from other objects. It must be as robust as possible to prevent generating different feature codes for the objects in the same class. The selected set of features should be a small set whose values efficiently discriminate among patterns of different classes but are similar for patterns within the same class [74]. Features can be classified into two categories:

(1) Local features, which are usually geometric (e.g. concave/convex parts, number of endpoints, branches, joints etc.)

(2) Global features, which are usually topological (connectivity, projection profiles, number of holes, etc.) or statistical (invariant moments etc.)

Figure 2.18 illustrates low-level feature detection methods, such as edge detection and corner detection.



(a) Original Lena image and Canny edge detection

(b) Cube image with
Shi-Tomasi corner detector

Figure 2.18: Low-level feature detection
[75]

The widely used feature extraction methods are:

(1) Crossing and Distances: Crossing counts the number of transitions from background to foreground pixels along vertical and horizontal lines through the character image. Distances calculate the distances of the first image pixel detected from the upper and lower boundaries of the image along the horizontal lines [76]. This is illustrated in Figure 2.19.

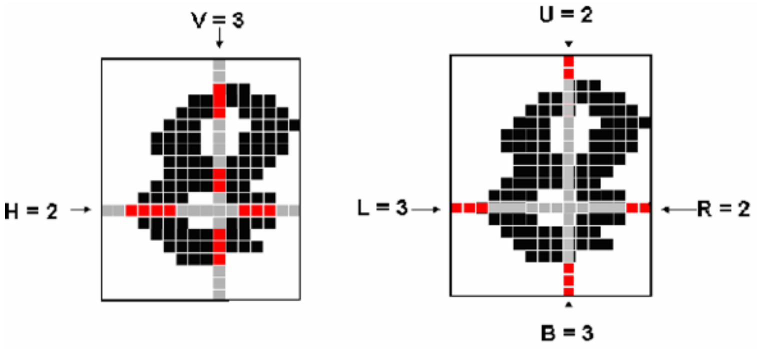


Figure 2.19: Crossing-Distance image feature extraction example
[76]

(2) Fourier Transforms: Spectral analysis using the Fourier Transform is a powerful technique for stationary time series where the characteristics of the signal do not

change with time. For non-stationary time series, the spectral content changes with time and hence time averaged amplitude spectrum found by using Fourier Transform is inadequate to track the changes in the signal magnitude, frequency or phase. The Fourier Transform is one of the oldest and most powerful tools in signal processing. This transform maps the signal in time domain to a frequency domain where certain useful features about the signal can be seen.

For Fourier transform, the general procedure is to choose magnitude spectrum of the measurement vector as the features in an n dimensional Euclidean space. One of the most attractive properties of the Fourier Transform is the ability to recognize the position shifted characters when it observes the magnitude spectrum and ignores the phase [77].

(3) Hough Transform: The Hough transform is a technique which can be used to isolate features of a particular shape within an image. Because it requires that the desired features be specified in some parametric form, the classical Hough transform is most commonly used for the detection of regular curves such as lines, circles, ellipses, etc. A generalized Hough transform can be employed in applications where a simple analytic description of a feature(s) is not possible [78].

(4) Gabor Transform: The Gabor transform is a variation of the windowed Fourier Transform. In this case the window used is not a discrete size but is defined by a Gaussian function.

(5) Wavelet Transform: This uses the gradually fine temporal or spatial sampling step size of the high frequency components, which can focus on any details of the object, with a strong selectivity of spatial position and orientation, and can capture a partial structure information corresponding to spatial and frequency [79]. As a result, wavelet transform has strong robustness of the change of image brightness and contrast as well as face posture change. Work done in [80] [81] have evidenced that the low-frequency sub-band image obtained by a multilayer wavelet decomposition of face image is a very suitable feature for face recognition.

This is used to reduce image information redundancy on account of only a subset of the transform coefficients are necessary to preserve the most important facial features, such as hair outline, eyes and mouth. Wavelet transform retains high-frequency edge information of horizontal, vertical and diagonal direction, which mainly describe the characters of human face expressions.

### 2.4.2 Image Classification

Image classification, which can be defined as the task of categorising images into one of several predefined classes, is a fundamental problem in computer vision [82]. This is one of the core problems in computer vision that forms the basis for other computer vision tasks, such as localisation, detection, and segmentation [83].

Image classification is the process of taking the input image and outputting the prediction of a single label or a probability that the input is a particular class. For example, Figure 2.20 displays images of 10 categories [84]. In order to recognise images and classify them into one of these 10 categories, we need to teach the computer how a cat or a dog look like. The more cats the computer sees, the better it gets in recognising cats. This process is known as supervised learning. We can carry this task by labelling the images, the computer will start recognising patterns present in cat pictures that are absent from other ones and will start building its own cognition [84].

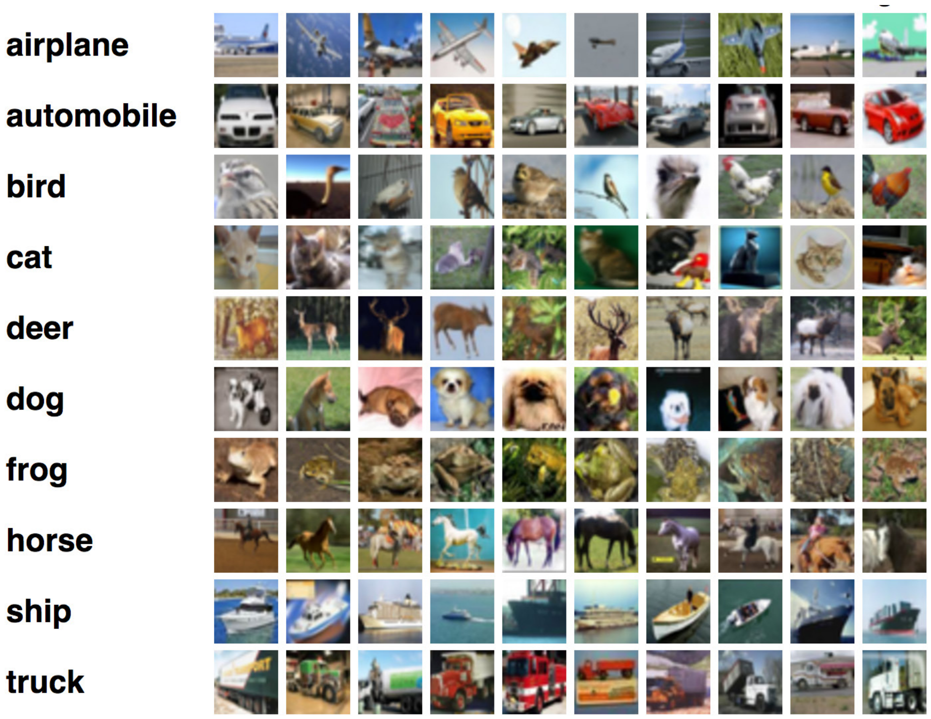


Figure 2.20: Image classification example
[84]

The main purpose of image classification is to accurately identify the features present in the image. Image classification uses both supervised and unsupervised classifications. For supervised classification, we used a trained database and human intervention. In the case of unsupervised classification, no manual intervention is required as it is entirely computer operated [85]. Supervised learning methods can be used to label the data. In this method, there are two parts of every data point : input and output. Input includes the feature variables, and output represents the desired value for this data point. With this data organisation, training algorithms can be run on a given input / output pair. This helps build a model that will predict the labelling of any future unlabelled test data [86]. Unsupervised learning methods is used when obtaining training data without labels. This will facilitate the process of inferences from the data in question. This method is based on statistical information, but can also be used for data mining. It helps to search for patterns in a given data for later grouping into different clusters [87].

Generally, a typical image classification system consists the steps [88]:

(1) Pre-processing: This step provides the processing of input data before being fed into the feature extraction step. Pre-processing techniques include the following sub-processes: filtering, normalisation, trimming, transformation, alignment, offset correcting, windowing, smoothing etc. However, the choice among these sub-processes depends on applications like: the intensity of the used brightness and color normality in face-recognition. Generally, these two techniques represent the most common ones.

(2) Features Extraction: This step implements the process of extracting features using one or more standard methods (mixing between two or more methods) like: principle component analysis, histograms of oriented gradients, local binary pattern etc. Based on mathematical computation for each type of function, some of these functions are important for a particular application.

(3) Reduction of Features Extraction: In general, the resulted features obtained from the most standard methods of feature extraction are huge. Therefore, strongest set of features need to be selected to describe input image by applying specific technique to minimise the computed feature. Initially, this minimisation issue was applicable to generate a limited number of features called active features, which were used as input data to the intelligent classification system instead of the original input data (images). Similarly, increasing the number of work features in an intelligent system can lead to increased complexity in designing the system and more time spent in the training stage.

(4) Machine learning: The images are classified based on the extracted features into predefined categories by using suitable methods.

(5) Evaluation of Classification Accuracy: This step implies determining the performance of the system by evaluating the classification process.

There are a lot of image classification techniques have been developed, such as the Decision

Tree (DT), k-Nearest Neighbours (k-NN), Support Vector Machine (SVM), and Artificial Neural Network (ANN). The former three methods are introduced below, while ANN will be introduced in following section.

Decision tree (DT): This is a non-parametric supervision method. It is a tree-similar graph of decision that all branches represent the decisions to be prepared graphically. It partition input into regular module. This technique permit the accepted and rejected of class label at all intermediate phase. The rule set provided this way is later than the classification that must be understood [89].

The k-Nearest-Neighbours (k-NN): It is a non-parametric method for classification. It is amongst the simplest of all machine learning algorithms. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common amongst its $k$ nearest neighbours [90]. For a data record $t$ to be classified, its $k$ nearest neighbours are retrieved, and this forms a neighbourhood of $t$. With or without consideration of distance-based weighting, majority voting among data records in the neighbourhood is usually used to determine the classification of $t$. However, to apply k-NN, we need to choose a suitable value for $k$, and the success of classification depends largely on this value. In a sense, the k-NN method is biased by $k$. There are many ways to choose the $k$ value, but an easy way is to run the algorithm multiple times with different $k$ values and then choose the method with the best performance [91].

Support Vector Machine (SVM) [92]: It is a classification and regression prediction tool that uses machine learning theory to maximise prediction accuracy while automatically avoiding over-fitting the data. A support vector machine can be defined as a system that uses a linear function hypothesis space in a high-dimensional feature space, trained with a learning algorithm from optimisation theory, which implements the learning bias derived from statistical learning theory[93].

### 2.4.3 Object Detection

Object Detection is a common problem in computer vision which deals with identifying and locating object of certain classes in the image. Object detection usually consists of different tasks such as face detection [94], pedestrian detection [95], etc. Due to the large number of factors that must be addressed, object detection is still a challenging issue: variety of possible objects' forms and colours, occlusions, lighting conditions, perspective etc. [96]. An example of object detection is shown in Figure 2.21 [97].

As one of the key problem in computer vision, object detection provides valuable information of images and videos and is related to many applications such as image classification

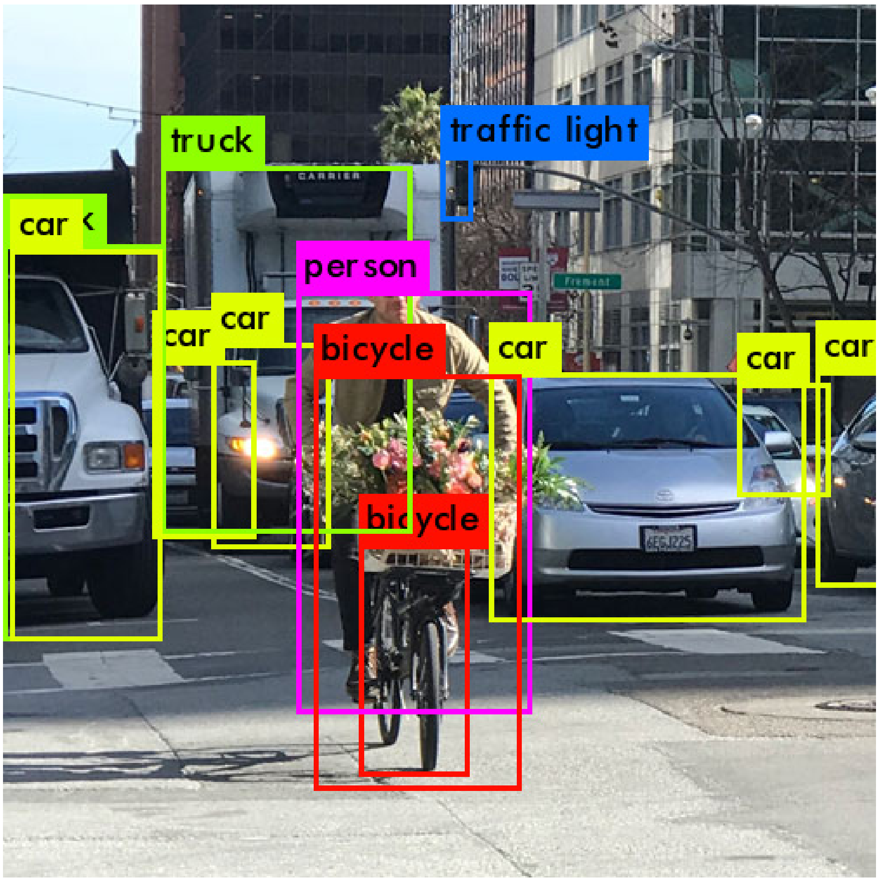


Figure 2.21: 2D perception engine for object detection and recognition for AV [97]

[98] [99], face recognition [100],human behaviour analysis [101], and autonomous driving [102] [103].

It is more general in sense that it needs to determine not only whether the object of interest exists in the image, but also the location of all its instances. Object detection is more challenging and combines these two tasks and draws a bounding box around each object of interest in the image and assigns them a class label [104]. Therefore, the pipeline of traditional object detection models can be mainly divided into three stages: informative region selection, feature extraction, and classification [105].

(1) Informative Region Selection: Since different objects may appear in the image at any position and have different aspect ratios or sizes, the natural choice is to use a multi-scale sliding window to scan the entire image. Although this exhaustive strategy can find out all possible positions of an object, its disadvantages are also obvious. This is computationally expensive due to the large number of candidate windows, and can create excessive redundant windows. However unsatisfactory regions may be produced if only a fixed number of sliding window templates are applied.

(2) Feature Extraction: Visual features that can provide a semantic and robust representation need to be extracted to recognise different objects. Scale-invariant feature transform [106], histograms of oriented gradients (HOG) [107], and Haar-like [108] features are the representative ones. This is due to the fact that these characteristics can produce representations related to complex cells in the human brain [106]. However, due to the variety of appearance, lighting conditions, and background, it is difficult to manually design robust feature descriptors to perfectly describe various objects.

(3) Classification: In addition, a classifier is required to distinguish the target object from all other categories and to make the representation more hierarchical, semantic and informative for visual recognition. In general, the supported vector machine (SVM) [92], AdaBoost [109], and deformable part-based model (DPM) [110] are good choices. Among these classifiers, DPM is a flexible model that handles severe deformations by combining object parts with deformation costs. In DPM, a graphical model is used to combine carefully designed low-level features with motion-inspired part decomposition. Discriminant learning of graphical models allows the construction of high-precision part-based models for various object classes.

There are mainly two categories of object detection methods, one follows the traditional object detection pipeline, generating region proposals at first and then classifying each proposal into different object categories. The other regards object detection as a regression or classification problem, adopting a unified framework to achieve final results (categories and locations) directly [105]. The region proposal-based object detection methods mainly include R-CNN [111], Fast R-CNN [112], and Faster R-CNN [113]. The regression / classification-based object detecion methods mainly include MultiBox [114], G-CNN [115], YOLO [116], and Single Shot MultiBox Detector (SSD) [117]. Figure 2.22 displays the structure of R-CNN [111]. This system first takes an input image, extracts around 2000 bottom-up region proposals, then computes features for each proposal using a large convolutional neural network (CNN), and finally classifies each region using class-specific linear SVMs. R-CNN achieves a mean average precision (mAP) of 53.7% on PASCAL VOC 2010.

### 2.4.4 Video Summarisation

Due to the great increase in the generation of digital videos in the last years, there is an increasingly need to develop techniques that are capable of manipulating these data in an automatic, efficient and accurate way, concerning the issues of searching, browsing,

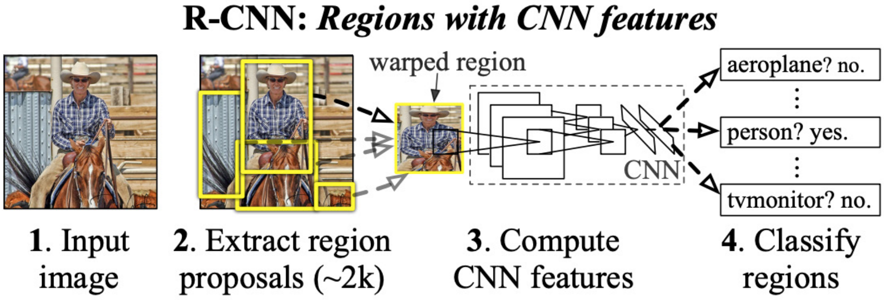


Figure 2.22: Structure of R-CNN
[111]

retrieval and content analysis. Video summarisation one such technique that aims to provide a summary of a long video for shortening the navigation and browsing the original video. The challenge of video summarisation is to effectively extract certain content of the video while preserving essential message of the original video [118].

In other words, a digital video can be defined as a collection of images that have the same dimensions, grouped according to a temporal sequence. Each of these images is known as frame, which corresponds to the smallest structural unit of a video, representing a picture captured by a camera in a given time instant of the video. The frames can be grouped into shots, which are sequences of frames, captured in a contiguous way, and that represent a continuous action in time or space. Finally, a group of shots that are semantically correlated constitutes a scene [119]. This process is illustrated in Figure 2.23 [118].

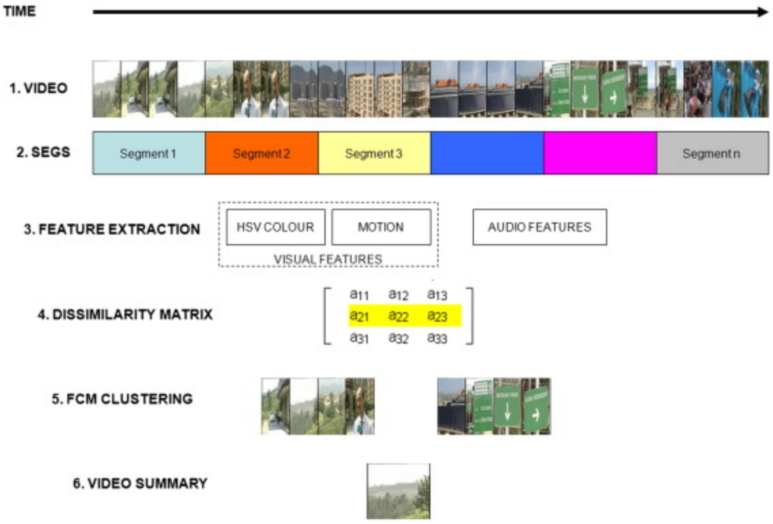


Figure 2.23: Video summarisation block diagram
[118]

In general, there are two types of video summarisation: static and dynamic.

In the first category, the summary is generated as a collection of still images denominated key frames [120], that represent the content of a video in the form of a storyboard. The advantage of this approach is in its simplicity and efficiency, usually being free of redundancies, but it may not preserve the temporal order of the selected key frames. In the second category, many segments of the video are chosen, which are then organized such that the temporal order of the video is preserved [121]. Dynamic summarisation has the main advantage of generating summaries which a higher richness of details, but it is more expensive than static summarisation approaches, besides the possible generation of redundancies.

Techniques developed for dynamic video summarisation consist of singular value decomposition [122], similarity measure optimisation [123] and attention model [124].

## 2.5 Machine Learning

Machine learning is the branch of computer science that has to do with building algorithms that are guided by data. Rather than relying on human programmers to provide explicit instructions, machine learning algorithms use training sets of real-world data to infer models that are more accurate and sophisticated than humans could devise on their own.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

Machine learning algorithms are often categorized as supervised or unsupervised.

Supervised machine learning [125]: algorithms can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

In contrast, unsupervised machine learning algorithms such as [126] are used when the information used to train is neither classified nor labelled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabelled data.

Semi-supervised machine learning algorithms such as [127] fall somewhere in between supervised and unsupervised learning, since they use both labelled and unlabelled data for training – typically a small amount of labelled data and a large amount of unlabelled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labelled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiring unlabelled data generally doesn't require additional resources. Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behaviour within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal [128].

While traditional machine learning using algorithms and models have had many successes, they can get mathematically extremely complicated and computationally very costly. As a result, combining machine learning with AI such as Artificial Neural Networks and cognitive technologies can make it even more effective in processing large volumes of information and solving very complex problems such as complex pattern recognition without the limitations mentioned above.

### 2.5.1 Artificial Neural Networks

Artificial neural networks (ANNs) are non-linear mapping structures based on the function of the human brain. They were developed initially to model biological functions. They learn from experience in a way and can rapidly solve hard computational problems, which is different than conventional computers do. An ANN is a 'black box' approach which has great capacity in predictive modelling, i.e. all the characters describing the unknown situation must be presented to the trained ANN, and the identification (prediction) is then given. They have been shown to be universal and highly flexible function approximators for any data. These make powerful tools for models, especially when the underlying data relationships are unknown [129]. Figure 2.24 following is an example of a couple of neurons [130].

In Figure 2.24, dendrites take input from other neurons in form of an electrical impulse, cell body generates inferences from those inputs and decide what action to take, and axon terminals transmit outputs in form of electrical impulse. In simple terms, each neuron takes input from numerous other neurons through the dendrites. It then performs the required processing on the input and sends another electrical pulse through the axiom into the

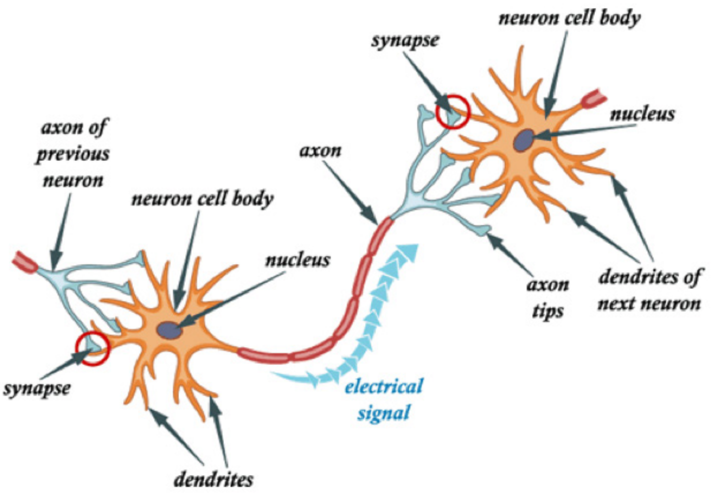


Figure 2.24: Neurons: the computer chip of human brain
[130]

terminal nodes from where it is transmitted to numerous other neurons.

An ANN is based on a collection of connected units or nodes called artificial neurons (a simplified version of biological neurons in an animal brain). Each connection (a simplified version of a synapse) between artificial neurons can transmit a signal from one to another. The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it. Example of an Artificial neuron is illustrated in Figure 2.25. where the circle represents the nucleus of the neuron, weights represent dendrites each connected to other neuron axon receiving signals from other neurons (x) and output (y) representing the axon of current neuron that sends signals to other neurons. The nucleus, computes the weight sum of the inputs to which an activation function is applied before sending the signal forward.

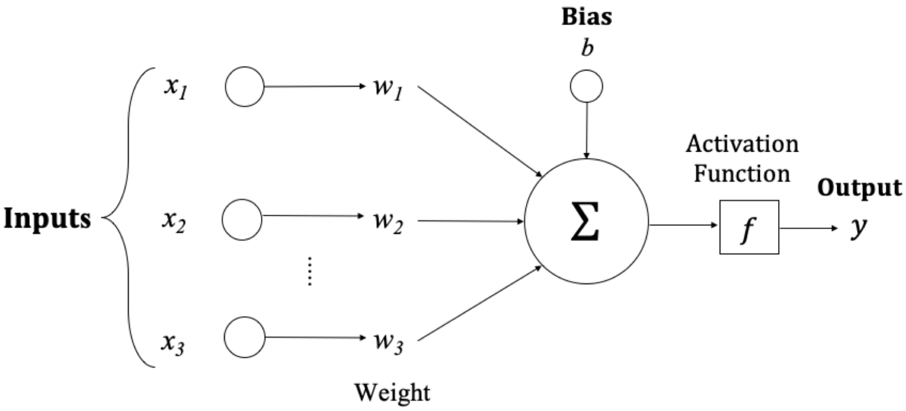


Figure 2.25: Artificial neuron

A standard neural network (NN) consists of many simple, connected processors called neurons, each producing a sequence of real-valued activations.

Artificial neurons and connections typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that only if the aggregate signal crosses that threshold is the signal sent. Typically, artificial neurons are organised in layers, as shown in Figure 2.26.

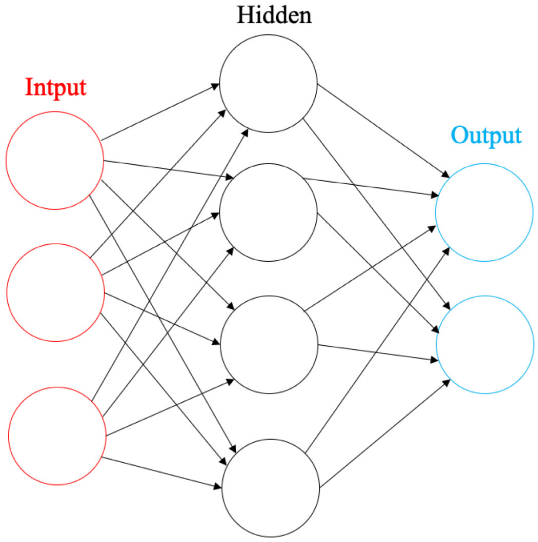


Figure 2.26: Artificial neural network architecture

Different layers may perform different kinds of transformations on their inputs. Signals travel from the first (input), to the last (output) layer, possibly after traversing the layers multiple times. Input neurons get activated through sensors perceiving the environment, other neurons get activated through weighted connections from previously active neurons. Some neurons may influence the environment by triggering actions.

Learning or credit assignment is about finding weights that make the NN exhibit desired behaviour, such as driving a car. Depending on the problem and how the neurons are connected, such behaviour may require long causal chains of computational stages, where each stage transforms (often in a non-linear way) the aggregate activation of the network [131].

One of the most common artificial neural networks types is called Back Propagation Neural Network (BPNN). Its importance was fully appreciated until 1986 [132]. Back propagation is a special case of an older and more general technique called automatic differentiation. In the context of learning, back propagation is commonly used by the gradient descent optimisation algorithm to adjust the weight of neurons by calculating the gradient of the loss function. This technique is also sometimes called backward propagation of errors, because the error is calculated at the output and distributed back through the network layers.

BPNN is a kind of forward neural network and has a multi-layer network structure. In the BPNN, the signal of the network is propagated forward through the network, and the error of the network is propagated in the reverse way. Usually the BPNN has three parts Figure 2.27: the input layer, the hidden layer and the output layer. In the forward propagation process, an input vector is first presented to the input layer, then it reaches the hidden layer and is propagated layer by layer, and the result of the hidden layer is propagated directly to the output layer, and finally result is output through the output layer. The state of each neuron in the network only affects the state of the next layer of neurons.

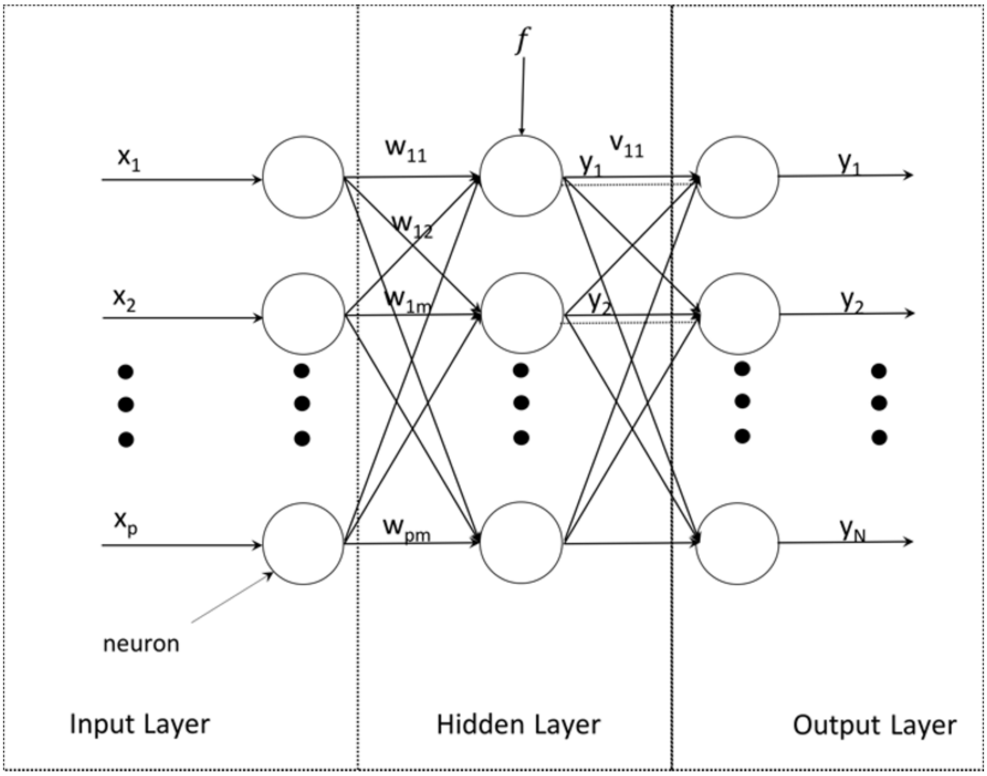


Figure 2.27: BP neural network structure

The output of the network is compared to the desired output and error value is calculated for each of the neurons in the output layer using loss function. If the network does not get the desired output value in the output layer, the network will be transferred to the reverse propagation mode. The network weights and thresholds are adjusted according to the error value thus the network output values gradually approaching the desired output value.

Although sufficient for some challenges, according to [133] argues that the number of units in network grows exponentially with task complexity. So, for challenges that very complex such as Natural language processing, and scene recognition, to be useful, a shallow network might need to be very big; possibly much bigger than a deep network. These models are called deep neural network models.

However, before going into review of deep neural network models, we will present details of the key steps or functions used in neural networks.

### Activation Function

The artificial neuron receives one or more inputs and sums them to produce an output. Usually each input is separately weighted, and the sum is passed through a non-linear function known as an activation function or transfer function. In NN the activation function of node defines the output of that node given an input or set of inputs. There are different types of activation function, used depending on application requirement [134]. These include:

(1) Sigmoid Function:

A sigmoid function is a bounded, differentiable, real function that is defined for all real input values and has a non-negative derivative at each point [135]. Given that the sum of the input values is $x$,

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} \tag{2.4}$$

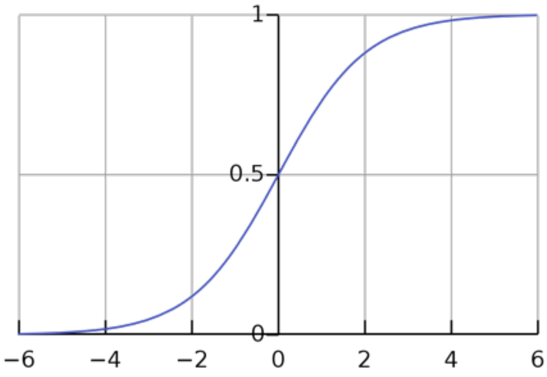


Figure 2.28: Sigmoid function plot

(1) Tan Sigmoid:

The function tansig is a transfer function. Transfer functions calculate a layer's output from its net input.

$$a = \tan sig(n) = \frac{2}{1+e^{-2n}} - 1 \tag{2.5}$$

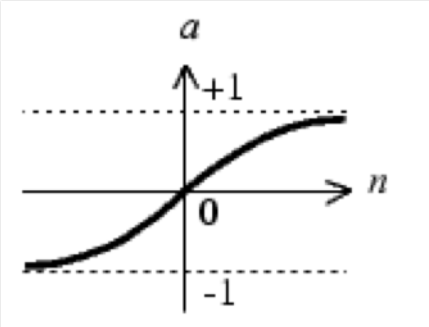


Figure 2.29: Tan sigmoid function plot

(3) ReLU:

ReLU performs a threshold operation, where any value less than zero is set to zero, , and raw output otherwise Figure 2.30. ReLU activations are the simplest non-linear activation function you can use, obviously. When you get the input is positive, the derivative is just 1, so there isn't the squeezing effect you meet on backpropagated errors from the sigmoid function. Research has shown that ReLUs result in much faster training for large networks. Most frameworks like TensorFlow and TFLearn make it simple to use ReLUs on the the hidden layers, so you won't need to implement them yourself.

$$f(x) = \begin{cases} x, & x \geq 0. \\ 0, & x < 0 \end{cases} \tag{2.6}$$

The sigmoid function has been widely used in machine learning intro materials, especially for the logistic regression and some basic neural network implementations. The derivate is easy to calculate and save times for building models.

However, for the backpropagation process in a neural network, it means that errors will be squeezed by at least a quarter at each layer. Therefore, the deeper the network is, the more knowledge from the data will be ”lost”. Some ”big” errors get from the output layer might not be able to affect the synapses weight of a neuron in a relatively shallow layer much. Due to this, sigmoid activation function has fallen out

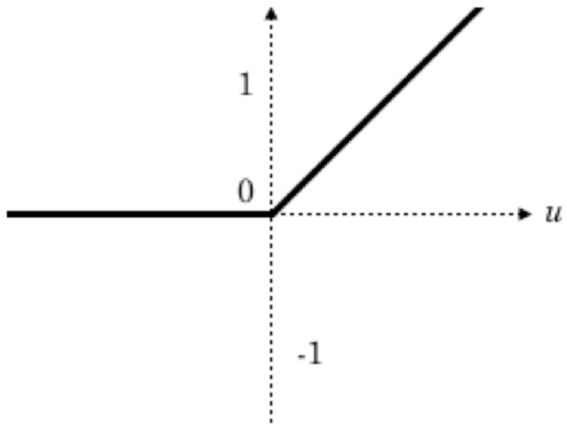


Figure 2.30: ReLU function plot

of favour on hidden units.

ReLU activations are the simplest non-linear activation function. When the input is positive, the derivative is just 1, so there isn't the squeezing effect on backpropagated errors from the sigmoid function. Research [136] has shown that ReLUs result in much faster training for large networks. Most frameworks like TensorFlow and TFLearn make it simple to use ReLUs on the hidden layers, so you won't need to implement them yourself. Figure 2.31 illustrates some of the other activation functions sometime used by researchers.

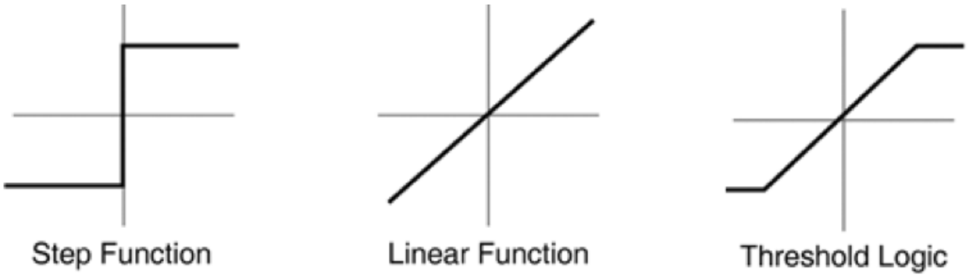


Figure 2.31: Some of the other activation functions

### Loss Function

Loss function $L(\hat{y}, y)$ is a crucial part of ANN. It is used during the training phase to measure the inconsistency between predicted value $(\hat{y})$ and actual label $(y)$. It is a non-negative value, where the robustness of model increases along with the decrease of the value of loss function. There are many types of loss function. Depending on application requirement one is preferred over another. Here we will present details of some of the main ones.

(1) Hinge Loss:

Hinge loss works well for its purposes in SVM as a classifier, since the more you violate the margin, the higher the penalty is. However, hinge loss is not well-suited for regression-based problems as a result of its one-sided error. Luckily, various other loss functions are more suitable for regression.

$$l(\hat{y}, y) = max(0, 1 - y \cdot \hat{y}) \tag{2.7}$$

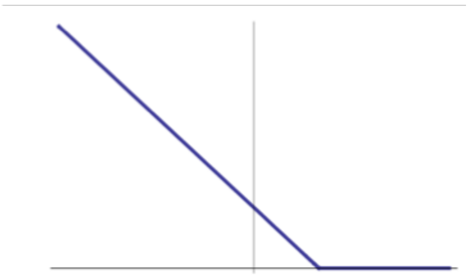


Figure 2.32: Hing loss function plot

(2) Square Loss:

Square loss is one such function that is well-suited for the purpose of regression problems. However, it suffers from one critical flaw: outliers in the data (isolated points that are far from the desired target function) are punished very heavily by the squaring of the error. As a result, data must be filtered for outliers first, or else the fit from this loss function may not be desirable.

$$l(\hat{y}, y) = (\hat{y} - y)^2 \tag{2.8}$$

(3) Mean Squared Error (MSE):

MSE, or quadratic loss function is widely used in linear regression as the performance measure, and the method of minimizing MSE is called Ordinary Least Squares (OLS).

$$l = \frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 \tag{2.9}$$

The basic principle of OLS is that the optimized fitting line should be a line which

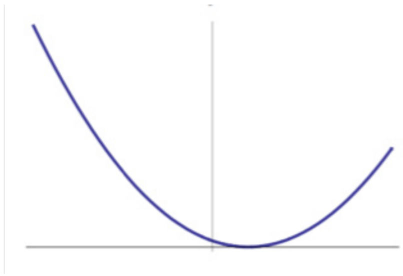


Figure 2.33: Square loss function plot

minimizes the sum of distance of each point to the regression line, i.e., minimizes the quadratic sum.

(4) Cross Entropy (CE):

Cross Entropy is commonly-used in binary classification (labels are assumed to take values 0 or 1) as a loss function (for multi-classification, use Multi-class Cross Entropy) [133] [137].

$$l = -\frac{1}{n}\sum_{i=1}^{n}[(y^{(i)}log(\hat{y}^{(i)}) + (1-y^{(i)})log(1-\hat{y}^{(i)})] \tag{2.10}$$

Cross entropy measures the divergence between two probability distribution, if the cross entropy is large, which means that the difference between two distributions is large, while if the cross entropy is small, which means that two distributions is similar to each other.

(5) Kullback Leibler Divergence (KLD):

KLD, also known as relative entropy, information divergence and gain. It is a measure of how one probability distribution diverges from a second expected probability distribution [133] [138].

KLD is a distribution-wise asymmetric measure and thus does not qualify as a statistical metric of spread. In the simple case, a KL divergence of 0 indicates that we can expect similar, if not the same behaviour from two different distributions, and a KLD of 1 indicates that the two distributions behave in such a different manner that the expectation given the first distribution approaches zero.

$$\begin{aligned} l &= \frac{1}{n}\sum_{i=1}^{n}D_{KL}(y^{(i)} \parallel \hat{y}^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} \cdot log\frac{y^{(i)}}{\hat{y}^{(i)}}) \\ &= \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} \cdot logy^{(i)}) - \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} \cdot log\hat{y}^{(i)}) \end{aligned} \tag{2.11}$$

### Back Propagation

Once we have selected the activation functions for the layers of our NN model and have decided on the type of loss function we are going to use. The final crucial part of the NN is training and learning from data. This is achieved by performing back propagation.

The core of back propagation simply consists of repeatedly applying the chain rule through all the possible paths in our network. However, there are an exponential number of directed paths from the input to the output. Back propagation's real power arises in the form of a dynamic programming algorithm, where we reuse intermediate results to calculate the gradient. We transmit intermediate errors backwards through a network, thus leading to the name back propagation. In fact, back propagation is closely related to forward propagation, but instead of propagating the inputs forward through the network, we propagate the error backwards.

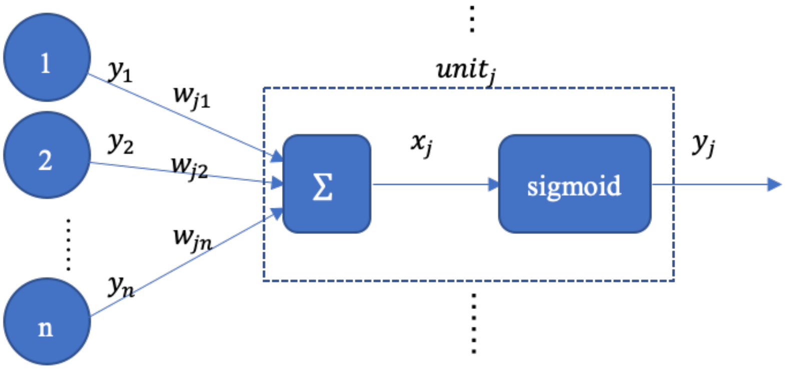


Figure 2.34: BPNN model
[132]

Let's consider the following simple example where a single neuron that takes input from n inputs and produces a single output [132]. The total input, $x_j$, to unit $j$ is a linear function of the outputs, $y_i$, of the units that are connected to $j$ and of the weights, $w_{ji}$, on these connections

$$x_j = \sum_i y_i w_{ji} \tag{2.12}$$

A unit has a real-valued output, $y_j$, which is a non-linear function of its total input. In this case a sigmoid function of total input.

$$y_i = \frac{1}{1+e^{-x_j}} \tag{2.13}$$

The total error for supervised training, $E$, is defined as the loss function MSE,

$$E = \frac{1}{2}\sum_{c}\sum_{j}(y_{j,c} - d_{j,c})^2 \tag{2.14}$$

where $c$ is an index over cases (input-output pairs), $j$ is an index over output units, $y$ is the actual state of an output unit and $d$ is its desired state. The backward pass starts by computing $\frac{\partial E}{\partial y}$ for each of the output units. Differentiating Equation 2.14 for a particular case, $c$, and suppressing the index $c$ gives

$$\frac{\partial E}{\partial y_j} = y_j - d_j \tag{2.15}$$

According to the chain rule

$$\frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{dx_j} \tag{2.16}$$

Differentiating Equation 2.13 to get the value of $\frac{dy_j}{dx_j}$ and substituting gives

$$\frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j} \cdot y_j(1 - y_j) \tag{2.17}$$

According to the chain rule

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial x_j} \cdot \frac{dx_j}{dw_{ji}} \tag{2.18}$$

From Equation 2.12 we can compute

$$\frac{\partial E}{\partial w_{ji}} = y_i \tag{2.19}$$

Thus

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial x_j} \cdot y_i \tag{2.20}$$

And for the output of the $i^t h$ unit the contribution to $\frac{\partial E}{\partial y_j}$ resulting from the effect of $i$ on $j$

is

$$\begin{aligned}\frac{\partial E}{\partial x_j} \cdot \frac{\partial x_j}{\partial y_i} &= \frac{\partial E}{\partial x_j} \cdot w_{ji} \\ \therefore \frac{\partial E}{\partial y_i} &= \sum_j \frac{\partial E}{\partial x_j} \cdot w_{ji}\end{aligned} \tag{2.21}$$

This is how to compute $\frac{\partial E}{\partial y}$ for any unit in the penultimate layer when it is given for all units in the last layer. Therefore, we can iterate this procedure to compute this term for successively earlier layers, computing $\frac{\partial E}{\partial w}$ for the weights as we go.

The simplest version of gradient descent is to change each weight by an amount proportional to the accumulated $\frac{\partial E}{\partial w}$

$$\Delta w = -\varepsilon \frac{\partial E}{\partial w} \tag{2.22}$$

To sum up, there are 6 steps to training a ANN:

(1) Select a network architecture, i.e. number of hidden layers, number of neurons in each layer and activation function

(2) Initialize weights randomly

(3) Use forward propagation to determine the output node using activation functions

(4) Find the error of the model using the known labels using a loss function

(5) Back-propagate the error into the network and determine the error for each node

(6) Update the weights to minimize gradient

Now that we have got fairly good understanding on ANN, we can discuss Deep Neural Networks (DNN).

### 2.5.2 Deep Learning Neural Network

Deep learning (also known as deep structured learning, hierarchical learning or deep machine learning) is a machine learning technique that performs learning in more than two hidden layers, Figure 2.35 shows the structure of layers of both ANN and DNN [139]. It works on unsupervised data and is known to provide accurate results than traditional machine learning algorithms.

Choosing a deep model encodes a very general belief that the function we want to learn should involve composition of several simpler functions. This can be interpreted from

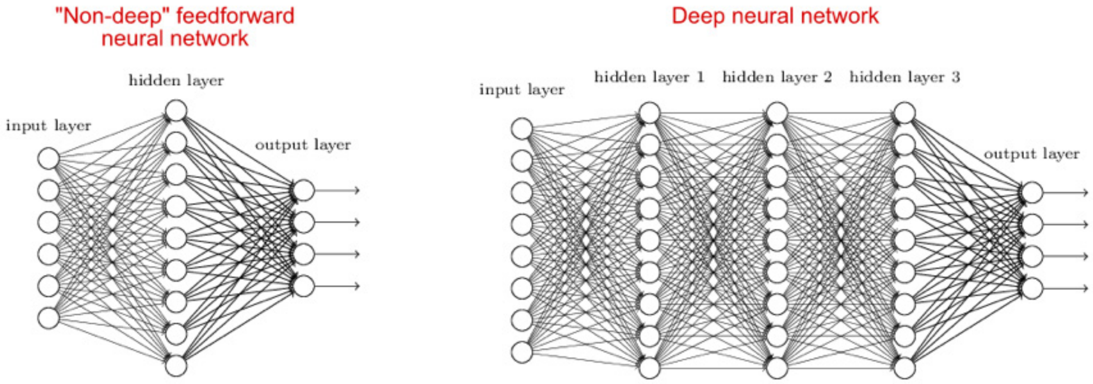


Figure 2.35: Example of ANN and DNN
[139]

a representation learning point of view as saying that we believe the learning problem consists of discovering a set of underlying factors of variation that can in turn be described in terms of other, simpler underlying factors of variation [133]. Therefore, DNNs consist multiple layers of nonlinear processing units (hidden layers). Each successive layer of DNN uses the output from the previous layer as input.

Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics [140]. Figure 2.36 illustrates example of very deep neural networks released by Google [141].

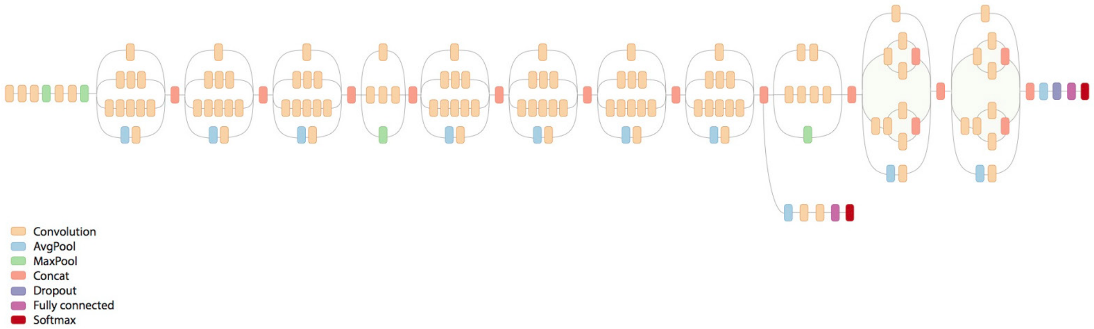


Figure 2.36: Google deep CNN model-Inception-v3
[141]

DNN discovers intricate structure in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer.

Unsupervised feature extraction is also an area where deep learning excels. Feature extraction is when an algorithm is able to automatically derive or construct meaningful features of the data to be used for further learning, generalization, and understanding. The burden is traditionally on the data scientist or programmer to carry out the feature extraction process in most other machine learning approaches, along with feature selection and engineering.

Feature extraction usually involves some amount dimensionality reduction as well, which is reducing the amount of input features and data required to generate meaningful results. This has many benefits, which include simplification, computational and memory power reduction, and so on. There are also other types of DNN, which include the mostly commonly used Convolutional Neural Networks (CNN) mainly used for image and video [142] and Recurrent Neural Networks mainly used for sequential data recognition like speech and sentences. Since we will work with H3D images, we will give brief overview of the CNN models.

### Convolutional Neural Network

CNNs have revolutionized the computational pattern recognition process [136]. Prior to the widespread adoption of CNNs, most pattern recognition tasks were performed using an initial stage of hand-crafted feature extraction followed by a classifier. The important breakthrough of CNNs is that features are now learned automatically from training examples. The CNN approach is especially powerful when applied to image recognition tasks because the convolution operation captures the 2D nature of images. By using the convolution kernels to scan an entire image, relatively few parameters need to be learned compared to the total number of operations.

While CNNs with learned features have been used commercially for over twenty years, their adoption has exploded in recent years because of two important developments. First, large, labelled data sets such as the ImageNet [143] for Large Scale Visual Recognition Challenge (ILSVRC). Visual geometry group image for face descriptors [144],CFIR dataset for different classes of objects [99] and MNIST dataset of hand written numbers [145] are now widely available for training and validation. Second, CNN learning algorithms are now implemented on massively parallel graphics processing units (GPUs), tremendously accelerating learning and inference ability.

Deep CNNs work by consecutively modelling small pieces of information and combining them deeper in network. One way to understand them is that the first layer will try to detect edges and form templates for edge detection. Then subsequent layers will try to combine them into simpler shapes and eventually into templates of different object positions, illumination, scales, etc. The final layers will match an input image with all the templates and the final prediction is like a weighted sum of all of them. So, deep CNNs are able to model complex variations and behaviour giving highly accurate predictions.

In general, the whole CNN architecture consists of two parts: image processing part that performs feature extraction and dimensionality reduction and then a fully connected ANN that produces the output illustrated in Figure 2.37 [146]. An input image of a traffic sign is filtered by 4 5 × 5 convolutional kernels which create 4 feature maps, these feature maps are subsampled by max pooling. The next layer applies 10 5 × 5 convolutional kernels to these subsampled images and again we pool the feature maps. The final layer is a fully connected layer where all generated features are combined and used in the classifier (essentially logistic regression).

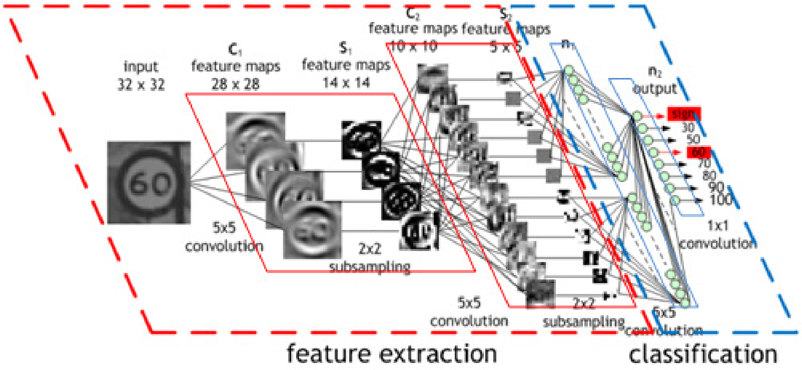


Figure 2.37: Example of a CNN model
[146]

## 2.6 Summary

In this chapter, an overview of existing sensor technologies along with imaging technologies used in automation is provided. The depth information is very important in automation. However, to achieve accurate depth information, often complex processes or very expensive sensors are used. Based on this, a new camera technology that provides both 2D and 3D information called H3D camera system is proposed to use. Compared with traditional perception sensors like LiDAR and 2D camera, H3D camera wins because of its simplicity, for example, for forward-facing direction, H3D camera is the only sensor needed instead of multiple sensors. Hence, its advantage is also in price, being consists of normal camera and holoscopic 3D components, it is cheaper than multiple sensors or expensive sensors. Most importantly, it is 3D. 3D images have become quite common in many fields and many ways to visualise 3D data which provides detailed and valuable information about the environment or examined objects. Traditional 2D image processing is used to analysis construction, product inspection or quality control. Compare with 2D processing, 3D imaging processing provides the important “depth information” to imitate third coordinate of real world.

In this chapter a literature review of computer vision approaches required to pre-process the H3D data is also provided before they are used in the proposed deep learning model. As it is detailed, there are three key pre-processing approaches image classification, object detection and video summarisation that can be used individually or jointly as a pre-processing step of the H3D data before they are applied to the deep learning model.

This chapter also provides an overview of machine learning and deep learning approaches. The key functions used in neural networks are detailed in order to understand neural networks. Examples of deep learning neural networks such as AlexNet and YOLOv3 will be provided in following chapters for training applied on H3D data.

# Chapter 3

# Holoscopic 3D Sensor Prototype

## 3.1 Introduction

In this chapter, the H3D camera sensor used in this thesis will be introduced. The inspiration and design of H3D camera layout is illustrated in Section 3.2, then every main component of this H3D camera is introduced in Section 3.3. After each component manufactured, the H3D camera is installed and calibrated which is explained in Section 3.4, and implemented for experimental use, the assembly of H3D camera is illustrated in Section 3.5. Section 3.6 comes the summary of this chapter.

## 3.2 Design of H3D Camera Layout

The H3D camera sensor implemented in this thesis is inspired from the H3D camera developed by 3DVIVANT Team at Brunel University. This camera setting has been developed and used to capture holoscopic 3D images for processing. Figure 3.1 illustrates the prototype of this holoscopic 3D camera [147]. The camera is built with the 5.6k sensor of the Canon 5D Mark2 (C5D M2) digital single-lens reflex (DLSR) camera. The main components of the holoscopic 3D camera are prime lens, micro-lens array, relay lens and digital camera sensors. The micro-lens array (MLA) is mounted in adjacent to the camera sensor as shown in Figure 3.1. The prime lens image plane is formed in front of the micro-lens array, which allows the micro-lens array to capture the positions from different perspectives in the scene.

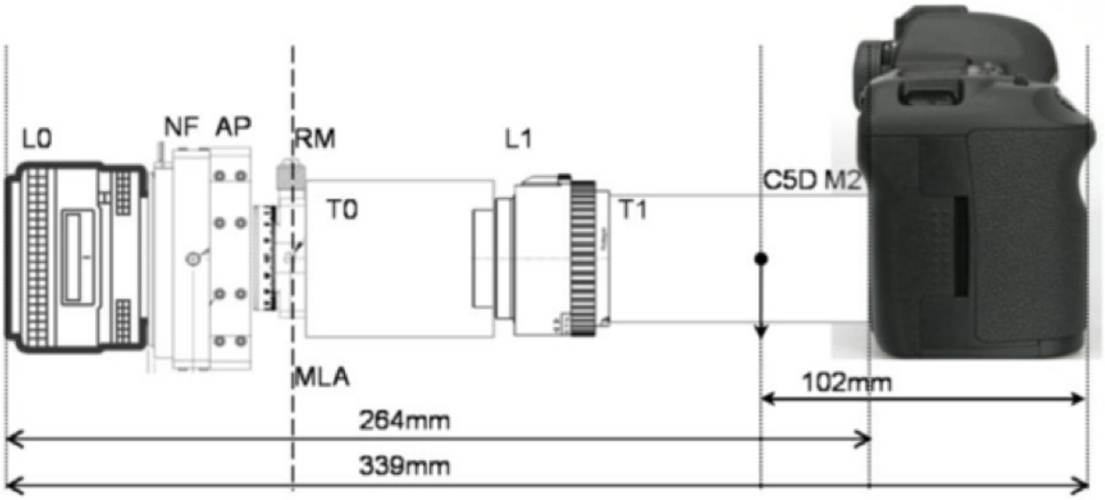


Figure 3.1: Holoscopic 3D camera prototype with Canon 5.6k sensor [147]

With the inspiration of the design of this H3D camera, a new holoscopic 3D camera is developed using a different camera sensor for this thesis. Figure 3.2 displays this design and layout. The key components of this H3D camera will be detailed in the following section.

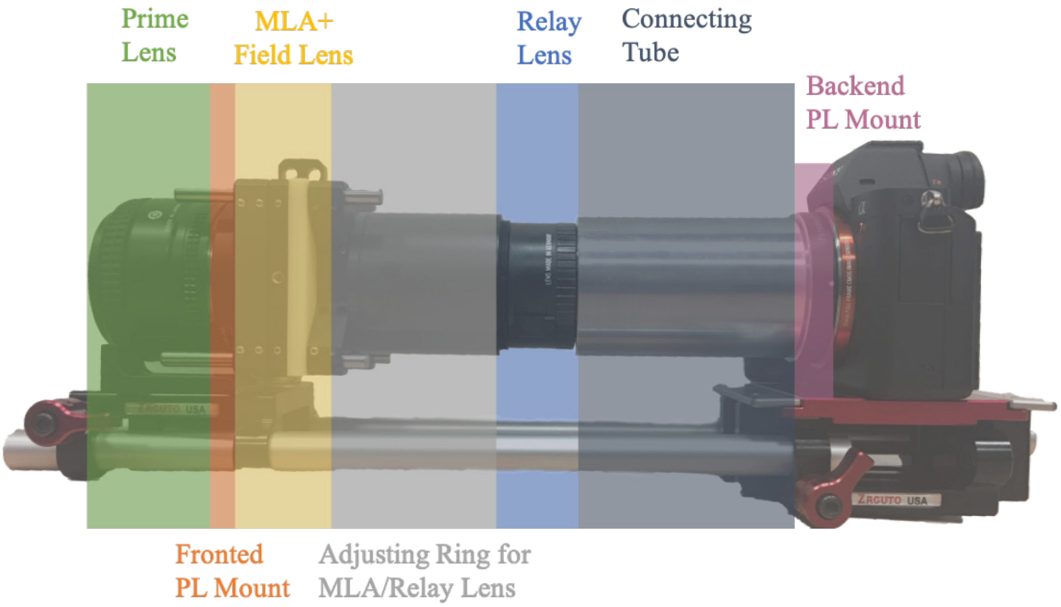


Figure 3.2: Holoscopic 3D camera design and layout

## 3.3 H3D Camera Components Design

In this thesis, a Sony alpha 7R camera as shown in Figure 3.3 is used because it has 35 mm full-frame image sensor and 4K HD as well. The 2D resolution of this camera is 7360 × 4912 pixels. This outstanding performance ensures ultra-good quality of videos and images captured in testing part. Due to a different camera is used, the measurement data are slightly different from Figure 3.1. The tube connecting Sony camera and relay lens is measured 83 mm, and the distance between the relay lens and the micro-lens array is approximately 75 mm to make the focal point exactly locate at the focal point of convex lens. Nikon Nikkor AF 35 mm f/2.0 is used as the prime lens and is shown in Figure 3.4. A Rodenstock Rodagon-N APO 50 mm f2.8 Enlarging Lens is used as the relay lens, and is shown in Figure 3.5.

Figure 3.3: Sony alpha 7R camera
[148]

Figure 3.4: Nikon Nikkor AF 35mm f/2.0 lens
[149]

According to the shape of a whole piece of micro-lens array and the convex lens, and in order to protect the completeness of micro-lens array, the previous rotating cage is not

Figure 3.5: Rodenstock Rodagon-N APO 50mm f2.8 Enlarging Lens
[150]

suitable for this camera layout. Thus, a new lens holder of micro-lens array and convex lens is designed and modelled in Solidworks. Figure 3.6 shows the sketch of the lens holder. Figure 3.7 shows the 3D printing lens holder when the two pieces assembled together. The round space in centre is to place the convex lens and the square space is to place the micro-lens array. The length of the side of the square micro-lens array is 51.2 mm, and the pitch size of micro lens is 125 $\mu$m.

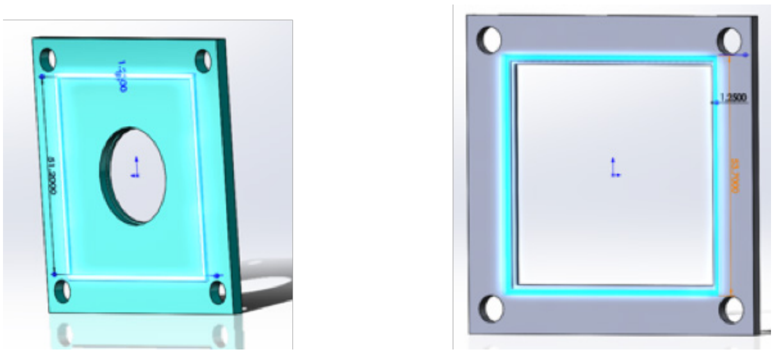


Figure 3.6: Sketch of the lens holder

The image fill factor of holoscopic 3D images is an important issue which is defined as the ratio between the effective image area and entire area of the display device [151]. Due to the structure of micro-lenses being square-shaped, the shape of micro images is square as well, the prime lens aperture was also required to be square to enable the sensor space to be used more effectively. With the aim of improving fill factor and eliminating loss of data and vignetting, various sizes of square apertures are needed to match the corresponding micro-lens pitch. Because the customised square aperture need to be attached to the prime lens, the size of outer circle should be exactly match the size of prime lens, and the size of inner square should be various to match the corresponding micro-lens pitch. Besides, different settings of aperture of both prime lens and relay lens require in different sizes of

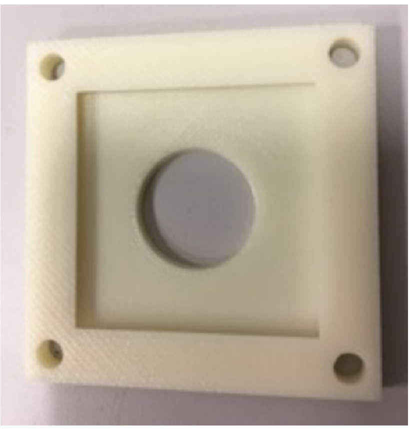


Figure 3.7: 3D printing lens holder

square aperture to acquire appropriate light.

As is mentioned above, the size of outer circle should be exactly match the size of prime lens and it is 49.26 mm in diameter via measurement, and the size of inner square should be various to match the corresponding micro-lens pitch. Therefore 7 various square apertures vary from 2 mm to 5 mm of the side length of the square hole, and every 0.5 mm by step. These plastic pieces of square apertures were printed using 3D printing machine to make sure they can attach the prime lens smoothly and standing well. Figure 3.8 displays some of the various square apertures with various sizes of the hole.

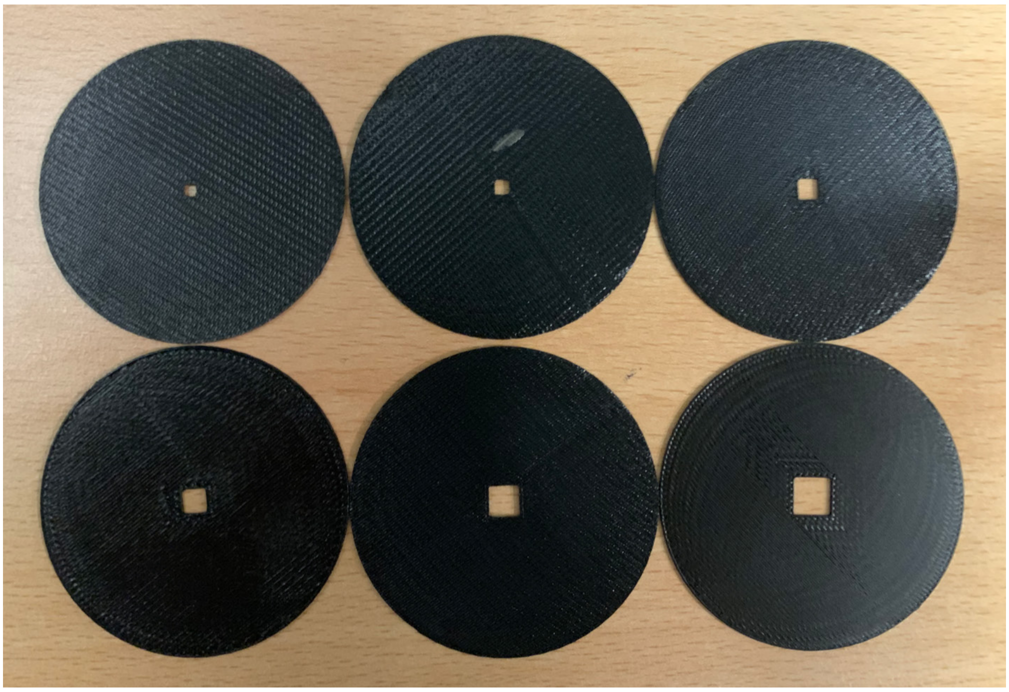


Figure 3.8: Square apertures to match the corresponding micro-lens pitch

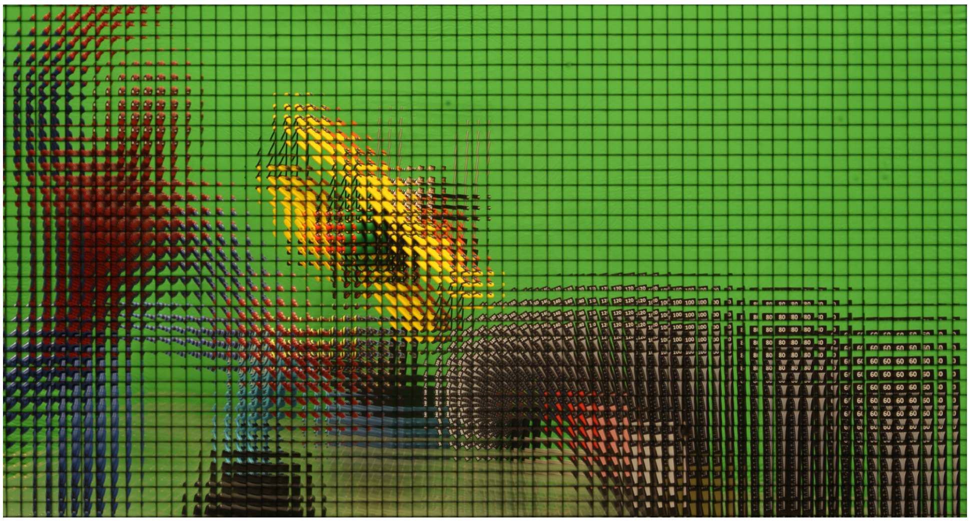


Figure 3.9: Reference for H3D image calibration
[147]

## 3.4 H3D Camera Calibration

After repeated experiments of adjusting the camera settings and aperture, the size of 3.5 mm of the side length of the square hole is determined to be the suitable square aperture when the aperture of prime lens is 2, and the aperture of relay lens is 2.8. This size of hole makes each micro-image block no overlapping or black borders, and there is very few dark corners in micro-image blocks. These settings can provide a relatively higher image fill factor under the existing equipment condition, which gives the original holoscopic images a good quality for following processing.

As is mentioned above, the tube connecting relay lens and the micro-lens array is approximately 75 mm to make the focal point exactly locate at the focal point of convex lens. This adjustment of the accurate length of lens tube is the most important section of calibration in order to achieve clear and crispy image. Taking the image of Figure 3.9 published in [147] as calibration reference, Figure 3.10 shows a good calibration with crispy image while Figure 3.11 shows four different situations of bad calibration. In order to determine the accurate length of the lens tube, this process in calibration is finalised by repeated experiments of adjusting the lens tube and shooting images.

The main task of calibration is to achieve clear and crispy details (i.e. the letters) in the image, which proves that the focal point locates exactly at the focal point of convex lens in reverse. The inaccurate adjustment of connecting tube makes the image blur which is shown in Figure 3.11 (a). The other task of calibration is to ensure the piece of square aperture is horizontally and vertically aligned, that makes every micro-image block a square. If the square aperture is slanted it will make micro-image a parallelogram which causes information loss, which is shown in Figure 3.11 (b). The adjustment of lens aperture

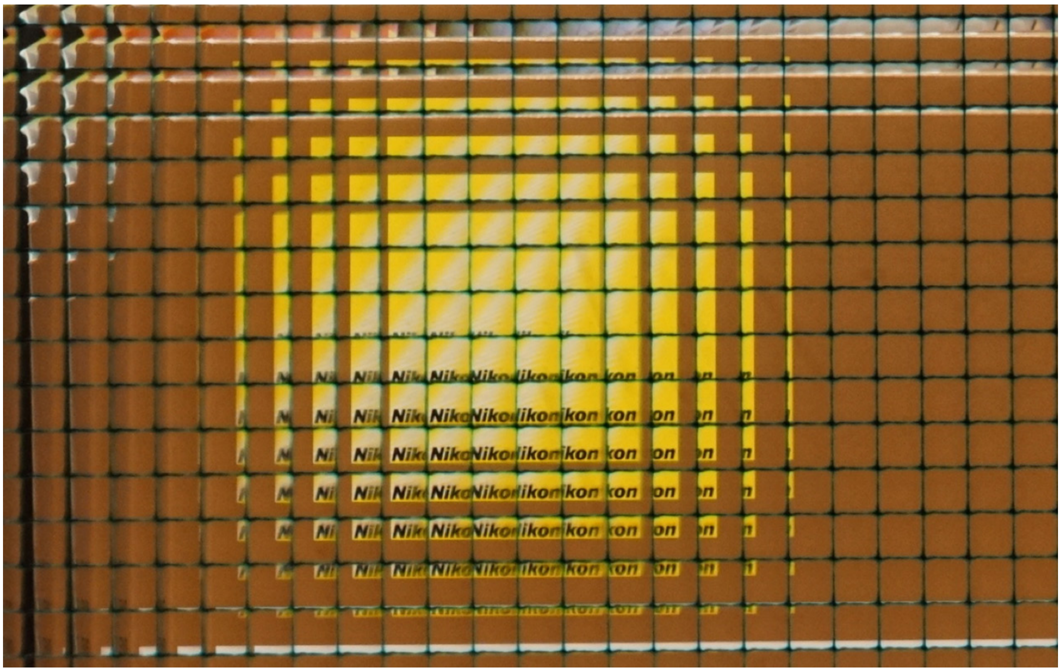
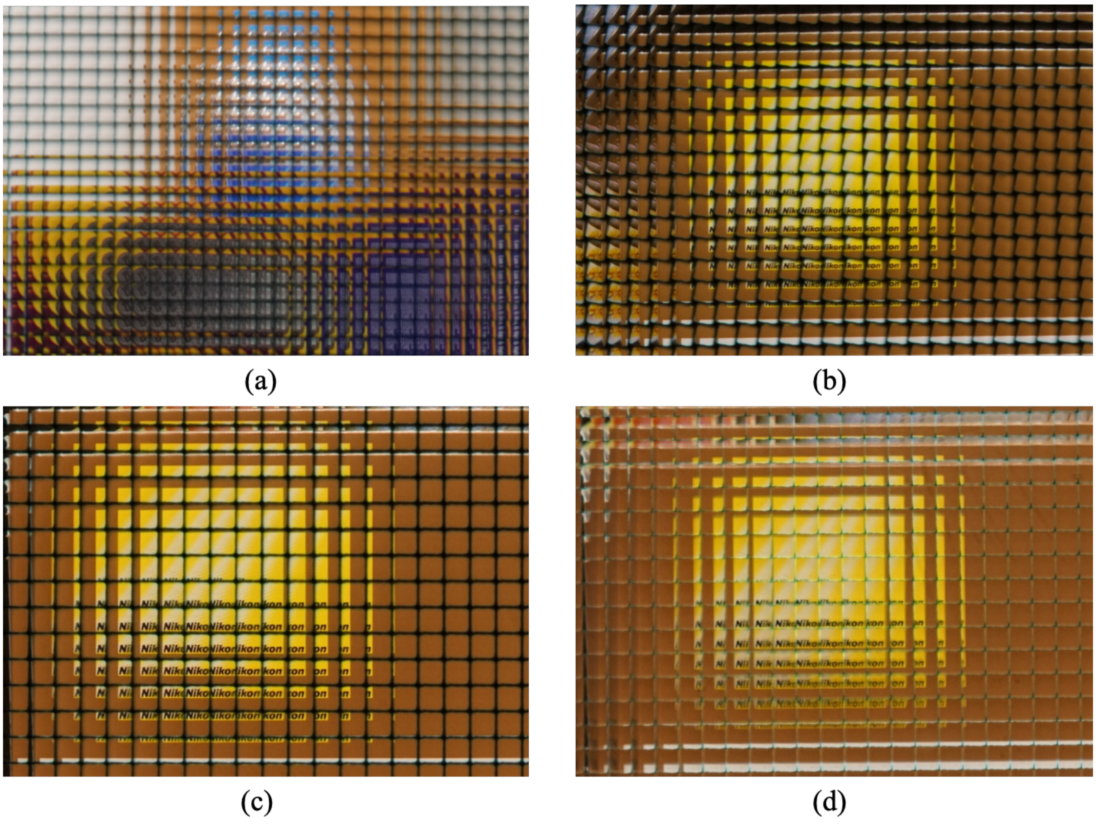


Figure 3.10: Example of good calibration



(a)

(b)

(c)

(d)

Figure 3.11: Examples of bad calibration: (a) Blur calibration (b) Slanted square aperture (c) Wide bundary (d) Ghosting

and the ISO value of camera is equally important to make appropriate light get through. Figure 3.11 (c) displays the wide boundary of elemental image caused by insufficient light, and Figure 3.11 (d) displays the ghosting caused by overexposure. These situations of bad calibration can all result in loss of data, and lead to problems in image processing. Figure 3.12 displays the calibrated H3D camera which will be used for the following experiments.

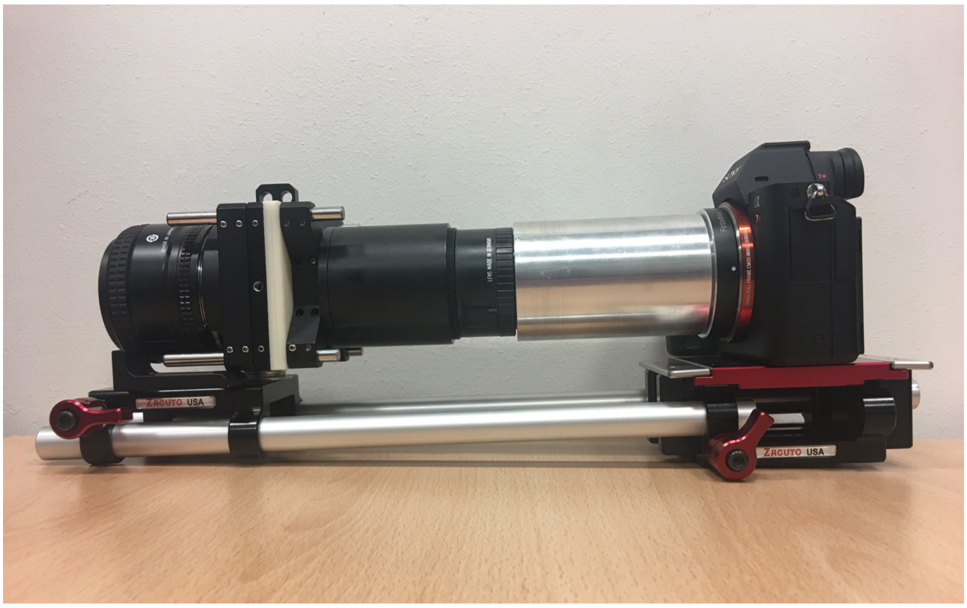


Figure 3.12: The calibrated H3D camera

## 3.5 H3D Camera Assembly

In order to conduct the experiments of data collection, the H3D camera sensor should be assembled on cars or tripod according to different requirements. As is shown in Figure 3.13, the HPI Baja 5B racing car is developed as one of the experimental car in this thesis. The 1/5 large scale off-road racing buggy is battery powered, and the H3D camera is assembled on the top of its roll cage as it is shooting the scene forward.

Using the integrated RC car, we performed a simple data acquisition test inside the lab in Tower C 005 Brunel University. The aim of the test is to ensure the gesture of H3D camera on top of RC car to achieve a good view of surroundings. The test route is marked by placing coloured tape on the ground as illustrated in Figure 3.14. Figure 3.15 shows samples of data collected inside the lab for some tests.

Beside that, the H3D camera is also assembled on top of a real car for real road testing and data acquisition. The H3D camera is fixed on rig and use the pressure of suction cup to ensure it standing on the top of the car tightly. Figure 3.16 displays the integration of H3D camera on a real car.

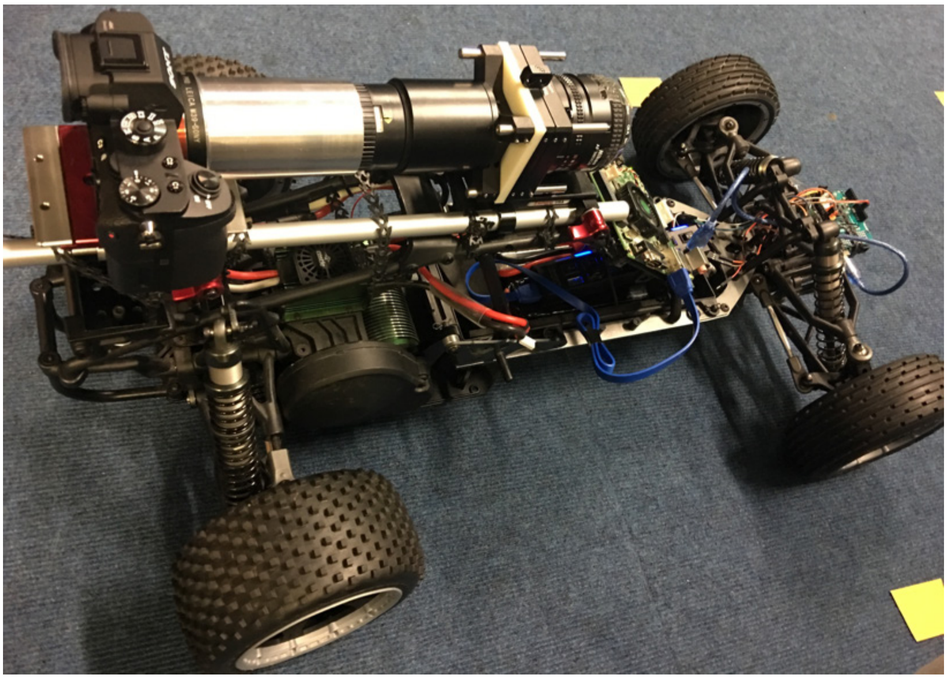


Figure 3.13: Integration of H3D on RC car

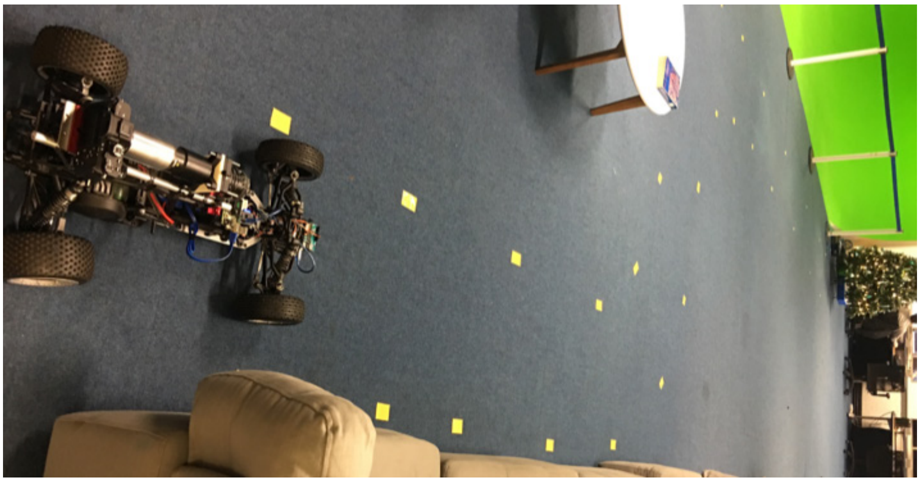


Figure 3.14: Data Collection Test Setup

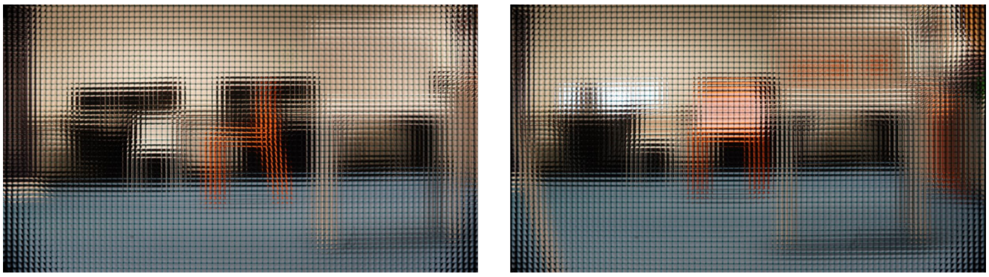


Figure 3.15: Samples of data collected in the lab

Figure 3.17 contains images captured on real road, which shows different scenes of surroundings than in the lab. This test also aims to adjust the gesture of the H3D camera on top of the car to achieve a proper view.

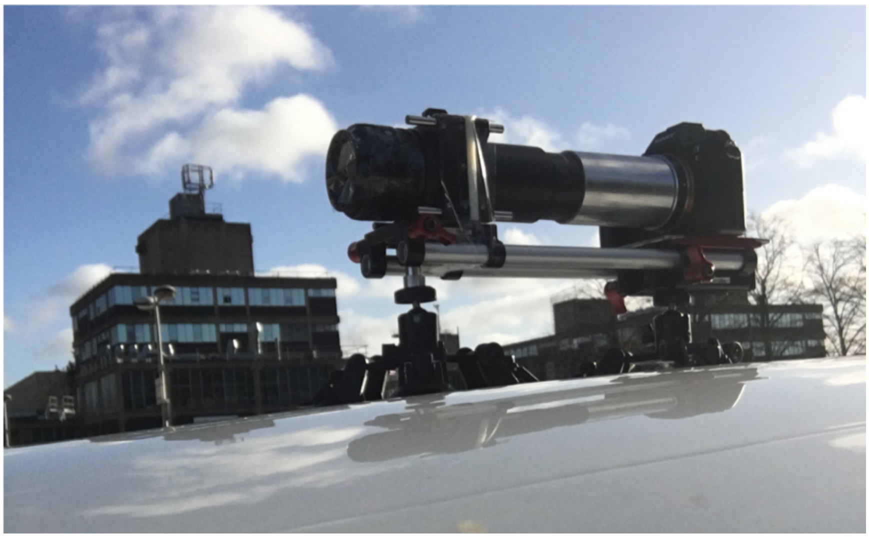


Figure 3.16: Integration of H3D on car

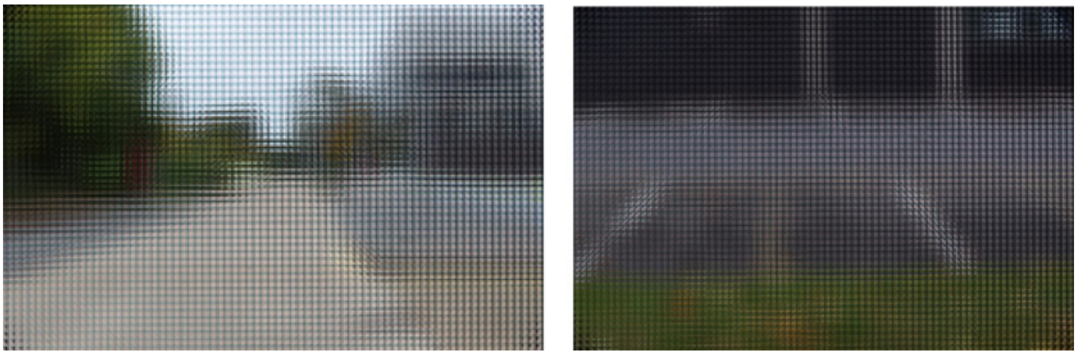


Figure 3.17: Samples of data collected on main road

## 3.6 Summary

In this chapter, a new camera sensor that provides both 2D and 3D information - H3D camera system - is proposed to use for this thesis which is inspired from the existing sensor technology. Based on the literature review have done, the design of the camera to achieve H3D image has been illustrated and detailed in figures. The technical details of its calibration has been explained via both good and bad calibration situations. The assembly of H3D camera on both RC car and real car has also been displayed, they will be applied for data collection for the following experiments in Chapter 5. Having detailed the proposed sensor technology, the types of image processing approaches need to be understood, and various algorithms for H3D database should be used to achieve good results.

# Chapter 4

# H3D Database Benchmark for Face Recognition

## 4.1 Introduction

For purpose of evaluating the performance of H3D imaging system on machine learning, it is necessary to build a benchmark of H3D database. The H3D database benchmark is aiming to compare the performance of H3D imaging database with traditional database, as well as to compare various machine learning algorithms applying on H3D imaging database. This H3D database for benchmark is supposed to be a small database which is relatively easy to acquire, and the process of acquisition and database processing should be relatively controllable. In this thesis the process measured by this benchmark will be applied on autonomous vehicle perception platform, e.g. scene recognition, thus this H3D database for benchmark should have common points with H3D scene database.

Therefore, in this chapter, H3D face database for face recognition is chosen to be the benchmark. On the one hand, face recognition can be analogised with scene recognition. One person can be compared to one specific scene, different face or expression images of this labelled person can be compared to different scene images taken at different positions or times of the same labelled scene. On the other hand, the acquisition of face database is easier than dynamic database, as the H3D camera stays still and volunteers could follow instructions. Moreover volunteers still stay dynamic, the images taken could be different even under same expression of same person, it can be analogised with images taken at different times of same scene, it is more suitable than taking images of inanimate objects.

Face recognition has made significant progress in recent years. It has been successfully applied to many applications from security entrance systems for staff identification to visual surveillance. However, the performance of current systems are highly affected by factors such as subject's head orientation, facial expressions, and face wear-on. In order to address this challenge, researchers usually resort to more training data or more complex recognition models.

In this chapter, H3D imaging system is used for face recognition based on a relatively small database. To show the effectiveness of the sensor, it is used with a simple Back Propagation Neural Network (BPNN) classifier. The accuracy of the system is illustrated through an experiment and shows that it has high accuracy and is robust against challenges aforementioned.

The proposed approach consists of three steps: (a) H3D face images are captured using the single aperture H3D camera. (b) Wavelet features are extracted and selected from multi-perspective viewpoint images obtained from the H3D images. (c) A simple Artificial Neural Network (ANN) is used for reliable face recognition.

In addition, with the aim of comparing the performance of H3D face database with 2D and stereo 3D database, the database is recombined to various comparison groups to train in different network. Meanwhile, in order to compare different face recognition algorithms, the deep convolutional neural network of AlexNet and classifiers such as SVM and KNN are used for comparison.

## 4.2 Face Recognition Methods

Vision based human face recognition is a key challenge and an active research topic within visual perception. The reason is that the perceived human face, in an uncontrolled application, will have to deal with subtle variations in face features. These can be in the form of head orientation changes that affect the position and visibility of face features in the frame, face wear-on like glasses, face emotions such as sad, angry and surprised expressions that affect the geometry of facial features. Although face recognition research began in the late 1960s and has had considerable achievements, demand for reliable and dynamic systems that can recognise faces even in the presence of such noise has grown in parallel. As a result, many researchers such as those in [152, 153, 154, 155, 156, 157] have been actively working to find robust and reliable face recognition models.

Face recognition approaches can generally be categorised in to five groups:

(1) Geometric feature based methods [158, 159, 160]: These methods use characteristics and structural geometry of face features including partial shape features such as eyes, nose, mouth, face shape [161] to recognise a person.

When extracting features, it tends to use some prior knowledge of the structure of facial patterns. Earlier work [159] in the field used extracted features of contour line from the side. Later works [160] have used features of front human face more. This because there are more information from front face than the contours, and relatively stronger anti-interference ability of these features. As a result, the geometric feature vector is based on shape and geometry of a human face organs. The components generally include the Euclidean distance [160], the curvature and angle between two specific points.

Geometric feature vector must have a certain uniqueness that can reflect differences between different faces, but must have a certain degree of flexibility to eliminate the influence of the time span, light and other factors. Integral projection curves on the image edge points are usually used to fix position of the eyes, nose and mouth and so on.

(2) Template matching methods [162, 163, 164, 165]: As the name suggests, this method uses face templates for recognition. The template is created using following steps:

- Face area is manually cut as sub face samples and the scale and grey-scale distribution of sub face samples are standardised.
- Average of grey scale of all samples are taken to compute an average face image, which is reduced to the same scale as the original template.
- Original template's eyes part is taken and its grey-scale distribution is standardised and saved as the eyes template.
- Original template is extended to different width and length ratios to fit different shapes of human faces. Again, their grey-scale distributions are standardised and saved as face templates.

Face recognition is performed by acquiring a new same size image of the face and checking it against the templates, searching for matching point in the search area.

(3) The Artificial Neural Network (ANN) method [166, 167]: This method typically takes grey-scale image pixels as neuron inputs and produces a classification result. ANNs consist of two steps:

- Training the system with many pictures of each subject to be classified, to pick the tiny details that differentiate one person from another.
- Performing classification using new images of the subjects.

(4) Elastic graph matching method [168, 169, 170, 171]: With this method, first, a rectangular grid of nodes is placed on an image of the face. Then, each node is described with the node multi-scale Gabor magnitude, and the connected relation between the nodes are represented by geometric distance, so as to constitute face description based on two-dimensional topology. The recognition is performed using the similarity of each node and the connection between two images (new face image with training image).

(5) Hidden Markov Model (HMM) method [172, 173, 174, 175]: This method typically contains several one- dimensional continuous HMM states representing human face features such as hair, forehead, eyes, nose and mouth. In this method, the face image is divided into blocks representing the face features, with some over lap. Each block is transformed with the KL transform [176]. Then, several transform coefficients are selected as an observation vector to train HMM.

As it is mentioned in the previously, these approaches can get either very complex or computationally expensive trying to deal with challenges aforementioned. In this chapter, the approach is proposed that focuses not the recognition method, but on the sensor type used. It is believed that the key to a successful and reliable face recognition system is not creating more complex recognition models or exhaustive training, but a more rich data source.

## 4.3 Convolutional Neural Network of AlexNet

AlexNet is a convolutional neural network designed by Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton [136], that is trained on more than a million images from the ImageNet database [143]. It won the 2012 ImageNet LSVRC-2012 competition by a large margin (15.3% VS 26.2% (second place) error rates) [177]. AlexNet had a very similar architecture as Yann LeCun's LeNet in 1998 [178], but much deeper. The model is a large, deep convolutional neural network trained on raw RGB pixel values. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of the convolutional layers are followed by max-pooling layers, and three globally-connected layers with a final 1000-way softmax. It took five to six days to train on two NVIDIA GTX 580 3GB GPUs. The non-saturating neurons and a very efficient

GPU implementation of convolutional nets were used to make training faster [136]. The top highlight of this model is that using overlapped pooling to reduce the size of network. It reduces the top-1 and top-5 error rates by 0.4% and 0.3% repectively. Dropout was used instead of regularisation to deal with overfitting, however it doubled the training time with the dropout rate of 0.5. Also ReLU was used instead of Tanh to add non-linearity, which accelerates the speed by 6 times at the same accuracy [179].

The architecture of AlexNet is shown in Figure 4.1 [180]. The AlexNet contains eight layers with weights: 5 convolutional layers and 3 fully connected layers. The output of the last fully connected layer is fed to a 1000-way softmax, which will produce a distribution over 1000 class labels. The network maximises the polynomial logistic regression objective, which is equivalent to maximising the average between training cases of the correct label log probability in the predicted distribution. The kernels of the second, fourth, and fifth convolutional layers are connected only to those kernel maps in the previous layer residing on the same GPU. The kernels of the third convolutional layer are connected to all kernel maps in the second layer. In the fully-connected layers the neurons are connected to all neurons in the previous layer [136]. The network has 62.3 million parameters and needs 1.1 billion computation units in a forward pass. We can also see convolution layers, which accounts for 6% of all the parameters, consumes 95% of the computation [181].

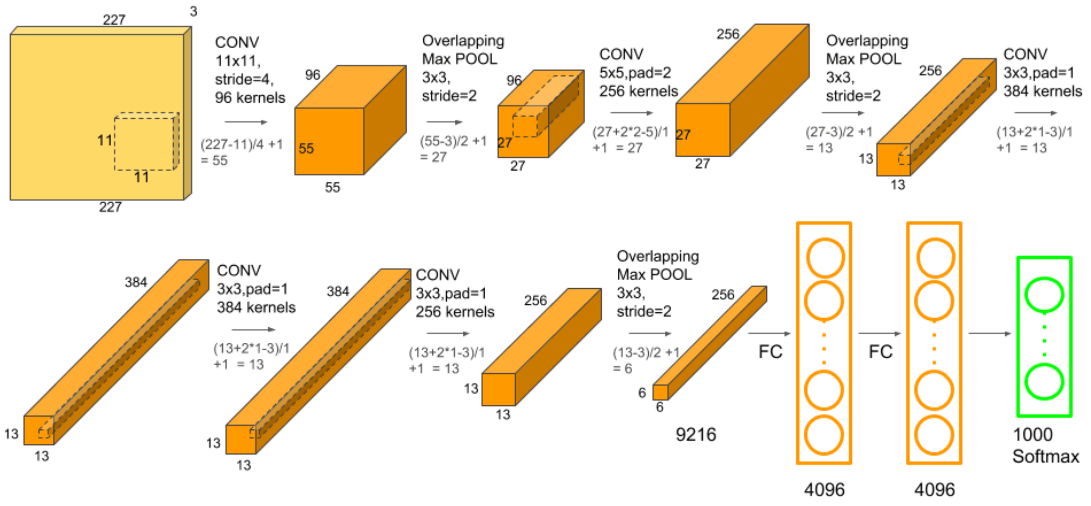


Figure 4.1: Network architecture of AlexNet
[180]

AlexNet takes roughly 90 cycles through the training set of 1.2 million images which were trained for five to six days simultaneously on two Nvidia Geforce GTX 580 GPUs. Stochastic gradient descent is using with a batch size of 128 examples, the learning rate is 0.01, momentum of 0.9, and weight decay of 0.0005 is used. The learning rate is divided by 10 once the validation error rate stopped improving with the current learning rate. The learning rate is decreased 3 times during the training process [136] [181]. The update rule for weight $w$ was [136]

$$\begin{aligned} v_{i+1} &:= 0.9 \cdot v_i - 0.0005 \cdot \varepsilon \cdot w_i - \varepsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i} \\ w_{i+1} &:= w_i + v_{i+1} \end{aligned} \tag{4.1}$$

where $i$ is the iteration index, $v$ is the momentum variable and $\varepsilon$ is the learning rate. $\left\langle \frac{\partial L}{\partial w}|_{w_i} \right\rangle$ is the average over the $i_{th}$ batch $D_i$ of the derivative of the objective with respect to $w$, evaluated at $w_i$ [136].

Figure 4.2 shows 8 ILSVRC-2010 test images and the five labels considered most probable by AlexNet [136]. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5).

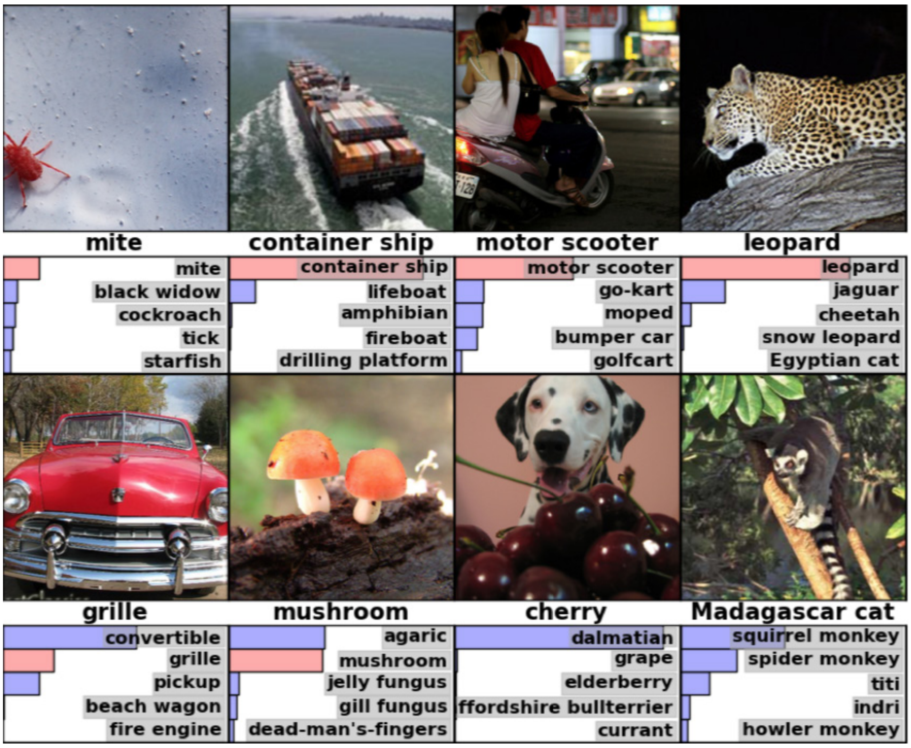


Figure 4.2: AlexNet test images and their top-5 predictions
[136]

In the 2010 ImageNet competition, the best model achieved 47.1% top-1 error and 28.2% top-5 error. AlexNet greatly exceeded this with a 37.5% top-1 error and a 17.0% top-5 error. AlexNet is able to recognise off-centre objects and most of its top five classes for each image are reasonable. It won the 2012 ImageNet competition with a top-5 error rate of 15.3%, compared to the second place top-5 error rate of 26.2% [182].

The results show that a large, deep convolutional neural network can use pure supervised learning to achieve record-breaking results on a very challenging data set. In the second year of AlexNet publication, all entries in the ImageNet competition used convolutional neural networks for classification tasks. AlexNet is the pioneer of CNN and has opened up a whole new era of research. After releasing so many deep learning libraries, AlexNet is very easy to implement [181].

## 4.4 H3D Face Database

There are a number of face image databases published such as the FERET database [183], the Yale face database [184] and the MIT-CBCL face recognition database [185]. However, these face databases are all for 2D face images. Because a novel H3D sensor is used in this thesis, the face database for H3D imaging system is needed.

### 4.4.1 Experiment Setup

Since no such database exists, a H3D face image database is created in this thesis. The face database is created by capturing faces of 20 volunteers of mixed races and genders. Each volunteer's instructed to sit in front of the H3D camera facing the camera lens. The guidance of 7 expressions from multiple existing face dataset are shown to the volunteers. 7 different facial expression: neutral, joy, anger, sadness, fear, surprise and disgust is captured for each volunteer, with the aim of enhancing the robustness of this face recognition approach.

The camera used is illustrated in Figure 4.3. It's layout consists of a prime lens, micro-lens array, relay lens and digital camera sensors with Canon 5D. To adjust the camera settings and aperture, the size of $6.7mm$ of the side length of the square hole is determined to be the suitable square aperture when the aperture is 1.8.

The images are captured in TC005e at Brunel University. The camera system was setup such that the distance between the volunteer and camera sensor was $1.8m$, and the vertical height from the camera sensor to the ground is $1.2m$ and the white background is a whiteboard. The shooting layout is shown in Figure 4.4.

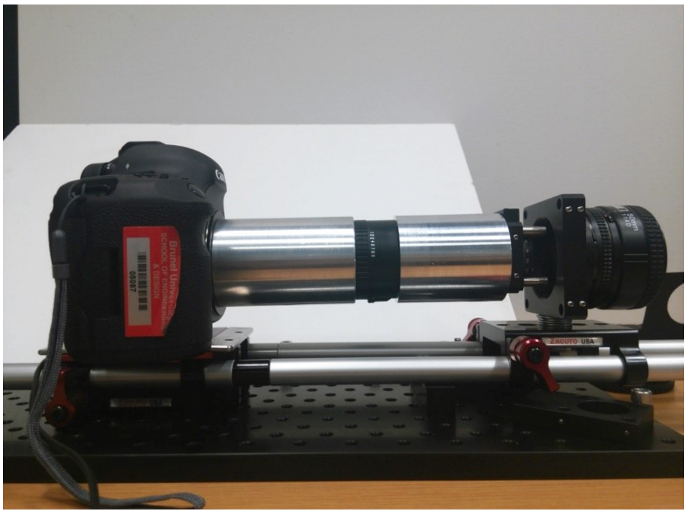


Figure 4.3: The H3D camera used for H3D face database capture

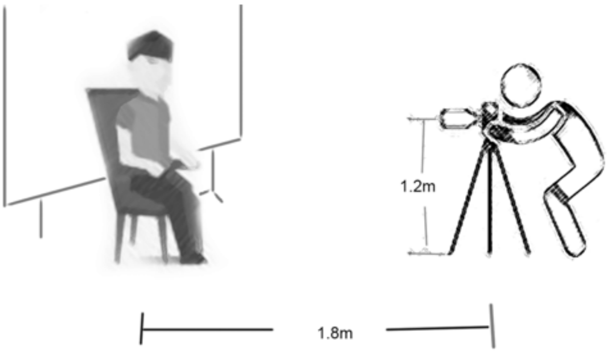


Figure 4.4: Shooting layout of the H3D face database

### 4.4.2 H3D Database Acquisition and Preparation

As mentioned above, there are 20 volunteers involved and 7 expressions are captured for each. This makes total of H3D images captured 140. Due to 2 of the 20 people wear glasses, images of both wearing and not wearing glasses are captured. Therefore, 154 raw H3D images are captured.

Figure 4.5 shows the H3D face database used for the face recognition training and testing. Neutral Face images are used for training, and face images with expression are used for testing. Figure 4.6 on the other hand shows few test cases without glasses which are used for advanced level of face recognition only.

After capturing image pre-processing is performed in Photoshop to prevent barrel distortion and scaling error distortion. Details of these procedures can be found in [186]. Then view point images are extracted from original H3D images using MATLAB. Figure 4.7 illustrates a H3D image as well as the process for extracting multi perspective viewpoint images from this H3D image. For viewpoint image extraction in Figure 4.7 (b), the number 82

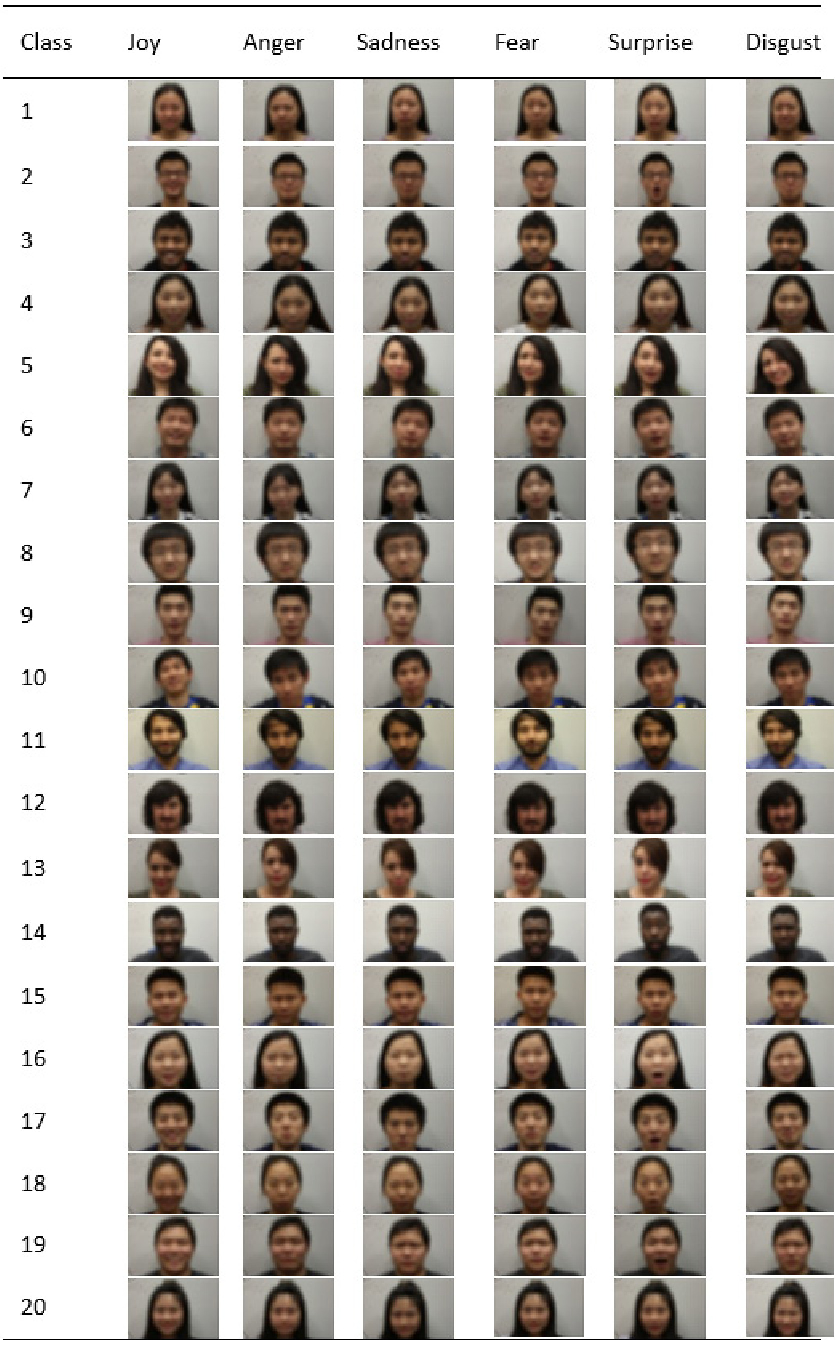


Figure 4.5: Dataset 1: Database of face images with different expressions

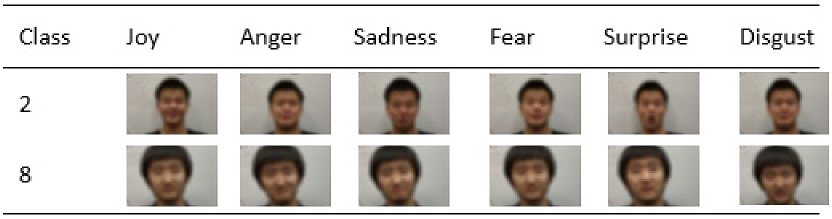
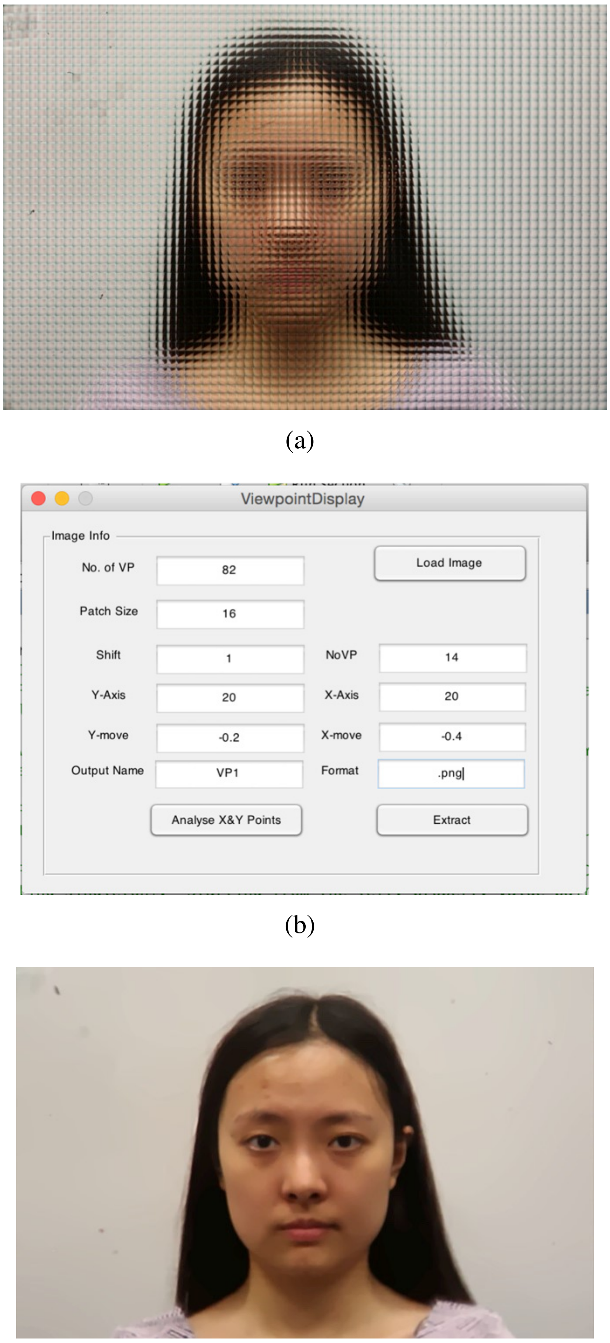


Figure 4.6: Dataset 2: Database of face images without glasses

of viewpoints means there are 82 pixels in both *x* and *y* axis in each elemental images. Patch size determines the size of output viewpoint image. X-move and Y-move makes fine tuning of extraction location, and the chosen viewpoint coordinates are filled into X-Axis and Y-Axis.


(a)

ViewpointDisplay

Image Info

No. of VP: 82

Patch Size: 16

Shift: 1

NoVP: 14

Y-Axis: 20

X-Axis: 20

Y-move: -0.2

X-move: -0.4

Output Name: VP1

Format: .png

(b)

(c)

Figure 4.7: Process of extracting viewpoint image from H3D image (a)Holoscopic 3D Image (b)Viewpoint image reconstruction software and parameters (c)A viewpoint image rendered from the H3D image

Figure 4.8 illustrated the structure of this face dataset.154 raw H3D images are captured, and each image is extracted into 25 different viewpoint images. The viewpoint images are extracted from 10 to 30 in both $x$ and $y$ axis directions, and the difference between every two points is 5. Therefore, the values of $x$ and $y$ are 10, 15, 20, 25 and 30, in this way the coordinate of viewpoints can be expressed as: $10 \times 10$, $10 \times 15$, $10 \times 20$, ......, $30 \times 25$, $30 \times 30$; 25 viewpoint images with corresponding viewpoint coordinates in total. Altogether there are 3850 viewpoint images with $1075 \times 691$ resolution in this dataset for both training and testing.

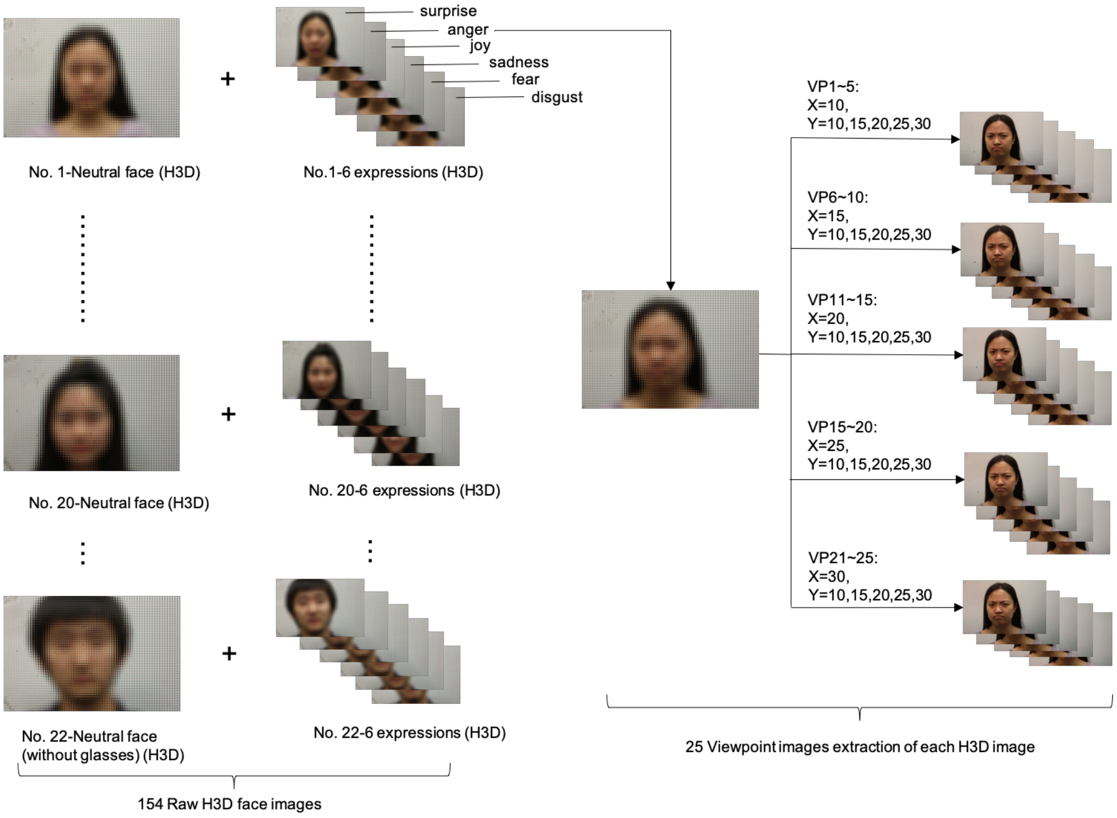


Figure 4.8: Face dataset structure

## 4.5 H3D Face Recognition using BPNN

With the H3D database built already, the features are expected to be extracted and fed into neural network for training and testing. The neural network applied here is simple BPNN, and the analysis of performance and accuracy is shown at final step.

### 4.5.1 Feature Extraction

For feature extraction, the wavelet transform method is used. Wavelet transform method uses a gradually fine temporal or spatial sampling step size of high frequency components.

As a result, it can both focus on details of the object with a strong selectivity of spatial position and orientation, and also capture partial structure of spatial and frequency information [187]. This makes wavelet transform is very robust. As a result, it can perform well even in the presence of light variation and changes in the face orientation. This have been evidenced in literature [188, 189], where it is found that the low-frequency sub-band image obtained by a multi-layer wavelet decomposition of face image is a robust feature for face recognition.

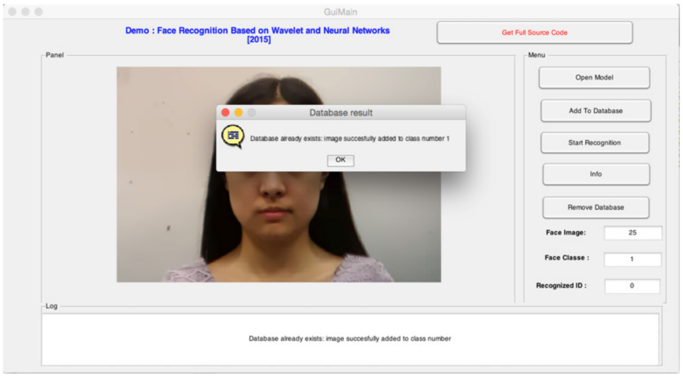
The other reason wavelet transform is used for feature extraction is because it reduces image information redundancy. Only a subset of the transform coefficients are necessary to preserve the most important facial features, such as hair outline, eyes and mouth. Wavelet transform retains high-frequency edge information of horizontal, vertical and diagonal direction, which mainly describe the characters of human face expressions. It is demonstrated experimentally that when wavelet coefficients are fed into a BP neural network for classification, a high recognition rate can be achieved by using a very small proportion of transform coefficients. This makes wavelet-based face recognition much more accurate than other approaches. In this chapter, the H3D face images are wavelet decomposed at level 3 using the wavelet Coiflet.

### 4.5.2 Training and Testing

For classification, the accuracy threshold is set to be 80% because of limited database size. If the accuracy rate is above 80%, this system is determined as an effective system. The BPNN is trained and tested using the viewpoint images constructed from the H3D images of volunteers in the database. Each micro-lens in the camera system captures 25 pixels, which means there are 25 viewpoint images for each of the H3D images.

For training, since only the neutral images of volunteers are using, there are $20 \times 25 = 500$ viewpoint image samples in the training database. Each image is $1075 \times 691$ in resolution.

For testing, the position of view point of six expressions (joy, anger, sadness, fear, surprise, disgust) were generated randomly in MATLAB for each volunteer. From Dataset 1 shown in Figure 4.5, we test with $20 \times 6 = 120$ images– a random viewpoint image of every expression for every volunteer. Also from Dataset 2 shown in Figure 4.6, we test with $2 \times 6 = 12$ images - a random viewpoint image of every expression for every volunteer. Figure 4.9 displays the process of training and testing in MATLAB.



Demo : Face Recognition Based on Wavelet and Neural Networks [2015]

Database result

Database already exists: image succesfully added to class number 1

Face Image: 25

Face Classe : 1

Recognized ID : 0

Database already exists: image succesfully added to class number

Figure 4.9: Process of training and testing in MATLAB

### 4.5.3 Results and Performance Evaluation

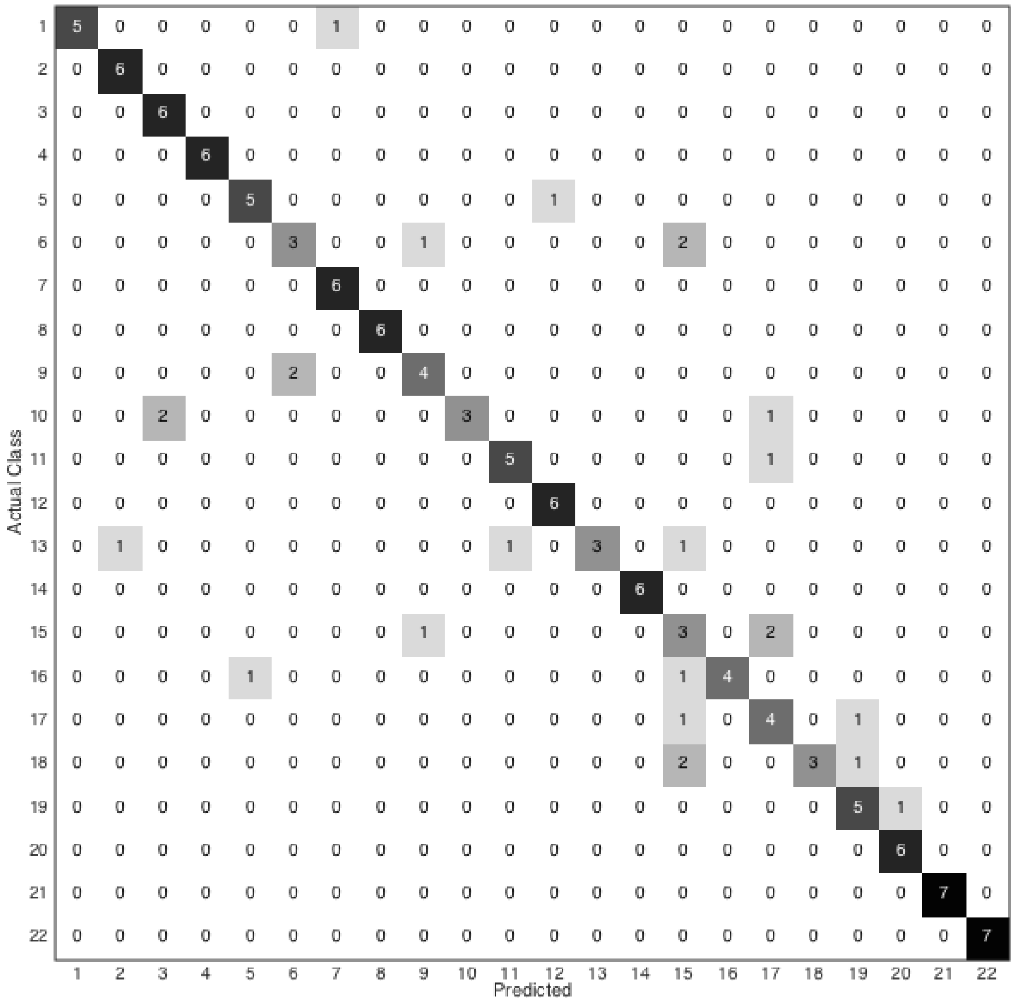
To analyse the performance of our model, we will use the confusion matrix also called known as an error matrix. A confusion matrix is a table that is used to describe the performance of a classification model (or ”classifier”) on a set of test data for which the true values are known [190]. Considering a simple Yes or No result we can get four types of results:

- True Positives (TP): The prediction is yes, but the actual value should be yes.
- True Negatives (TN): The prediction is no, but the actual value should be no.
- False Positives (FP): The prediction is yes, but the actual value should be no.
- False Negatives (FN): The prediction is no, but the actual value should be yes.

Given this information, we can calculate the accuracy of our model using:

$$x = \frac{TP+TN}{TP+TN+FP+FN} \tag{4.2}$$

The confusion matrix in Figure 4.10 illustrates the performance of the proposed approach. The rows represent the actual classes and the columns represent the predicted classes. It also illustrates the prediction accuracy by using different grey-scale colour intensities. The deeper the colour, and the higher the prediction accuracy. As can be seen, in most cases the system agrees with the actual desired result. In some classes such as class 6, 10 and 15, the system tends to perform a higher error rate. When check back the database, the reasons of not so well head pose, and in some viewpoint images the volunteer’s face does not display properly, are likely to result in the false positive rate.



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 3 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 2 | 0 | 0 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 4 | 0 | 1 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 3 | 1 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 1 | 0 | 0 |
| 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 |

Figure 4.10: Confusion Matrix of the proposed algorithms

Using the Equation 4.2, the accuracy of the proposed face recognition system using the datasets 1 and 2 is $(96+2\times7)\div(120+2\times7) = 0.820896$. This is equivalent to 82.1%, which is greater than the threshold of 80%. This shows that our proposed approach for face recognition using the novel H3D sensor is effective in recognition of faces under different face expression or face with glasses.

## 4.6 H3D Face Recognition Comparison

In order to compare the performance of H3D face dataset on different approaches, as well as to compare the performance between 2D, stereo 3D and H3D dataset among each approach, in this section, more experiments are proceeded. These are based on comparison groups of face dataset, and different machine learning network or classifiers.

### 4.6.1 Evaluation Plan

The viewpoint images are composed to different combinations for comparison. As explained in the database image processing part previously, the viewpoint images are extracted from 10 to 30 in both $x$ and $y$ axis directions, and the difference between every two points is 5. Therefore, 25 viewpoint images with corresponding viewpoint coordinates in total.

There are three categories of viewpoint images combination: (1) 2D / stereo 3D. In this category two groups of viewpoint images are included, one is $10 \times 10$ and $10 \times 30$ which mimic a view of stereo 3D from two positions horizontally; the other is $10 \times 10$ and $30 \times 30$ which can be considered as 2 images of 2D because coordinates are different both vertically and horizontally. (2) Unidirectional. This category contains 4 groups of viewpoint images, $x = 10, y = 10, x = 10, 20, 30, y = 10, 20, 30$. Namely, they are the first line and the first column, the first, third and last line and column. (3) Omnidirectional. This category includes all 25 viewpoint images; in other words it is the H3D group. Figure 4.11 illustrates the combinations of the three categories.

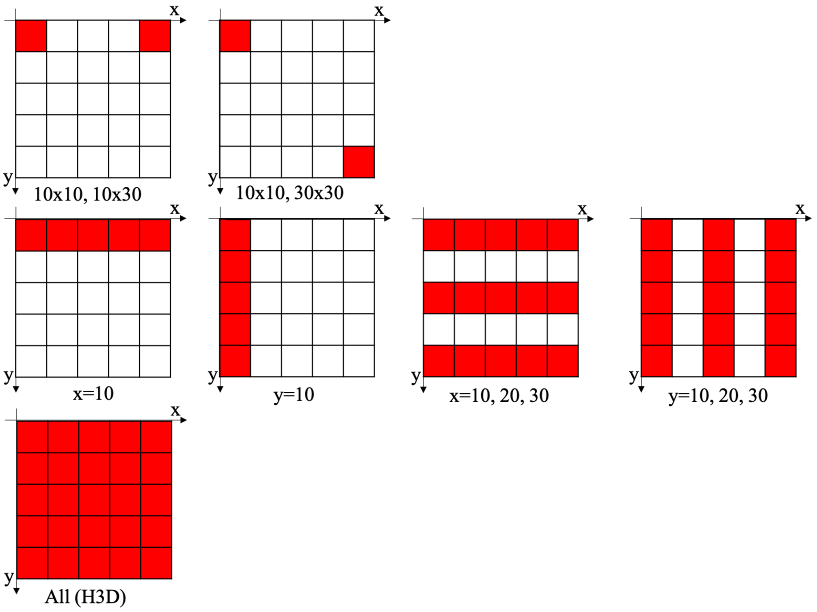
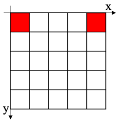
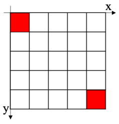
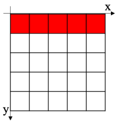
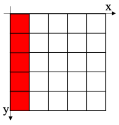
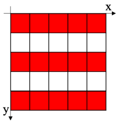
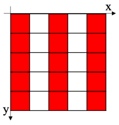
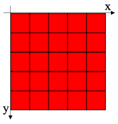


Figure 4.11: Diagram of combinations of viewpoint images

The three categories of combinations are training in 3 classifiers: AlexNet, SVM and subspace KNN. The features used in SVM and subspace KNN are extracted using wavelet transform, which is same as the previous approach used in BPNN. In SVM gaussian kernel is used. The performance of different combinations in different classifiers can be evaluated to verify the robustness of use of H3D to face recognition, and the optimal classifiers of different approaches can be figured out simultaneously.

### 4.6.2 Performance Evaluation

Table 4.1: Performances of different viewpoint image combinations in different approaches

| Combinations | Diagrams | AlexNet | Features+SVM | Features+subspace KNN |
|---|---|---|---|---|
| $10 \times 10\&10 \times 30$ |  | 0.2687 | 0.3000 | 0.6833 |
| $10 \times 10\&30 \times 30$ |  | 0.3507 | 0.4833 | 0.7250 |
| $x = 10$ |  | 0.3060 | 0.3750 | 0.6833 |
| $y = 10$ |  | 0.3955 | 0.5083 | 0.5083 |
| $x = 10, 20, 30$ |  | 0.4104 | 0.5917 | 0.7667 |
| $y = 10, 20, 30$ |  | 0.4179 | 0.5833 | 0.7667 |
| $All(H3D)$ |  | 0.8657 | 0.5833 | 0.7750 |

The Table 4.1 shows the results of different viewpoint image combinations in different approaches.

Basically, the correct rates are increasing gradually by comparing vertically. In general, the accuracies of unidirectional groups perform better than 2D/stereo 3D groups except the subspace KNN approach, and in unidirectional groups, the combination of either three lines of three columns perform higher accuracy than only one line or column. The combinations of all viewpoint images, also named as H3D group, achieve the best performance among all combinations, even though in SVM approach it displays very little disadvantage than one of the unidirectional combinations. This verifies the robustness of H3D face database, with its comprehensive viewpoints of all diversities.

In 2D/stereo 3D groups, the lowest accuracies almost among all groups are due to small size of samples and lack of diversities. The $10 \times 10\&10 \times 30$ combination attributes obviously better accuracy than stereo 3D ($10 \times 10\&10 \times 30$) combination to more diversity it contains. In unidirectional groups, the correct rates of vertical combination are higher than horizontal combination in AlexNet approach and SVM approach, however it appears reverse result in subspace KNN approach. When it comes to three lines or columns, the correct rates are relatively similar either in horizontal direction or vertical direction. This can be summarised as the two combinations have certain size of samples and contain considerably diversities.

The performances of the three approaches are comparatively different by comparing the results in the table horizontally. The AlexNet approach shows its superior performance in notable improvement of correct rate by expansion of samples, and with the highest accuracy of 86.57% in H3D group as well. This accuracy is relatively higher than the accuracy of using simple BPNN, due to the superiority of deep learning neural network. In 2D / stereo 3D combination, AlexNet performs quite low accuracy than the other two approaches, and it has been improved to some extent in unidirectional combination. In H3D group, the dramatically increase of the correct rate proves that AlexNet is a reliable approaches for H3D face recognition. The performance of SVM approaches improves by expansion of samples in the same way, however this approaches is considered not suitable for this H3D face database due to its lowest accuracy in H3D group. The subspace KNN approaches achieves moderately high performance among every combination, even in small sample size group of 2D / stereo 3D.

## 4.7 Summary

To illustrate the use of H3D images with artificial neural networks, it starts by performing a relatively small experiment whereby one of the hard questions of classifying human faces is addressed in the presence of untrained emotion. For the model of H3D face database to perform well, the accuracy rate target of 80+% has been set. The accuracy has reached 82.1% with training in BPNN using features extracted by wavelet transform. The performances of 2D and stereo 3D face database have been compared with H3D face database in various algorithms, this holistic evaluation confirms the rich content of H3D performs higher accuracy than traditional 2D content in the adopted face recognition approaches. Among all the algorithms applied in this chapter, the deep convolutional neural network of AlexNet displays its superiority on relatively small database than other algorithms, as a result, it will be used in the following chapter for H3D perception platform on car.

# Chapter 5

# Holoscopic 3D Perception Engine for Autonomous Vehicles

## 5.1 Introduction

In this chapter, the experiments of H3D perception platform are conducted on both RC car and real car in order to face variety of surrounding circumstances. Due to the scale of RC car and the shooting height of the camera on it, using RC car is suited to environments that are not complicated, such as indoor shooting and sidewalks. When facing the real world traffic conditions, like multiple vehicle streams and pedestrians on main street, the real car is the primary that H3D perception platform will be applied on.

The main mission of H3D perception platform on car can be divided into two parts: scene classification and object detection. Section 5.2 focuses on the application of H3D perception platform on RC car. The experiment part of data acquisition is introduced by explaining the data capture plan, database setup and data pre-processing. Then the dataset is training using the pre-trained deep learning network - AlexNet - for scene classification. The network of AlexNet is confirmed high performance on H3D database via the conclusion drawn from previous chapter. The training progress is illustrated and the results as well as accuracy is analysed and evaluated in this section. Section 5.3 focuses on the application of H3D perception platform on real car. The data acquisition part is also detailed in experiment setup, database acquisition and preparation. With different algorithm is using, the data need to be annotated first, then training using YOLOv3 for object detection. The results and accuracy is analysed and evaluated at the end of this section. Finally the summary is given over the two applications of H3D perception platform in Section 5.4.

## 5.2 Holoscopic 3D Perception Platform on RC Car

Before proceeding the experiment on real car, the RC car is adopted for relatively simple environment shooting. It is easier to control the environment variables such as pedestrians and road conditions. Since the data is captured in the form of video, it is also convenient and fast to shoot every single video and make any adjustment in between.

### 5.2.1 Experiment Setup

The route of data shooting is selected from Brunel University campus. The route should be a sidewalk with variety of surrounding scenes in a quiet area. Figure 5.1 displays the selected path of route.

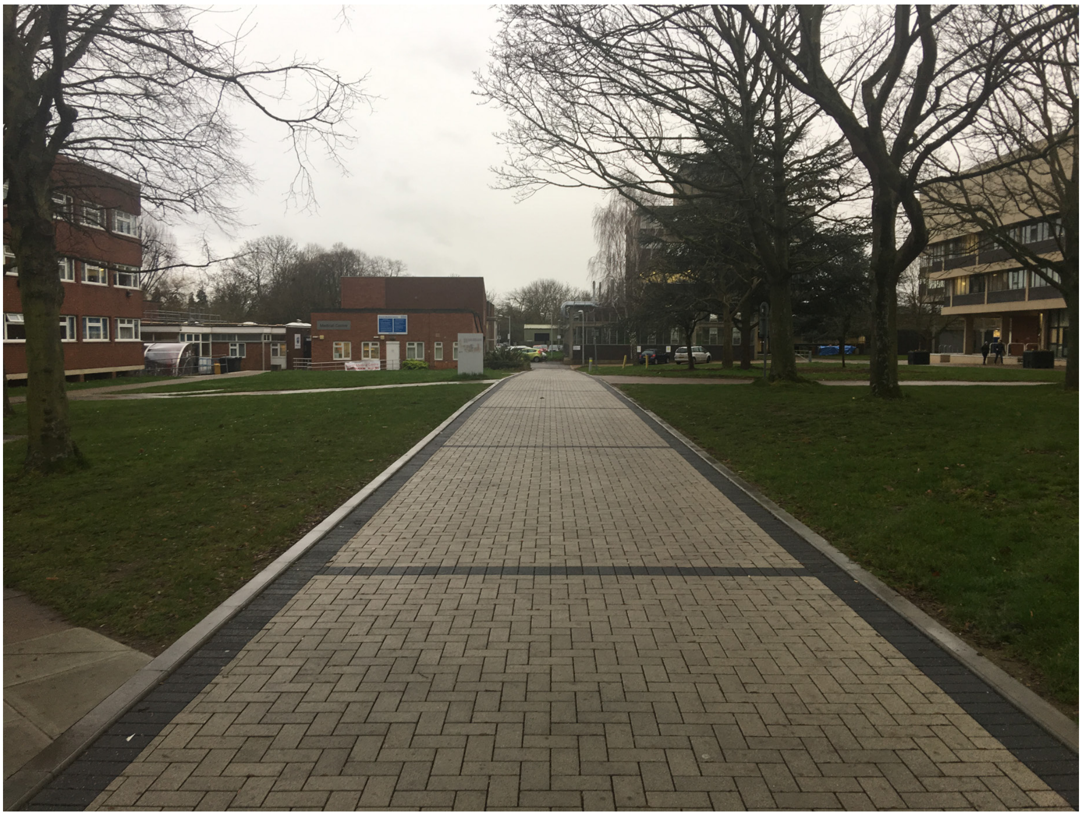


Figure 5.1: Selected data capture route in Brunel University campus

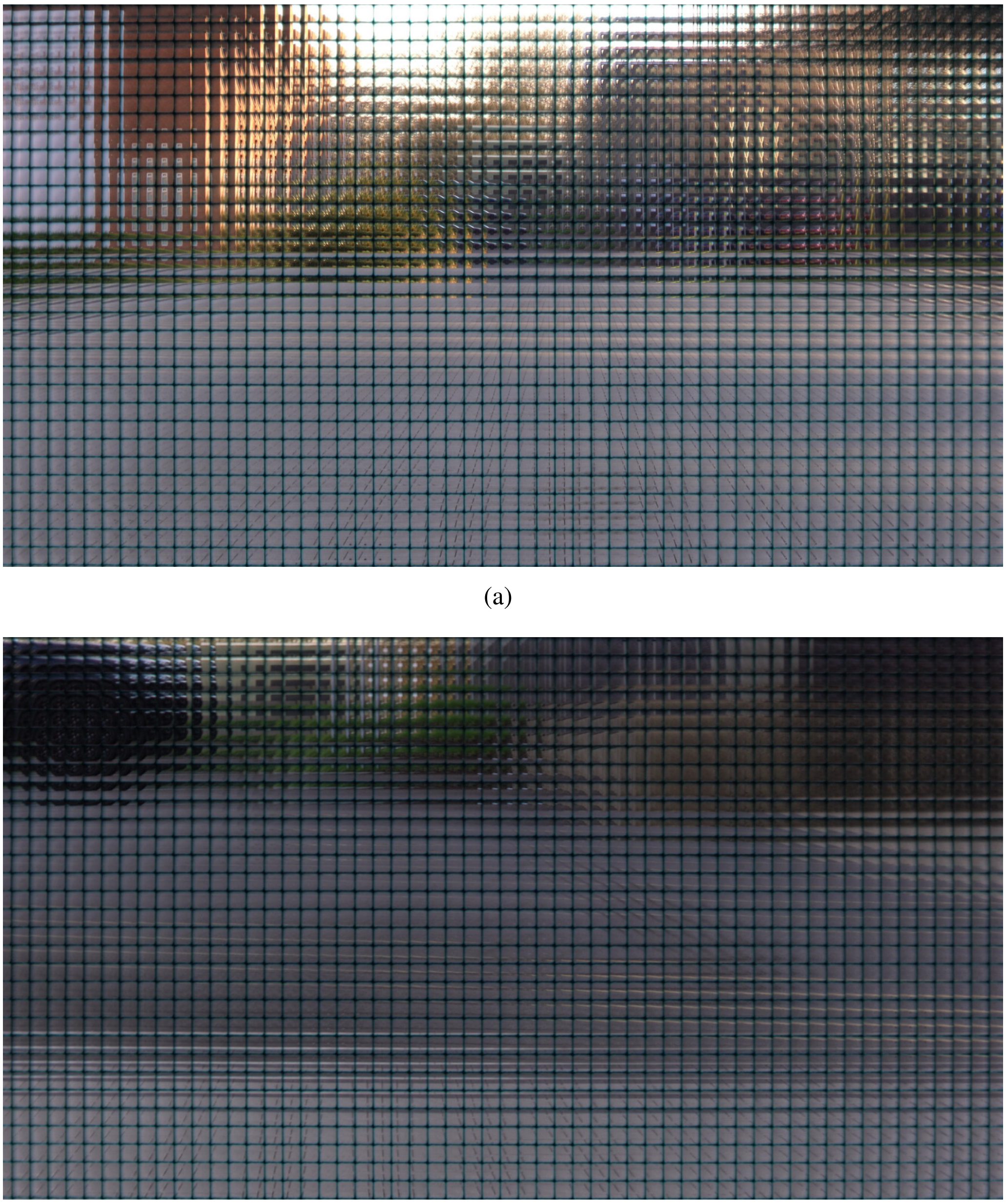
As shown in Figure 5.1, there are coloured sidewalk bricks form several lines on the ground. For scene classification, these brick lines become the markers of scene segmentation automatically. The route between every two adjacent brick lines is called one cell, or one segment. In the video shooting stage, the number of segments contained in each video is 10. When the first brick line appears on camera, the first segment starts to be counted until the second brick line disappears on camera. This means in every single video, there are 11 brick lines appear and then disappear.

Since the route has two opposite directions to choose for video shooting, one of the directions that contains the scene with varied and richer details. This direction is chosen to be captured in video because it provides more features for scene classification. The timings of video shooting should be chosen depending on the ambient light and quantity of pedestrians. As the chosen route is in a quiet area on campus, the better timings for shooting are afternoons with appropriate ambient light when very few or no pedestrians passing by. Too many pedestrians will lead to more uncertain features contained in videos. The H3D camera settings of ISO and shutter can be adjusted according to the light condition at the time of shooting. The resolution of the video is $3840 \times 2160$ and video quality is set to be the highest. The frame rate is 25 fps.

### 5.2.2 H3D Database Acquisition and Preparation

In data acquisition, the H3D camera starts to shoot when the RC car starts to run, and stops when the RC car reach the end of the 10th segment. The RC car is controlled by its joystick, and the speed can be fixed in a certain range to ensure the RC car not run too fast or slow. Since the speed cannot fixed to a specific value, the control of RC car is very careful to make it running in a really steady condition. A steady running condition contributes to an average time of every single video, as well as a similar time of every single segment.

There are 28 H3D videos have been taken and time duration of each video is between 2 minutes to 3 minutes. Each video contains 10 video segments, so in total there are 280 video segments with average of 12 seconds to 18 seconds each. The 10 video segments can be determined to 10 classes. Figure 5.2 displays two frames from the first segment and the last segment in the same raw video. As can be seen from Figure 5.2, the two segments contains totally different scenes.



(a)

(b)

Figure 5.2: Two frames from 1st and 10th segment in same video
(a) Frame from 1st segment (b) Frame from 10th segment

### 5.2.3 Data Pre-processing

The 28 raw H3D videos have been segmented manually with the purpose of classifying. According to the brick line marks, the principle of segmenting is when the brick line appears in the video, the current segment starts until the next brick line disappears iv video. Therefore, each raw H3D video has been segmented to 10 video segments, and there are 280 video segments altogether.

Since the frame rate is 25 fps and average time duration of each video segment is between 12 and 18 seconds, there are around 300 to 450 frames in every video segment. These frames are sampling in order to simplify the data processing, the sampling interval is 50 frames which means select one frame in every 2 seconds. This sampling action ensures the sampled frames to maintain proper diversity and reduce the repetition of content and scene between adjacent frames.

In order to extract H3D raw frames to viewpoint images, a traversal survey of all the raw H3D videos has been conducted to realise the suitable viewpoints range of every video. This ensures that frames of every video can be extracted into high quality viewpoint images when extracting from same viewpoints. The suitable viewpoints range is decided to be from 40 to 50 in $x$ axis and from 30 to 40 in $y$ axis. Finally there are 9 viewpoints chosen: $(40,30)$, $(40,35)$, $(40,40)$,$(45,30)$, $(45,35)$, $(45,40)$, $(50,30)$, $(50,35)$ and $(50,40)$. The resolution of each viewpoint image is $3840 \times 2160$. The structure of H3D RC car video database is illustrated in Figure 5.3 with an example of one video segment.

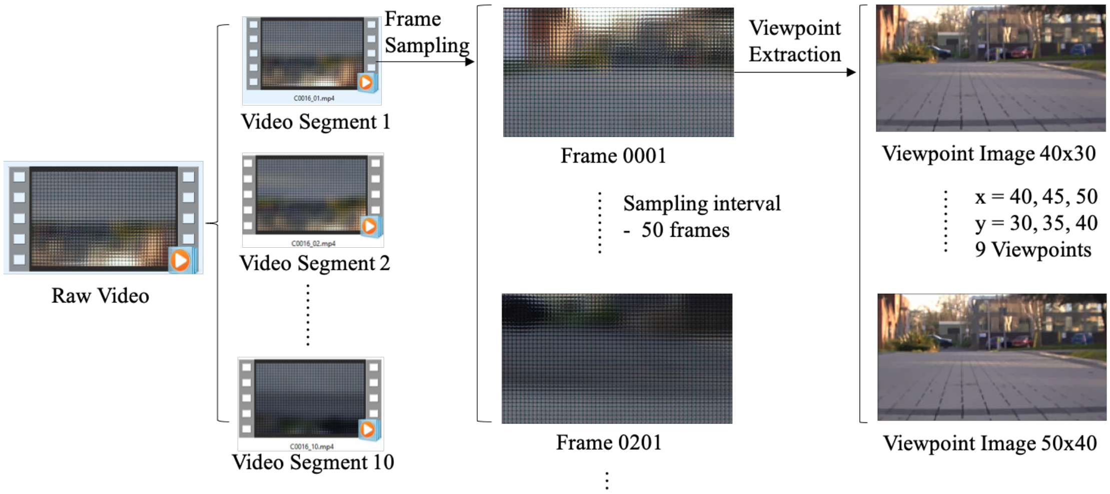


Figure 5.3: The structure of H3D RC car video database

The parameters used in viewpoint image extraction is shown in Figure 5.4. The number 69 of viewpoints means there are 69 pixels in both $x$ and $y$ axis in each elemental images. Patch size determines the size of output viewpoint image. X-move and Y-move makes fine tuning of extraction location, and the chosen viewpoint coordinates are filled into X-Axis and Y-Axis.

By initial experiment of training in proposed AlexNet using small dataset, the result tends to be not as good as expected. The reason could be the bottom half of every viewpoint image is the same - the ground os the sidewalk. Therefore, the viewpoint images are cutting to half to reduce the repeated features, and the resolution of half viewpoint image

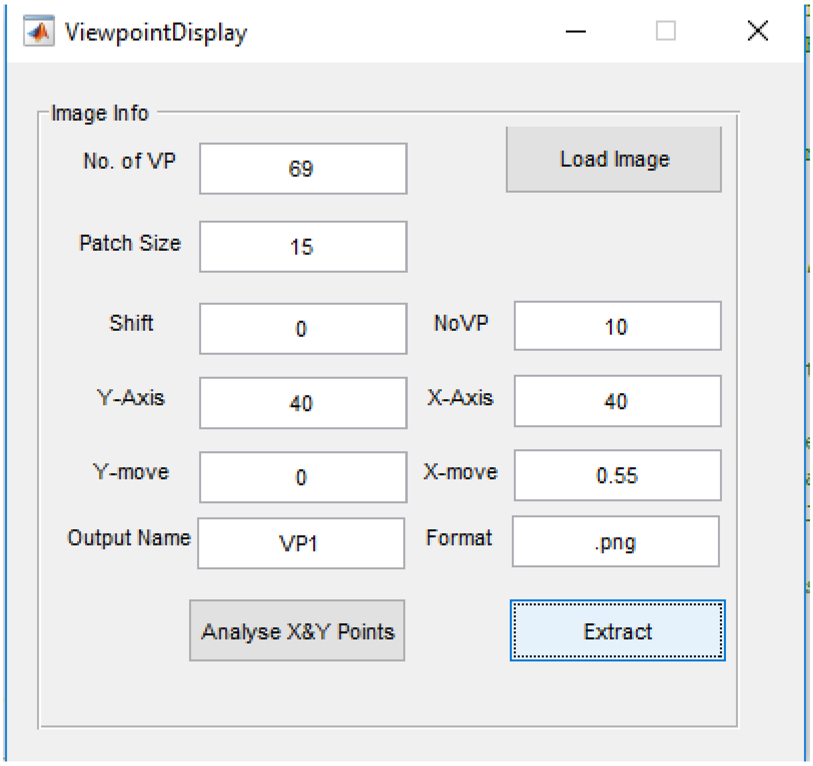


Image Info

No. of VP 69

Patch Size 15

Shift 0

NoVP 10

Y-Axis 40

X-Axis 40

Y-move 0

X-move 0.55

Output Name VP1

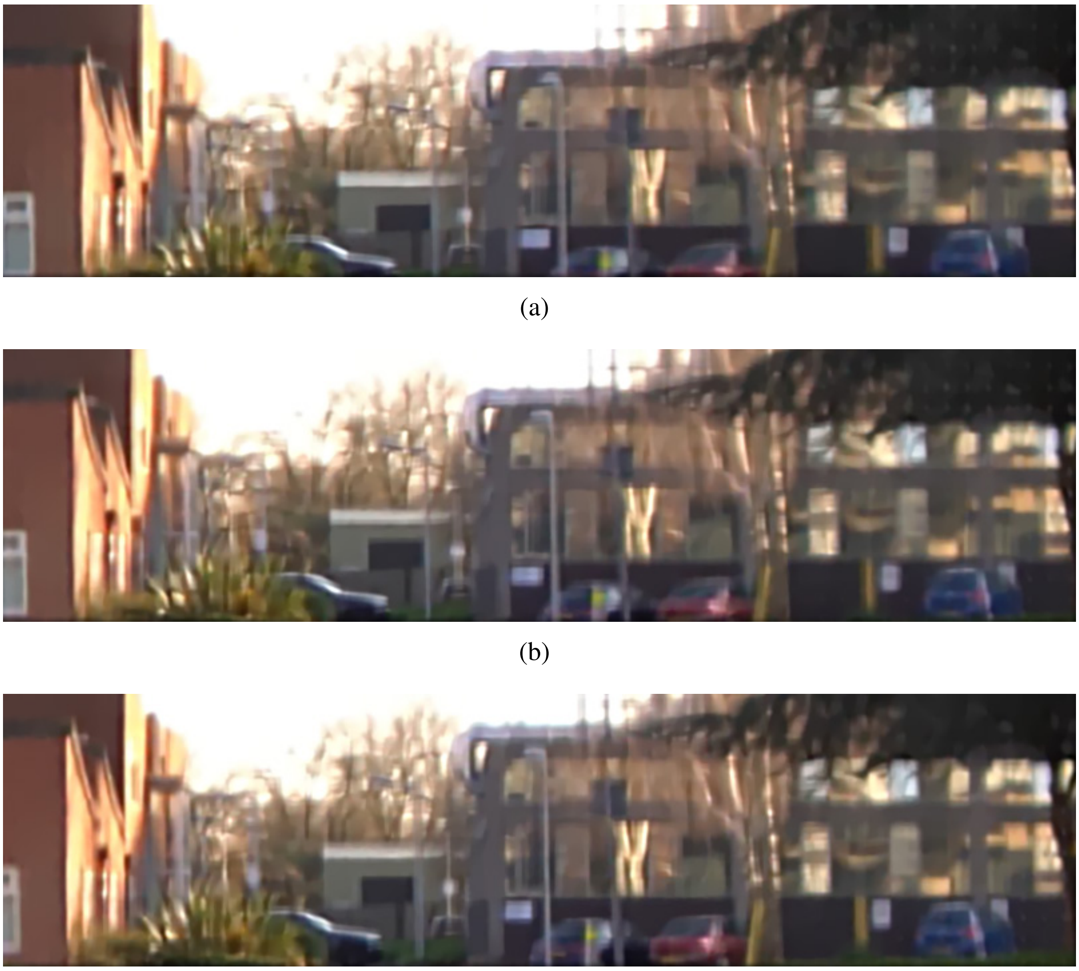
Format .png

Figure 5.4: Viewpoint image extraction parameters

is 3840 × 1080. The upper half of viewpoint images are remained as shown in Figure 5.5 and Figure 5.6.

Figure 5.5 displays different viewpoint images from the same frame of the same video segment. As can be seen, the three viewpoint images contains almost the same scene but from slightly different point of views, so there is slightly difference in occlusion.

Figure 5.6 displays the same scene / position from the same viewpoint of 30 × 40 in different videos. Because the videos are captured at different time so these three viewpoint images shows various ambient light. Moreover, although the shooting route is certain, there might be different running paths of the RC car in different shooting, so these three viewpoint images displays some varieties of content.

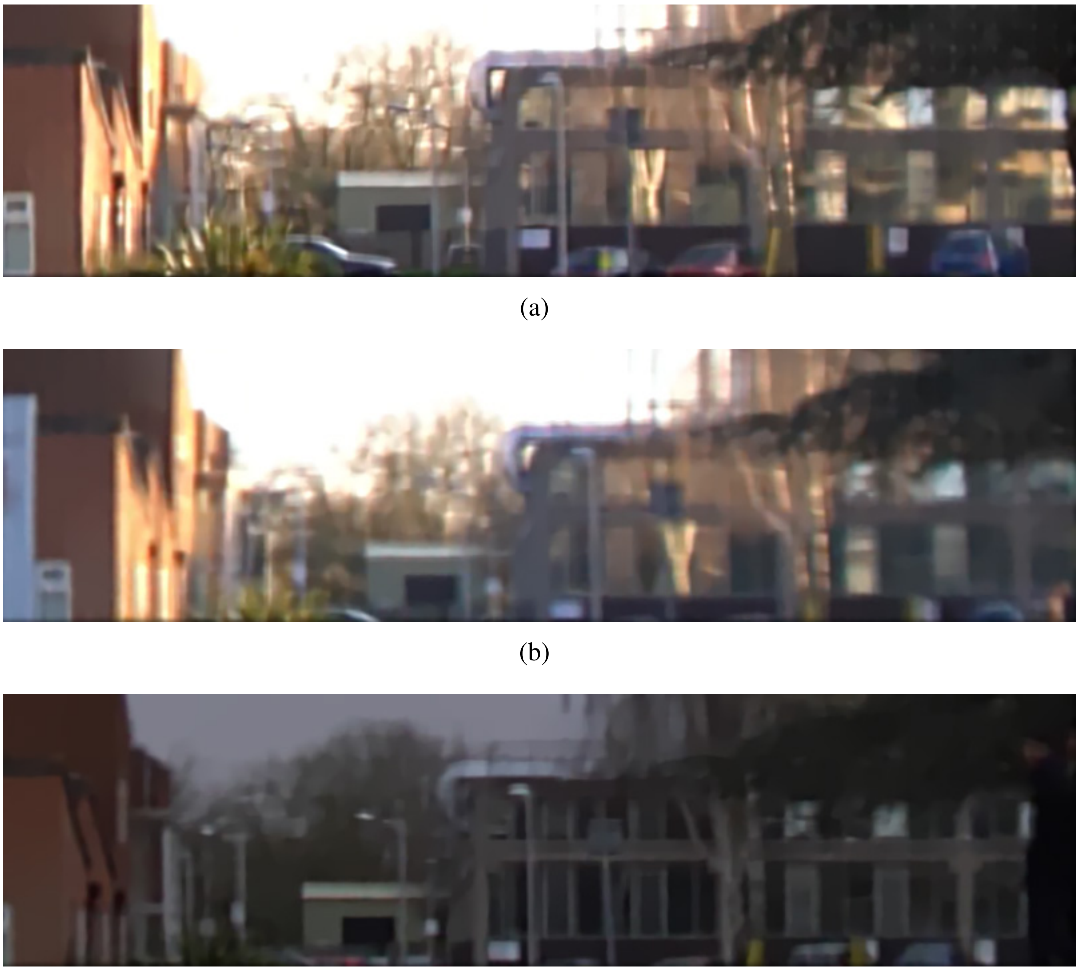


(a)

(b)

(c)

Figure 5.5: Different viewpoints from the same frame
(a)Viewpoint 30 × 40 (b)Viewpoint 35 × 45 (c)Viewpoint 40 × 50



(a)

(b)

(c)

Figure 5.6: The same scene / position from same viewpoint in different videos (a)Viewpoint 30 × 40 from video C0016 (b)Viewpoint 30 × 40 from video C0028 (c)Viewpoint 30 × 40 from video C0051

The viewpoint images are classified by different videos and video segments in current status, however for scene classification they need to be classified into labels by different scene classes. All viewpoint frames from video segment 01 of each H3D video are classified as Label 01, viewpoint frames from video segment 02 of each H3D video are classified as Label 02,...etc. Table 5.1 displays the number of frames in each label. There are 12276 sampled viewpoint frames altogether in 10 labels, and each label includes more than a thousand viewpoint frames, in average each label has about 1227 viewpoint frames.

Table 5.1: Number of viewpoint frames in each label

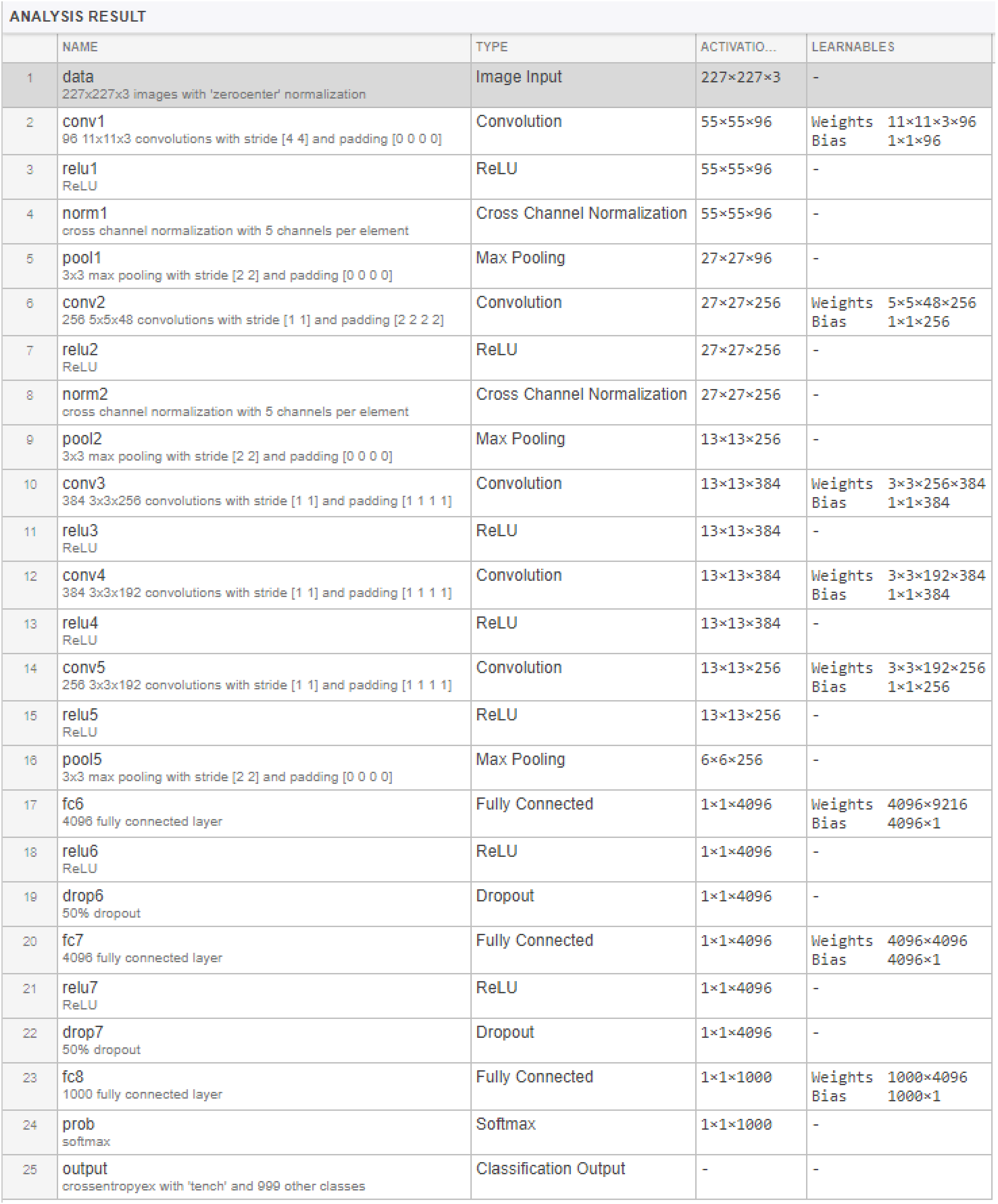
| Class Label | Number of Viewpoint Frames |
|---|---|
| 01 | 1431 |
| 02 | 1296 |
| 03 | 1269 |
| 04 | 1071 |
| 05 | 1152 |
| 06 | 1116 |
| 07 | 1125 |
| 08 | 1143 |
| 09 | 1134 |
| 10 | 1539 |

### 5.2.4 H3D Scene Classification using AlexNet

Because the deep convolutional neural network of AlexNet performs well on H3D face database classification, based on the conclusion drawn from Chapter 4, the pre-trained AlexNet implement of DeepLearning Toolbox in MATLAB is proposed to train H3D scene database for classification. The GPU using is NVIDIA Tesla K40c, which provides a high performance with its $12GB$ memory size, $745MHz$ clock speed and $288.4GB/s$ bandwidth.

The layers structure of pre-trained AlexNet implement on MATLAB is shown in Figure 5.7 along with its parameters.

There are 25 layers including the input and the output layer. It contains 5 convolutional layers and 3 fully connected layers. Relu is applied after every convolutional and fully connected layer. Dropout is applied before the second and the third fully connected layer. The input image size is $227 \times 227 \times 3$ but the images in the database have different sizes.



ANALYSIS RESULT

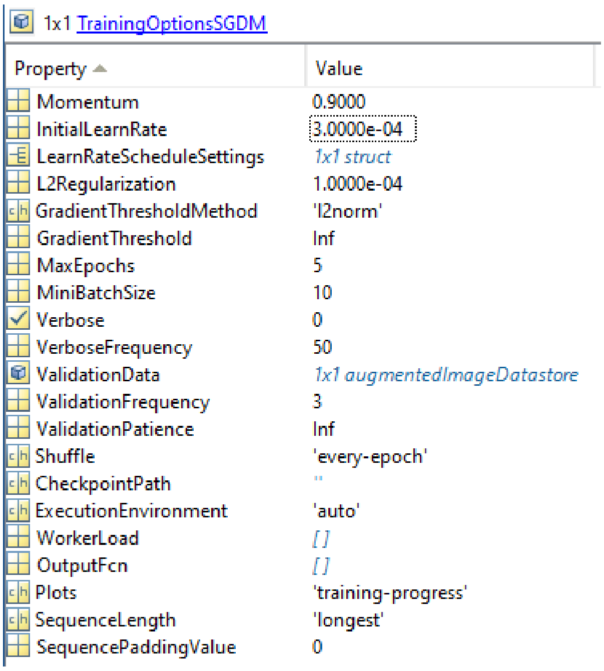
| | NAME | TYPE | ACTIVATIO... | LEARNABLES |
|---|---|---|---|---|
| 1 | data<br>227x227x3 images with 'zerocenter' normalization | Image Input | 227×227×3 | - |
| 2 | conv1<br>96 11x11x3 convolutions with stride [4 4] and padding [0 0 0 0] | Convolution | 55×55×96 | Weights 11×11×3×96<br>Bias 1×1×96 |
| 3 | relu1<br>ReLU | ReLU | 55×55×96 | - |
| 4 | norm1<br>cross channel normalization with 5 channels per element | Cross Channel Normalization | 55×55×96 | - |
| 5 | pool1<br>3x3 max pooling with stride [2 2] and padding [0 0 0 0] | Max Pooling | 27×27×96 | - |
| 6 | conv2<br>256 5x5x48 convolutions with stride [1 1] and padding [2 2 2 2] | Convolution | 27×27×256 | Weights 5×5×48×256<br>Bias 1×1×256 |
| 7 | relu2<br>ReLU | ReLU | 27×27×256 | - |
| 8 | norm2<br>cross channel normalization with 5 channels per element | Cross Channel Normalization | 27×27×256 | - |
| 9 | pool2<br>3x3 max pooling with stride [2 2] and padding [0 0 0 0] | Max Pooling | 13×13×256 | - |
| 10 | conv3<br>384 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1] | Convolution | 13×13×384 | Weights 3×3×256×384<br>Bias 1×1×384 |
| 11 | relu3<br>ReLU | ReLU | 13×13×384 | - |
| 12 | conv4<br>384 3x3x192 convolutions with stride [1 1] and padding [1 1 1 1] | Convolution | 13×13×384 | Weights 3×3×192×384<br>Bias 1×1×384 |
| 13 | relu4<br>ReLU | ReLU | 13×13×384 | - |
| 14 | conv5<br>256 3x3x192 convolutions with stride [1 1] and padding [1 1 1 1] | Convolution | 13×13×256 | Weights 3×3×192×256<br>Bias 1×1×256 |
| 15 | relu5<br>ReLU | ReLU | 13×13×256 | - |
| 16 | pool5<br>3x3 max pooling with stride [2 2] and padding [0 0 0 0] | Max Pooling | 6×6×256 | - |
| 17 | fc6<br>4096 fully connected layer | Fully Connected | 1×1×4096 | Weights 4096×9216<br>Bias 4096×1 |
| 18 | relu6<br>ReLU | ReLU | 1×1×4096 | - |
| 19 | drop6<br>50% dropout | Dropout | 1×1×4096 | - |
| 20 | fc7<br>4096 fully connected layer | Fully Connected | 1×1×4096 | Weights 4096×4096<br>Bias 4096×1 |
| 21 | relu7<br>ReLU | ReLU | 1×1×4096 | - |
| 22 | drop7<br>50% dropout | Dropout | 1×1×4096 | - |
| 23 | fc8<br>1000 fully connected layer | Fully Connected | 1×1×1000 | Weights 1000×4096<br>Bias 1000×1 |
| 24 | prob<br>softmax | Softmax | 1×1×1000 | - |
| 25 | output<br>crossentropyex with 'tench' and 999 other classes | Classification Output | - | - |

Figure 5.7: AlexNet layers and parameters

Thus an augmented image datastore is used to automatically resize the training images. The last fully connected layer and the output layer are replaced to match the number of 10 classes of this H3D scene database.

Since there are 12276 viewpoint images in the database, the train set, validation set and test set are split randomised by MATLAB at the split ratio of 70% : 15% : 15%. Therefore, in the first training there are 8593 images in train set, 1842 images in validation set and 1841 images in test set.

The parameters of the network in first training are listed in Figure 5.8. The initial learning rate is set to 0.0003, the validation frequency is set to 3 for the software to validate the network every 3 iterations during training process, and the mini batch size is set to 10. The maximum epoch number is set to 5 for the first training.

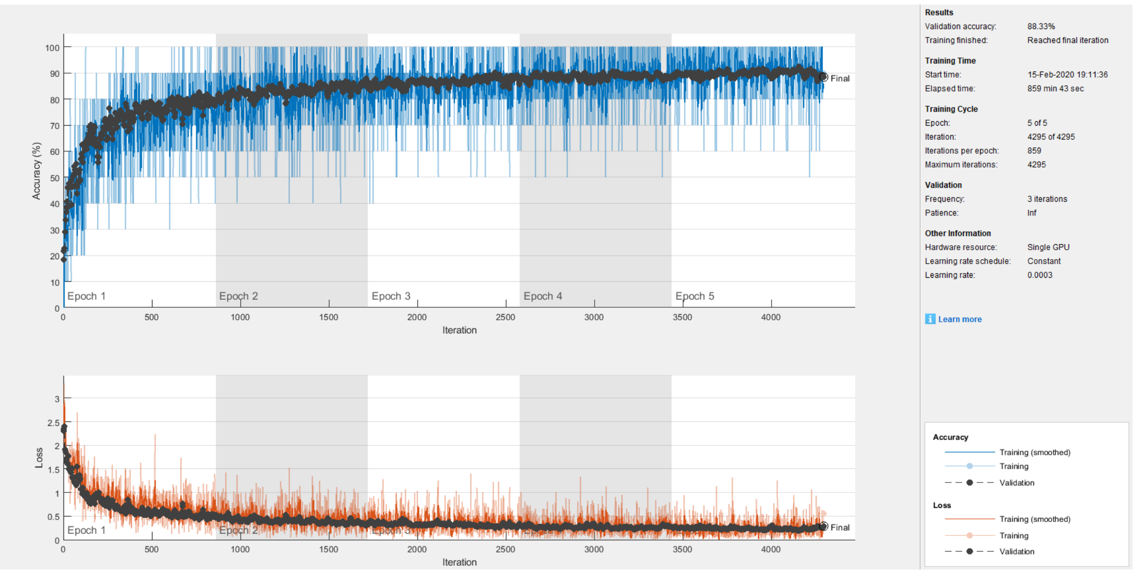


1x1 TrainingOptionsSGDM

| Property | Value |
|---|---|
| Momentum | 0.9000 |
| InitialLearnRate | 3.0000e-04 |
| LearnRateScheduleSettings | 1x1 struct |
| L2Regularization | 1.0000e-04 |
| GradientThresholdMethod | 'l2norm' |
| GradientThreshold | Inf |
| MaxEpochs | 5 |
| MiniBatchSize | 10 |
| Verbose | 0 |
| VerboseFrequency | 50 |
| ValidationData | 1x1 augmentedImageDatastore |
| ValidationFrequency | 3 |
| ValidationPatience | Inf |
| Shuffle | 'every-epoch' |
| CheckpointPath | '' |
| ExecutionEnvironment | 'auto' |
| WorkerLoad | [ ] |
| OutputFcn | [ ] |
| Plots | 'training-progress' |
| SequenceLength | 'longest' |
| SequencePaddingValue | 0 |

Figure 5.8: Parameters of the network in first training

The training progress is shown in Figure 5.9, including the diagrams of accuracy and loss. There are 4295 iterations in total 5 epochs and 859 iterations in each epoch. The network takes 5 epochs in 859 min 43 sec to train on single NVIDIA Tesla K40c GPU.



**Results**
Validation accuracy: 88.33%
Training finished: Reached final iteration

**Training Time**
Start time: 15-Feb-2020 19:11:36
Elapsed time: 859 min 43 sec

**Training Cycle**
Epoch: 5 of 5
Iteration: 4295 of 4295
Iterations per epoch: 859
Maximum iterations: 4295

**Validation**
Frequency: 3 iterations
Patience: Inf

**Other Information**
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.0003

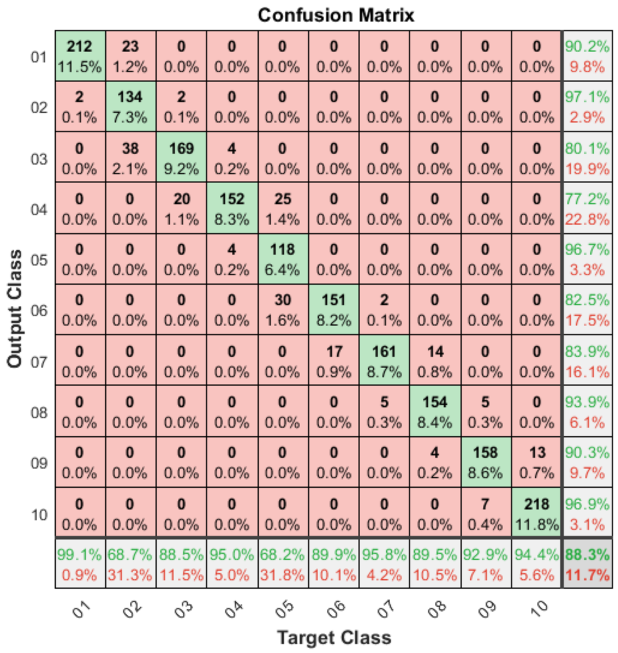
Figure 5.9: Progress of the first training

### 5.2.5 Results and Performance Evaluation

As can be seen in Figure 5.9, the blue curve is the training accuracy curve and the black curve is the validation accuracy curve. The trend of accuracy is growing rapidly at the beginning of epoch 1 and keeps growing steadily during the whole training progress. It tends to be stable at the end of epoch 5, and the accuracy is over 90% at some points in epoch 5. The curves of training and validation are overlapped at the end of epoch 5. The final accuracy of validation is 88.33%. The trend of loss is dropping rapidly at the beginning, and tends to be convergent and under 0.5 at the end of epoch 5.

For purpose of comparing the accuracy of test set with validation set, the test set is classified and turned out the accuracy of 88.97%. It is approximate to the accuracy of validation. The confusion matrix of validation set is displayed in Figure 5.10 (a) and the confusion matrix of test set is displayed in Figure 5.10 (b) for comparison.

Comparing the two confusion matrix, there are a number of similarities. In both validation set and test set, the network performs quite well on label of 01, 04, 07 and 10. The 4 labels all have nearly or over 95% accuracy. For the worst performance labels, label 02 and 05 share the least accuracy of less than 80%. The network tends to classify label 02 to its adjacent labels - label 01 and 03, and same as label 05. This phenomenon can be understandable because the adjacent labels share similar scenes and contain similar features to some extent. On the other hand, label 02 and 05 contain the fewest test images in quantity among all labels, which might lead to high misjudgement rate of the network. Label 01 and 10 contain the most quantity of viewpoint images due to at the beginning and the end of each shooting process, the RC car is running at a slower speed, which makes video segments 01 and 10 have longer elapsed time. As the train set is randomised chosen from the database, these two labels might have more training data than other labels on the probability. Further more, label 01 and 10 share the most quantity of test images, thus, the high accuracy of the two labels can be demonstrated.

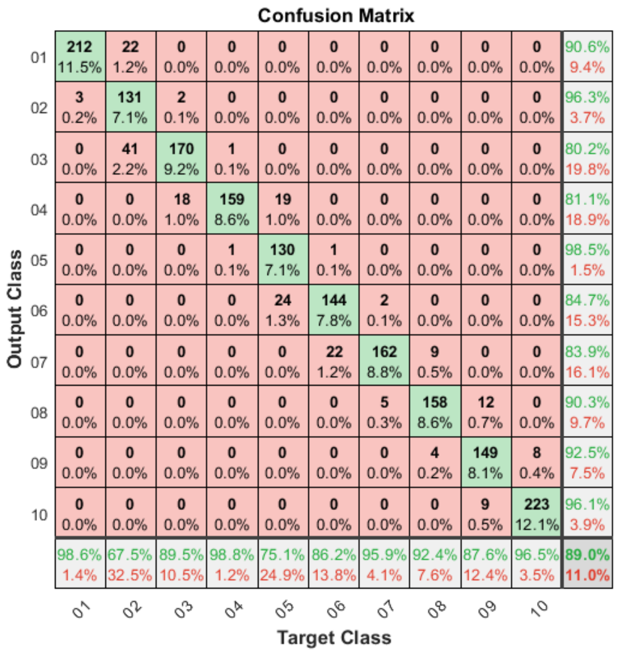


Confusion Matrix

| Output Class | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 01 | 212<br>11.5% | 23<br>1.2% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 90.2%<br>9.8% |
| 02 | 2<br>0.1% | 134<br>7.3% | 2<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 97.1%<br>2.9% |
| 03 | 0<br>0.0% | 38<br>2.1% | 169<br>9.2% | 4<br>0.2% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 80.1%<br>19.9% |
| 04 | 0<br>0.0% | 0<br>0.0% | 20<br>1.1% | 152<br>8.3% | 25<br>1.4% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 77.2%<br>22.8% |
| 05 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 4<br>0.2% | 118<br>6.4% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 96.7%<br>3.3% |
| 06 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 30<br>1.6% | 151<br>8.2% | 2<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 82.5%<br>17.5% |
| 07 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 17<br>0.9% | 161<br>8.7% | 14<br>0.8% | 0<br>0.0% | 0<br>0.0% | 83.9%<br>16.1% |
| 08 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 5<br>0.3% | 154<br>8.4% | 5<br>0.3% | 0<br>0.0% | 93.9%<br>6.1% |
| 09 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 4<br>0.2% | 158<br>8.6% | 13<br>0.7% | 90.3%<br>9.7% |
| 10 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 7<br>0.4% | 218<br>11.8% | 96.9%<br>3.1% |
| | 99.1%<br>0.9% | 68.7%<br>31.3% | 88.5%<br>11.5% | 95.0%<br>5.0% | 68.2%<br>31.8% | 89.9%<br>10.1% | 95.8%<br>4.2% | 89.5%<br>10.5% | 92.9%<br>7.1% | 94.4%<br>5.6% | 88.3%<br>11.7% |

Target Class

(a)



Confusion Matrix

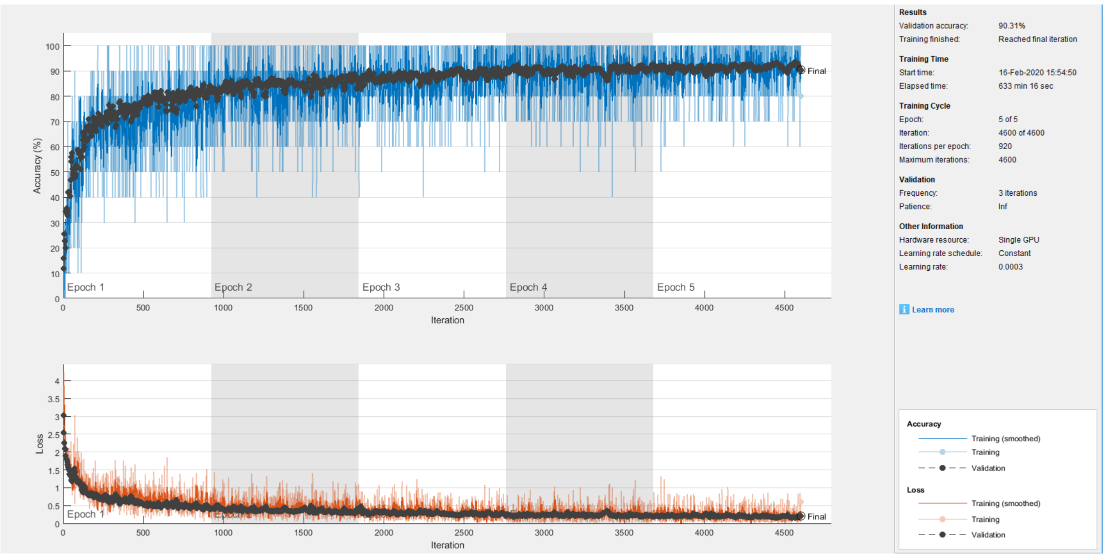
| Output Class | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 01 | 212<br>11.5% | 22<br>1.2% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 90.6%<br>9.4% |
| 02 | 3<br>0.2% | 131<br>7.1% | 2<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 96.3%<br>3.7% |
| 03 | 0<br>0.0% | 41<br>2.2% | 170<br>9.2% | 1<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 80.2%<br>19.8% |
| 04 | 0<br>0.0% | 0<br>0.0% | 18<br>1.0% | 159<br>8.6% | 19<br>1.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 81.1%<br>18.9% |
| 05 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 1<br>0.1% | 130<br>7.1% | 1<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 98.5%<br>1.5% |
| 06 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 24<br>1.3% | 144<br>7.8% | 2<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 84.7%<br>15.3% |
| 07 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 22<br>1.2% | 162<br>8.8% | 9<br>0.5% | 0<br>0.0% | 0<br>0.0% | 83.9%<br>16.1% |
| 08 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 5<br>0.3% | 158<br>8.6% | 12<br>0.7% | 0<br>0.0% | 90.3%<br>9.7% |
| 09 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 4<br>0.2% | 149<br>8.1% | 8<br>0.4% | 92.5%<br>7.5% |
| 10 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 9<br>0.5% | 223<br>12.1% | 96.1%<br>3.9% |
| | 98.6%<br>1.4% | 67.5%<br>32.5% | 89.5%<br>10.5% | 98.8%<br>1.2% | 75.1%<br>24.9% | 86.2%<br>13.8% | 95.9%<br>4.1% | 92.4%<br>7.6% | 87.6%<br>12.4% | 96.5%<br>3.5% | 89.0%<br>11.0% |

Target Class

(b)

Figure 5.10: Confusion matrix of the first training result
(a)Confusion matrix of validation set (b)Confusion matrix of test set

To prove the point of view above, the second experiment has been proceeded. The training progress of the second training is shown in Figure 5.11.



**Results**
Validation accuracy: 90.31%
Training finished: Reached final iteration

**Training Time**
Start time: 16-Feb-2020 15:54:50
Elapsed time: 633 min 16 sec

**Training Cycle**
Epoch: 5 of 5
Iteration: 4600 of 4600
Iterations per epoch: 920
Maximum iterations: 4600

**Validation**
Frequency: 3 iterations
Patience: Inf

**Other Information**
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.0003

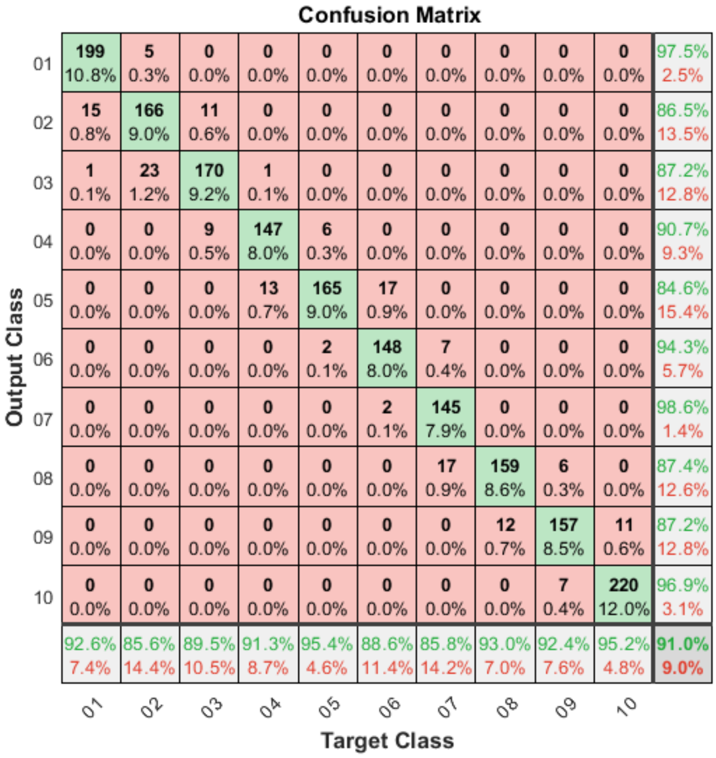
Figure 5.11: Progress of the second training

In the second training, the split ratio of the train set, validation set and test set is set to 75% : 10% : 15% and database is split randomised by MATLAB. The increase ratio of 5% in train set makes extra 614 images to fed in training. The validation frequency is still set to 3 and the maximum epoch number is 5. There are 4600 iterations in total and 920 iterations in each epoch. The elapsed time is 633 min 16 sec.

Compared with the progress of first training in Figure 5.9, the trends of accuracy and loss are very similar, however the accuracy has been increased and the loss has been decreased at the end of epoch 5. The final validation accuracy is 90.31%, which is about 2% higher than the accuracy of the first training.

When it turns to the test set, the accuracy is 91.04% and it is slightly higher than the accuracy of validation. Figure 5.12 displays the confusion matrix of the result of the test set in the second training.

Taking an overview of this confusion matrix, all the labels have their accuracy of no more than 96% and no less than 85%. Comparing with previous training, the best accuracy of each label has dropped while the poorest accuracy has been raised a lot. This means the accuracy of each label is trending towards average, and the average deviation has been reduced. Label 02, 03, 06 and 07 have the worst performance on accuracy between 85% and 90%, and all of other labels have good performances of accuracy over 90%. Label 05 and 10 has the best accuracy of over 95%. The improvement of accuracy might due to the increase of the ratio of train set.



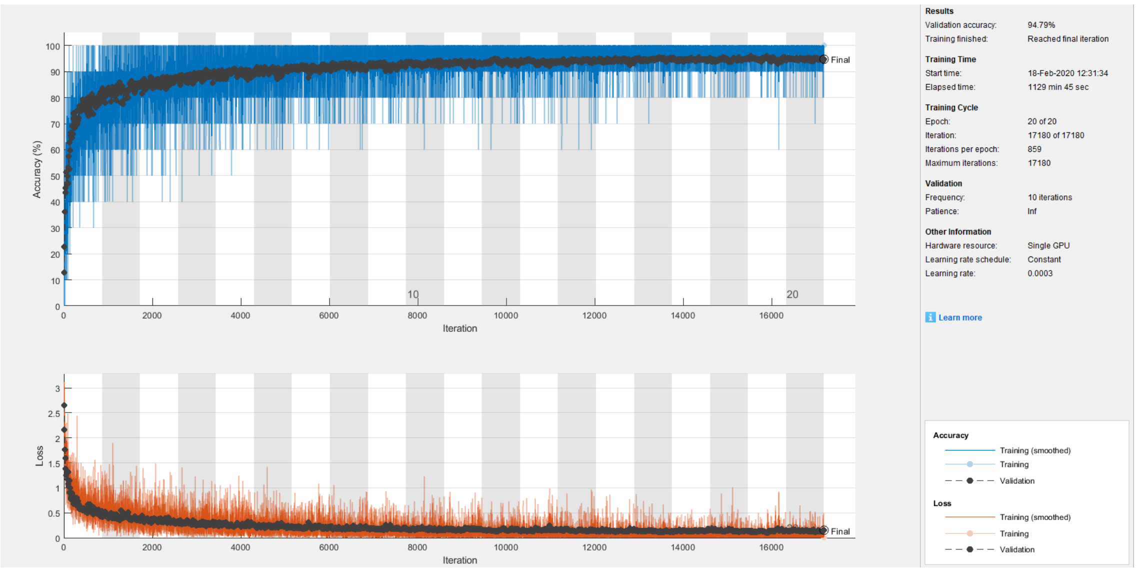
Confusion Matrix

Output Class / Target Class

| | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 01 | 199<br>10.8% | 5<br>0.3% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 97.5%<br>2.5% |
| 02 | 15<br>0.8% | 166<br>9.0% | 11<br>0.6% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 86.5%<br>13.5% |
| 03 | 1<br>0.1% | 23<br>1.2% | 170<br>9.2% | 1<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 87.2%<br>12.8% |
| 04 | 0<br>0.0% | 0<br>0.0% | 9<br>0.5% | 147<br>8.0% | 6<br>0.3% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 90.7%<br>9.3% |
| 05 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 13<br>0.7% | 165<br>9.0% | 17<br>0.9% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 84.6%<br>15.4% |
| 06 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 2<br>0.1% | 148<br>8.0% | 7<br>0.4% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 94.3%<br>5.7% |
| 07 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 2<br>0.1% | 145<br>7.9% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 98.6%<br>1.4% |
| 08 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 17<br>0.9% | 159<br>8.6% | 6<br>0.3% | 0<br>0.0% | 87.4%<br>12.6% |
| 09 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 12<br>0.7% | 157<br>8.5% | 11<br>0.6% | 87.2%<br>12.8% |
| 10 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 7<br>0.4% | 220<br>12.0% | 96.9%<br>3.1% |
| | 92.6%<br>7.4% | 85.6%<br>14.4% | 89.5%<br>10.5% | 91.3%<br>8.7% | 95.4%<br>4.6% | 88.6%<br>11.4% | 85.8%<br>14.2% | 93.0%<br>7.0% | 92.4%<br>7.6% | 95.2%<br>4.8% | 91.0%<br>9.0% |

Figure 5.12: Confusion matrix of the second training result

In order to improve the accuracy while keeping the split ratio of the dataset, the third experiment has been proceeded. Figure 5.13 displays the training process of the third experiment. The split ratio is as same as the first time, that is set to 70% : 15% : 15%. By observing the training process of previous two training in Figure 5.9 and Figure 5.11, the accuracy of both still seems to has a trend to increase. If the learning iterations continue, it may turns out a better accuracy. Based on this assumption, in this training, the maximum epoch number is set to 20 to get more training iterations. The validation frequency is set to 10 for time saving. There are 17180 iterations in total, which is four times of the first training, and 859 iterations in each epoch. The elapsed time is 1129 min 45 sec, and it is longer than the previous two.



**Results**

| | |
|---|---|
| Validation accuracy: | 94.79% |
| Training finished: | Reached final iteration |

**Training Time**

| | |
|---|---|
| Start time: | 18-Feb-2020 12:31:34 |
| Elapsed time: | 1129 min 45 sec |

**Training Cycle**

| | |
|---|---|
| Epoch: | 20 of 20 |
| Iteration: | 17180 of 17180 |
| Iterations per epoch: | 859 |
| Maximum iterations: | 17180 |

**Validation**

| | |
|---|---|
| Frequency: | 10 iterations |
| Patience: | Inf |

**Other Information**

| | |
|---|---|
| Hardware resource: | Single GPU |
| Learning rate schedule: | Constant |
| Learning rate: | 0.0003 |

Figure 5.13: Progress of the third training

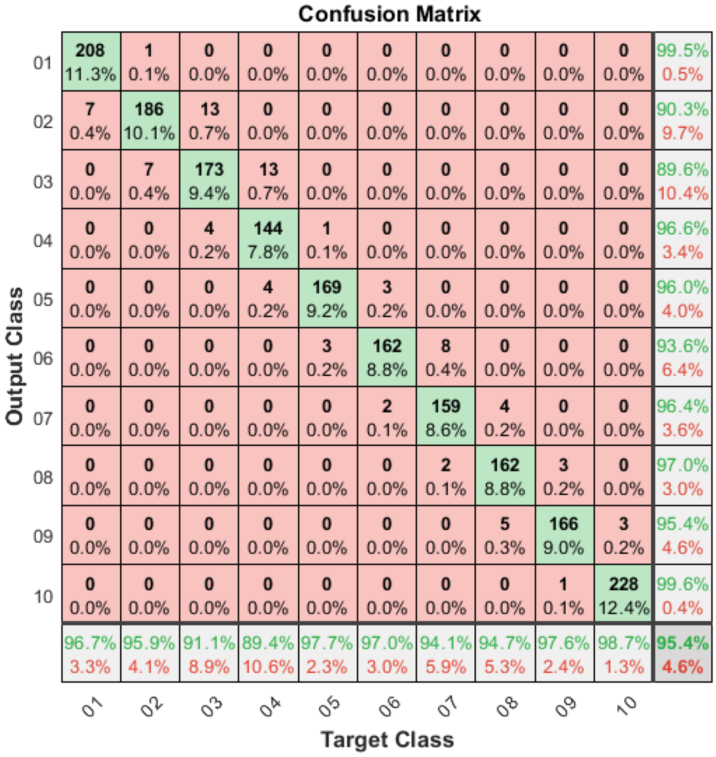
In the third training process, the trends of accuracy and loss are very similar with previous two times, however the accuracy has been increased at the end of epoch 20. The final validation accuracy is 94.79%, which is about 3.5% higher than the accuracy of the second training, and about 6% higher than the first training.

When it turns to the test set, the accuracy is 95.44% and it is slightly higher than the accuracy of validation. Figure 5.14 displays the confusion matrix of the result of the test set in the third training.

Comparing with the second training, when it comes to the errors, the network still tends to classify labels to their adjacent labels, who share similar scenes and contain similar features to some extent. The accuracy of label 02, 06 and 07, which performed not well in last training, have been increased to nearly 95% or higher. Label 04 has the worst accuracy of 89.4% and label 03 has the accuracy of 91.1%. Almost all of other labels have good performances of accuracy over 95%.

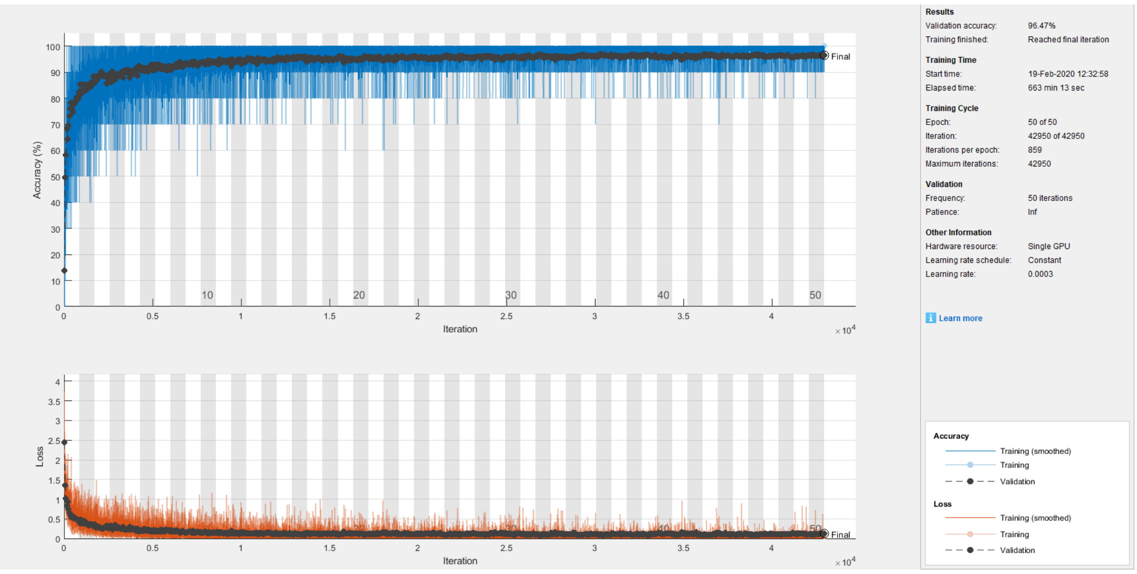
In order to let the accuracy to reach the plateaus, the fourth training with 50 epochs is proceeded. The split ratio is keeping the same as 70% : 15% : 15%, and the validation frequency is set to 895. The network takes 42950 iterations of 50 epochs in 663 min 13 sec to train.

As can be seen from the training process in Figure 5.15, the line of accuracy has reached a plateaus and the loss has been convergent to the minimum compared with previous training processes. The final validation accuracy turns to 96.47% and the test accuracy is 96.31% that is approximate to the validation accuracy. The confusion matrix of the test result is displayed in Figure 5.16.



| Output Class | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 01 | 208<br>11.3% | 1<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 99.5%<br>0.5% |
| 02 | 7<br>0.4% | 186<br>10.1% | 13<br>0.7% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 90.3%<br>9.7% |
| 03 | 0<br>0.0% | 7<br>0.4% | 173<br>9.4% | 13<br>0.7% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 89.6%<br>10.4% |
| 04 | 0<br>0.0% | 0<br>0.0% | 4<br>0.2% | 144<br>7.8% | 1<br>0.1% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 96.6%<br>3.4% |
| 05 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 4<br>0.2% | 169<br>9.2% | 3<br>0.2% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 96.0%<br>4.0% |
| 06 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 3<br>0.2% | 162<br>8.8% | 8<br>0.4% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 93.6%<br>6.4% |
| 07 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 2<br>0.1% | 159<br>8.6% | 4<br>0.2% | 0<br>0.0% | 0<br>0.0% | 96.4%<br>3.6% |
| 08 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 2<br>0.1% | 162<br>8.8% | 3<br>0.2% | 0<br>0.0% | 97.0%<br>3.0% |
| 09 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 5<br>0.3% | 166<br>9.0% | 3<br>0.2% | 95.4%<br>4.6% |
| 10 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 1<br>0.1% | 228<br>12.4% | 99.6%<br>0.4% |
| | 96.7%<br>3.3% | 95.9%<br>4.1% | 91.1%<br>8.9% | 89.4%<br>10.6% | 97.7%<br>2.3% | 97.0%<br>3.0% | 94.1%<br>5.9% | 94.7%<br>5.3% | 97.6%<br>2.4% | 98.7%<br>1.3% | 95.4%<br>4.6% |

Figure 5.14: Confusion matrix of the third training result



**Results**
Validation accuracy: 96.47%
Training finished: Reached final iteration

**Training Time**
Start time: 19-Feb-2020 12:32:58
Elapsed time: 663 min 13 sec

**Training Cycle**
Epoch: 50 of 50
Iteration: 42950 of 42950
Iterations per epoch: 859
Maximum iterations: 42950

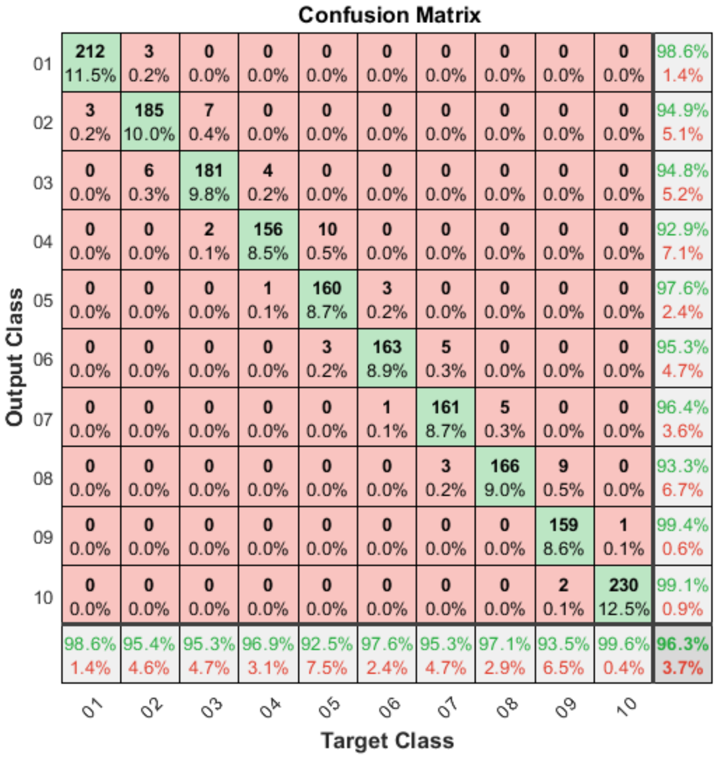
**Validation**
Frequency: 50 iterations
Patience: Inf

**Other Information**
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.0003

**Accuracy**
Training (smoothed)
Training
Validation

**Loss**
Training (smoothed)
Training
Validation

Figure 5.15: Progress of the fourth training



Confusion Matrix

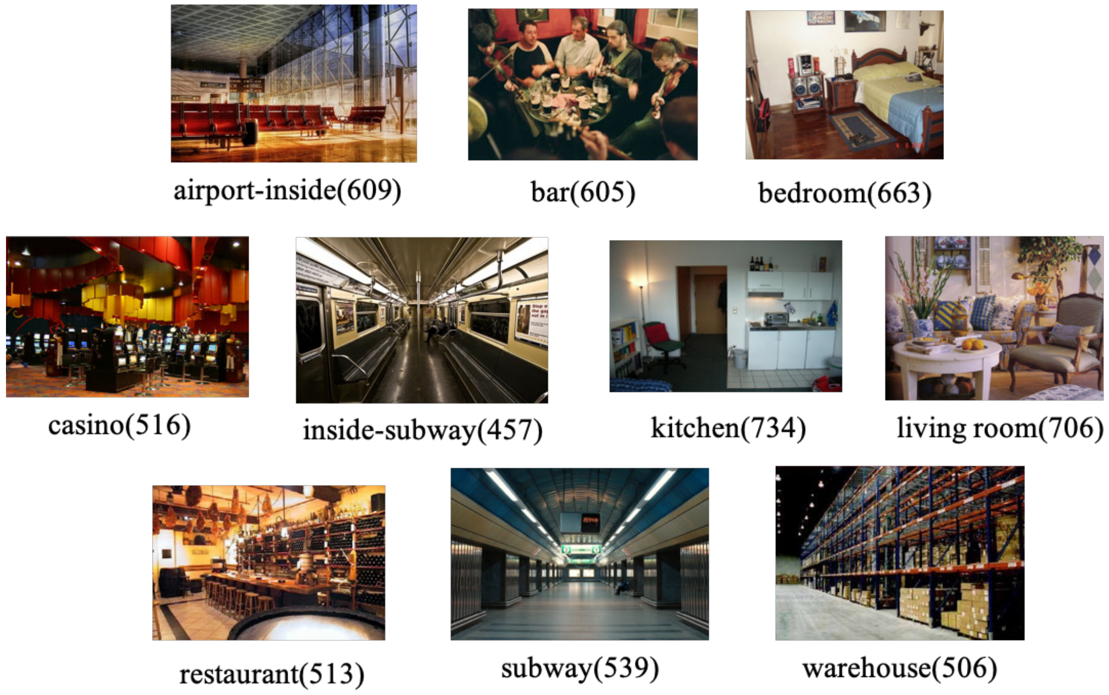
| | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 01 | 212<br>11.5% | 3<br>0.2% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 98.6%<br>1.4% |
| 02 | 3<br>0.2% | 185<br>10.0% | 7<br>0.4% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 94.9%<br>5.1% |
| 03 | 0<br>0.0% | 6<br>0.3% | 181<br>9.8% | 4<br>0.2% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 94.8%<br>5.2% |
| 04 | 0<br>0.0% | 0<br>0.0% | 2<br>0.1% | 156<br>8.5% | 10<br>0.5% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 92.9%<br>7.1% |
| 05 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 1<br>0.1% | 160<br>8.7% | 3<br>0.2% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 97.6%<br>2.4% |
| 06 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 3<br>0.2% | 163<br>8.9% | 5<br>0.3% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 95.3%<br>4.7% |
| 07 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 1<br>0.1% | 161<br>8.7% | 5<br>0.3% | 0<br>0.0% | 0<br>0.0% | 96.4%<br>3.6% |
| 08 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 3<br>0.2% | 166<br>9.0% | 9<br>0.5% | 0<br>0.0% | 93.3%<br>6.7% |
| 09 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 159<br>8.6% | 1<br>0.1% | 99.4%<br>0.6% |
| 10 | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 0<br>0.0% | 2<br>0.1% | 230<br>12.5% | 99.1%<br>0.9% |
| | 98.6%<br>1.4% | 95.4%<br>4.6% | 95.3%<br>4.7% | 96.9%<br>3.1% | 92.5%<br>7.5% | 97.6%<br>2.4% | 95.3%<br>4.7% | 97.1%<br>2.9% | 93.5%<br>6.5% | 99.6%<br>0.4% | 96.3%<br>3.7% |

Figure 5.16: Confusion matrix of the fourth training result

As can be seen from the confusion matrix in Figure 5.16, the accuracy of all the labels has been increased. Label 10 has the best accuracy of 99.6%, label 05 and 09 have the least accuracy of 92.5% and 93.5%, while all of other labels have accuracy above 95%. This final accuracy of over 96% can be determined as an outstanding performance of the network. The deep convolutional neural network of AlexNet displays its superiority on H3D scene database.

In order to compare with normal 2D scene database, an open source indoor scene recognition is used here [191] [192]. This database contains 67 Indoor categories, and a total of 15620 images. Because the number of images varies across categories, ten categories with the most images are selected. The example image of each category and the number of images in each category is shown in Figure 5.17. There are 5844 images of 10 categories, while H3D scene database contains 12276 images, so these selected indoor scene images have been flipped to mirror images. Along with original images, there are 11688 images in this 2D indoor scene database, which is as similar size as H3D scene database.

The split ratio of 2D indoor scene database is set to 75% : 10% : 15%, thus there are 8766 images in train set and 1753 images in test set. This split ratio maintains similar quantity of images of train set and test set as H3D scene database. Other training parameters stays same with the last training. The training progress is displayed in Figure 5.18.

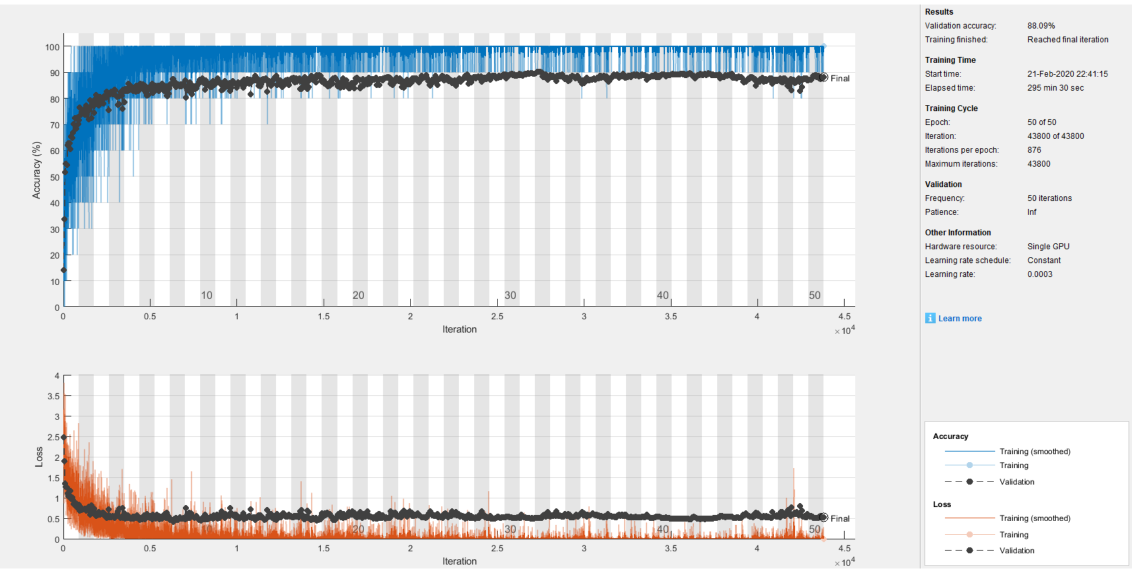


airport-inside(609) bar(605) bedroom(663)

casino(516) inside-subway(457) kitchen(734) living room(706)

restaurant(513) subway(539) warehouse(506)

Figure 5.17: 2D indoor scene database example
[191] [192]



**Results**

Validation accuracy: 88.09%

Training finished: Reached final iteration

**Training Time**

Start time: 21-Feb-2020 22:41:15

Elapsed time: 295 min 30 sec

**Training Cycle**

Epoch: 50 of 50

Iteration: 43800 of 43800

Iterations per epoch: 876

Maximum iterations: 43800

**Validation**

Frequency: 50 iterations

Patience: Inf

**Other Information**

Hardware resource: Single GPU

Learning rate schedule: Constant

Learning rate: 0.0003

Figure 5.18: Training process of 2D indoor scene database

As can be seen in Figure 5.18, the accuracy curve of training is almost 100% at the end of epoch 50, but the validation accuracy stays around 90%, at the same time the loss curve of training is almost 0, while the validation loss is still very high at about 0.5. This means the network performs very well on train set but performs not well on validation set because it does not really learn the features of this database. The final accuracy of validation is 88.09% and accuracy of test set is 88.82%, they are both worse than the result of H3D scene database. Comparing the results of 2D indoor scene database training on AlexNet demonstrated that H3D scene database has a remarkable performance on AlexNet.

## 5.3 Holoscopic 3D Perception Platform on Real Car

Beside capturing H3D video on RC car for simple environment, when facing the real world traffic conditions, like multiple vehicle streams and pedestrians on main street, the real car is the primary that H3D perception platform will be applied on. This is a more challenging task that contains many unknown factors.

### 5.3.1 YOLOv3

You only look once (YOLO) [116] is a state-of-the-art object detection system targeted for real-time processing. YOLOv3 made an improvement base on YOLO and YOLOv2. On a Pascal Titan X it processes images at 30 FPS and has a mAP of 57.9% on COCO test-dev. Compared to 57.5 AP50 in 198 ms by RetinaNet, similar performance but 3.8 times faster [193].

Figure 5.19 displays the structure of YOLOv3 [194]. The newer architecture boasts of residual skip connections, and up-sampling. The most salient feature of YOLOv3 is that it makes detections at three different scales. YOLO is a fully convolutional network and its eventual output is generated by applying a $1 \times 1$ kernel on a feature map. In YOLOv3, the detection is done by applying $1 \times 1$ detection kernels on feature maps of three different sizes at three different places in the network.

The highlight of YOLOv3 is it uses a variant of Darknet-53. As it's name suggests, it contains of 53 convolutional layers, each followed by batch normalisation layer and Leaky ReLU activation. No form of pooling is used, and a convolutional layer with stride 2 is used to down sample the feature maps. This helps in preventing loss of low-level features often attributed to pooling [195]. The layers of Darknet-53 is shown in Figure 5.20 [193].

### 5.3.2 Experiment Setup

The route of data shooting is selected in Uxbridge town, which contains main roads, side roads, car park and roundabout. As shown in Figure 5.21, the driving direction is from Brunel University car park, through Brunel main entrance, then towards Hayes End, turning around to Uxbridge centre, then turning around at the roundabout, finally driving back to Brunel University.

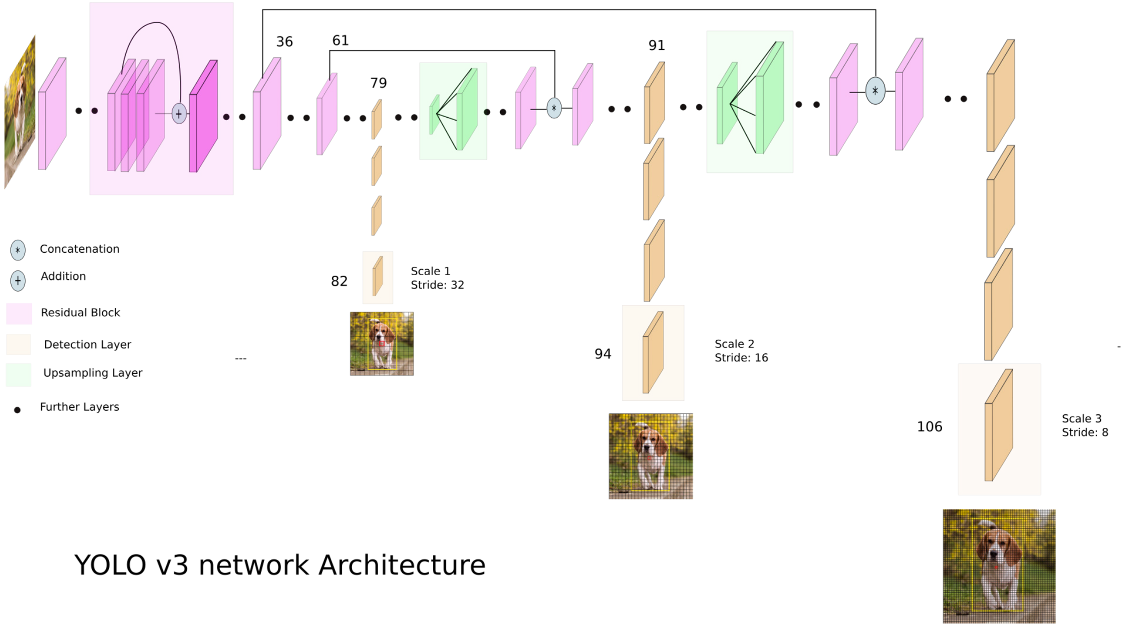


Figure 5.19: YOLOv3 structure
[194]

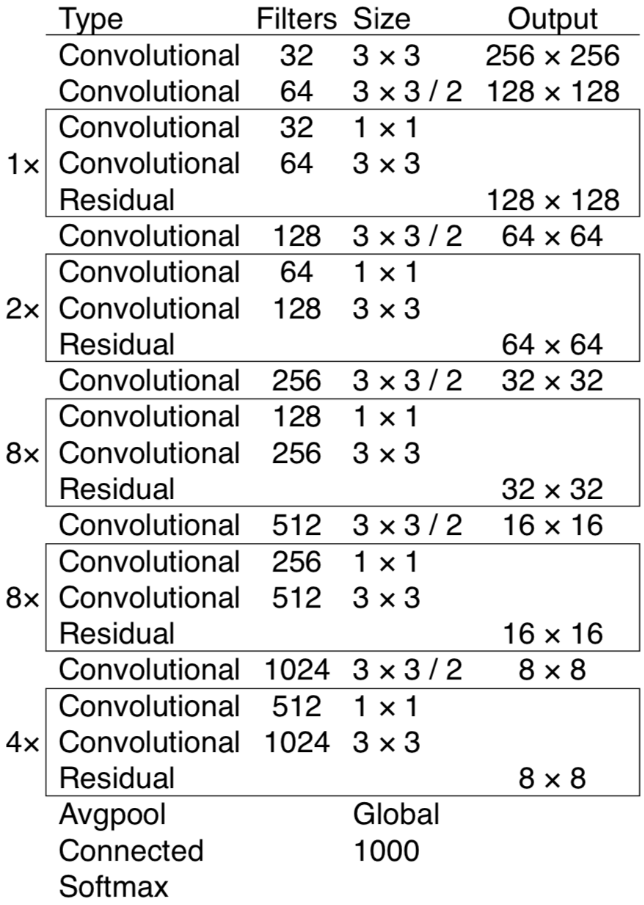
The timings of video shooting should be chosen depending on the ambient light and weather. The H3D camera settings of ISO and shutter can be adjusted according to the light condition at the time of shooting. The resolution of the video is 3840 × 2160 and video quality is set to be the highest. The frame rate is 25 fps. The estimate time of each data shooting journey is about 20 minutes, thus in each video there are about 30000 frames.

### 5.3.3 H3D Database Acquisition and Preparation

In data acquisition, the H3D camera starts to shoot before the car starts to run. The car is driving on the same route and due to reasons of traffic light and others, the speed and time of each journey may varies.

There are 8 H3D videos have been taken and time duration of each video is about 20 minutes. These videos are captured on different time of the day, and on different weather conditions, to obtain various ambient light. Apart from the starting and parking stage the car stays at the same location, there are more than 200000 frames in total. In this experiment on real road, the database is consisted of H3D video frames, the following annotation and training are based on H3D frames. Viewpoint images are extracted as well as a backup. Figure 5.22 displays the H3D driving database structure.

Figure 5.23 displays two frames from the videos that are captured with different light conditions in different time of the day.



| | Type | Filters | Size | Output |
|---|---|---|---|---|
| | Convolutional | 32 | 3 × 3 | 256 × 256 |
| | Convolutional | 64 | 3 × 3 / 2 | 128 × 128 |
| 1× | Convolutional | 32 | 1 × 1 | |
| | Convolutional | 64 | 3 × 3 | |
| | Residual | | | 128 × 128 |
| | Convolutional | 128 | 3 × 3 / 2 | 64 × 64 |
| 2× | Convolutional | 64 | 1 × 1 | |
| | Convolutional | 128 | 3 × 3 | |
| | Residual | | | 64 × 64 |
| | Convolutional | 256 | 3 × 3 / 2 | 32 × 32 |
| 8× | Convolutional | 128 | 1 × 1 | |
| | Convolutional | 256 | 3 × 3 | |
| | Residual | | | 32 × 32 |
| | Convolutional | 512 | 3 × 3 / 2 | 16 × 16 |
| 8× | Convolutional | 256 | 1 × 1 | |
| | Convolutional | 512 | 3 × 3 | |
| | Residual | | | 16 × 16 |
| | Convolutional | 1024 | 3 × 3 / 2 | 8 × 8 |
| 4× | Convolutional | 512 | 1 × 1 | |
| | Convolutional | 1024 | 3 × 3 | |
| | Residual | | | 8 × 8 |
| | Avgpool | | Global | |
| | Connected | | 1000 | |
| | Softmax | | | |

Figure 5.20: Darknet-53
[193]

### 5.3.4 Holoscopic 3D Data Cube Preparation

In order to detect the object in the video, the video frames need to be annotated first. An open source tool for annotating digital images and videos - Computer Vision Annotation Tool (CVAT) is used for annotation. The top 3 labels appear in H3D video are car, bus and person. The video frames are selected every 50 frames to reduce content repetition. In CVAT, car, bus and person are annotated by creating bounding box around them. Figure 5.24 displays an example of an annotated frame.

When doing the annotation, the tip of creating bounding box is to select the outer edge of the object into the box. In H3D images, objects are repeating. In order to contain all the features in the bounding box, the edge of bounding box should contain the outermost edge of the object. The skills are trying to recognise some iconic things, for car i.e. the mirror, wheel, the plate, the top of the car (boundary of different colours), for person i.e. the foot, the head, the clothes (different colours). Figure 5.25 illustrates some examples of the edges of bounding boxes.

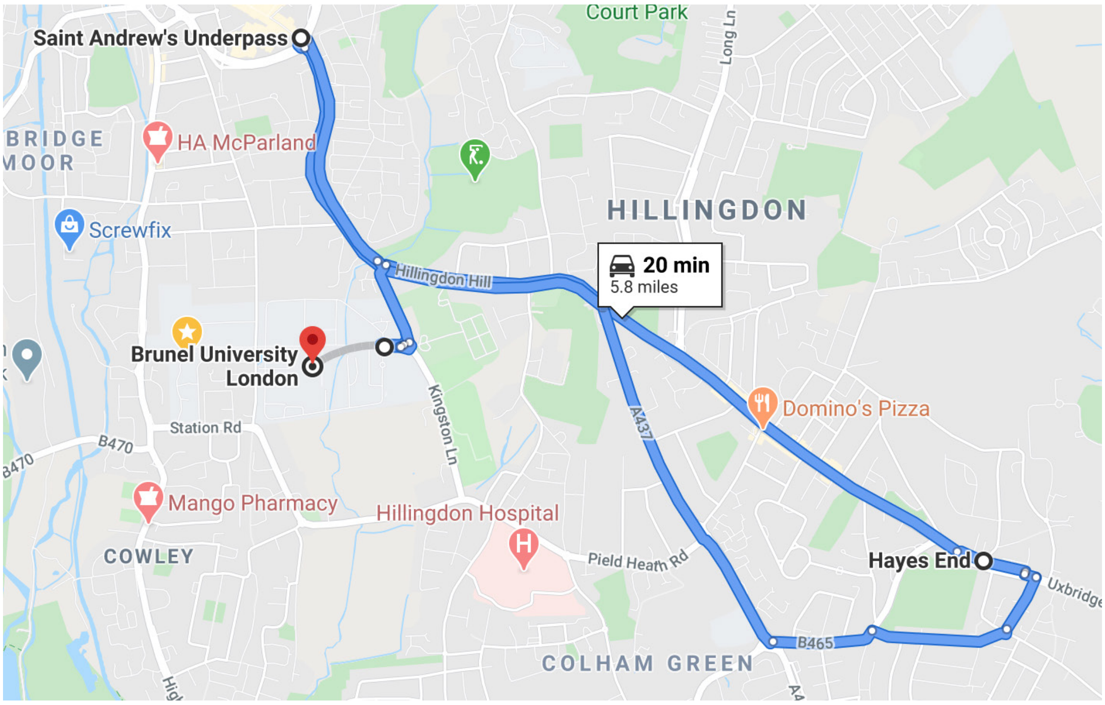


Figure 5.21: The shooting route on map

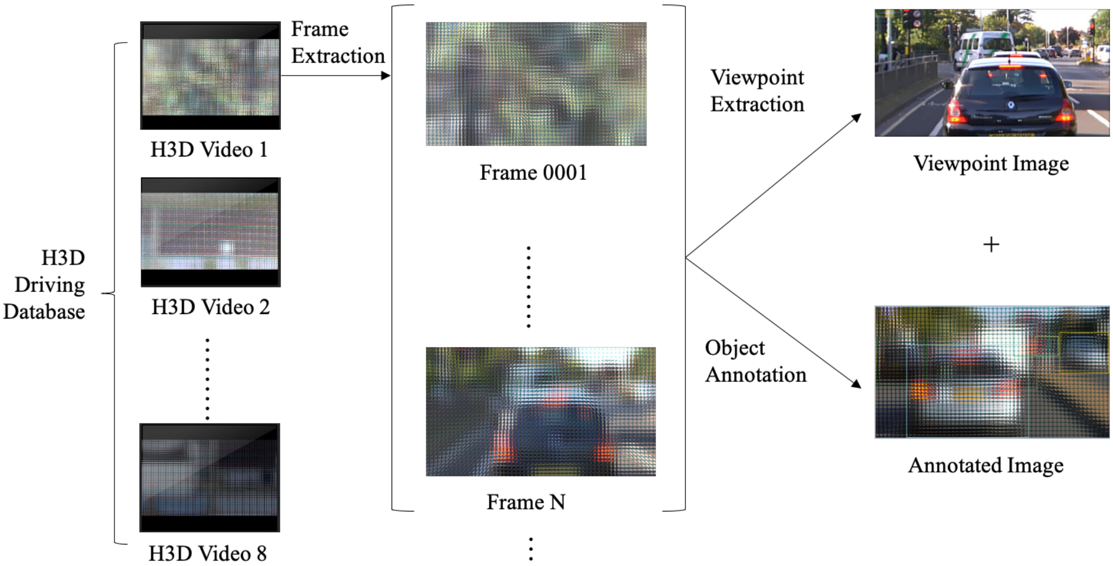
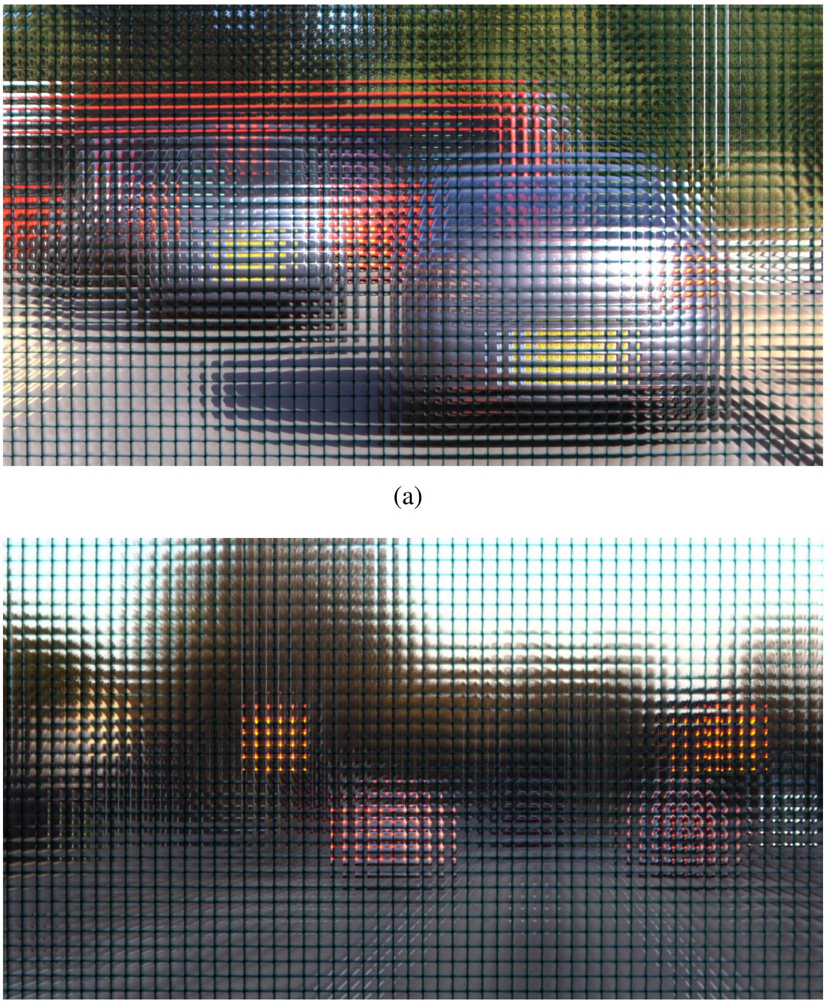


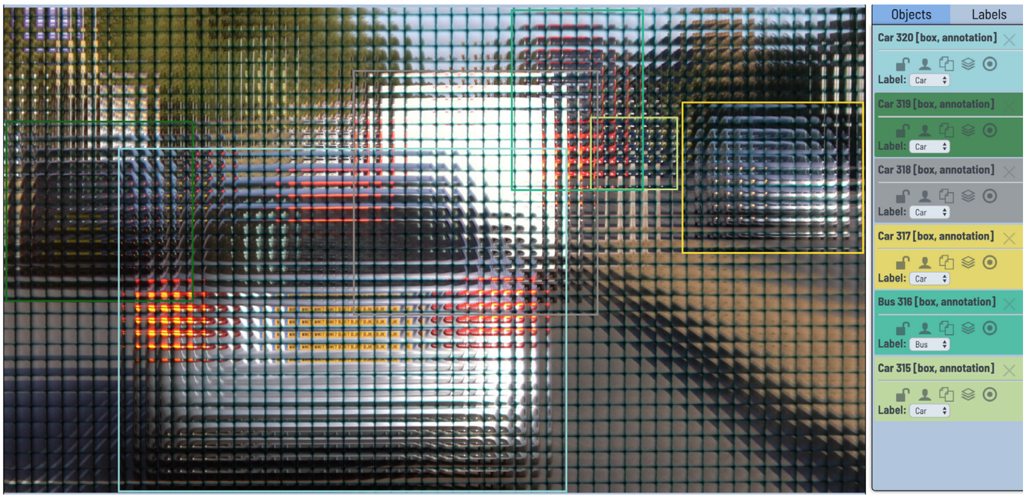
Figure 5.22: H3D driving database structure



(a)

(b)

Figure 5.23: Example frames in different light conditions
(a) Frame with good light condition (b) Frame captured at dusk



Car 320 [box, annotation]
Label: Car

Car 319 [box, annotation]
Label: Car

Car 318 [box, annotation]
Label: Car

Car 317 [box, annotation]
Label: Car

Bus 316 [box, annotation]
Label: Bus

Car 315 [box, annotation]
Label: Car

Figure 5.24: Holoscopic 3D Data Cube Preparation Process

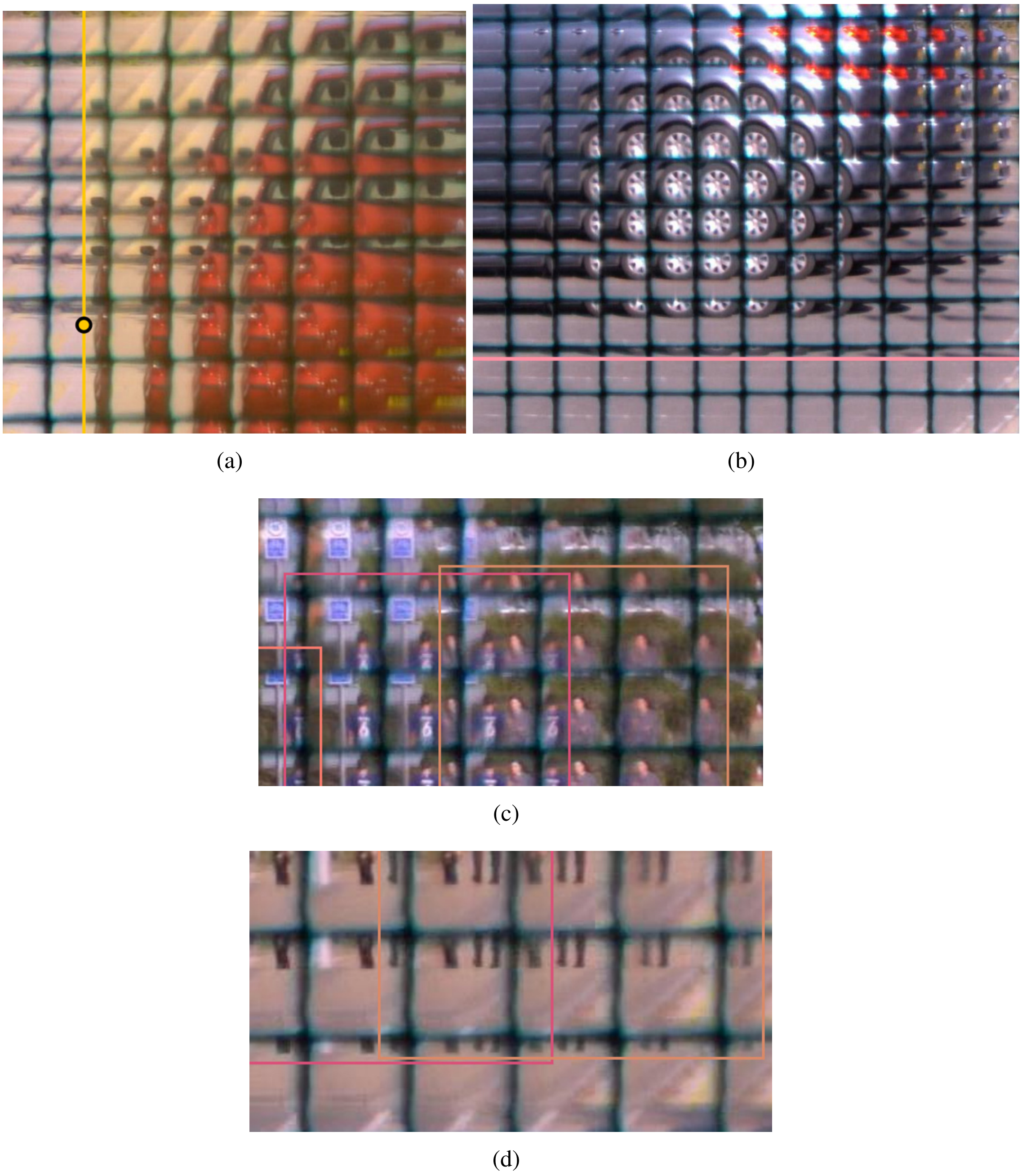


Figure 5.25: Examples in annotation
(a) Car mirror's edge (b) Car wheel's edge (c) Person's clothes (d) Person's feet

### 5.3.5 Holoscopic 3D Object Detection based on YOLOv3

The real time object detection system - YOLOv3 [193] is used to train H3D driving database in the framework of Darknet. In total there are around 2000 annotated images, of which around 1500 images are used as the train set, and around 500 images are used as test set. The split ratio of train set and validation set is around 3 : 1. The images have been trained and validated at a resolution of 800 × 448, while its original resolution is

3840 × 2160. The training used YOLOv3-spp with the standard COCO anchors with a batch size of 64 and 8 subdivisions which is a popular well known standard model. The deep learning network of YOLOv3 takes 5000 iterations in around 10 hours to train on GeForce RTX 2080 Ti. Figure 5.26 displays the training process of the first training.

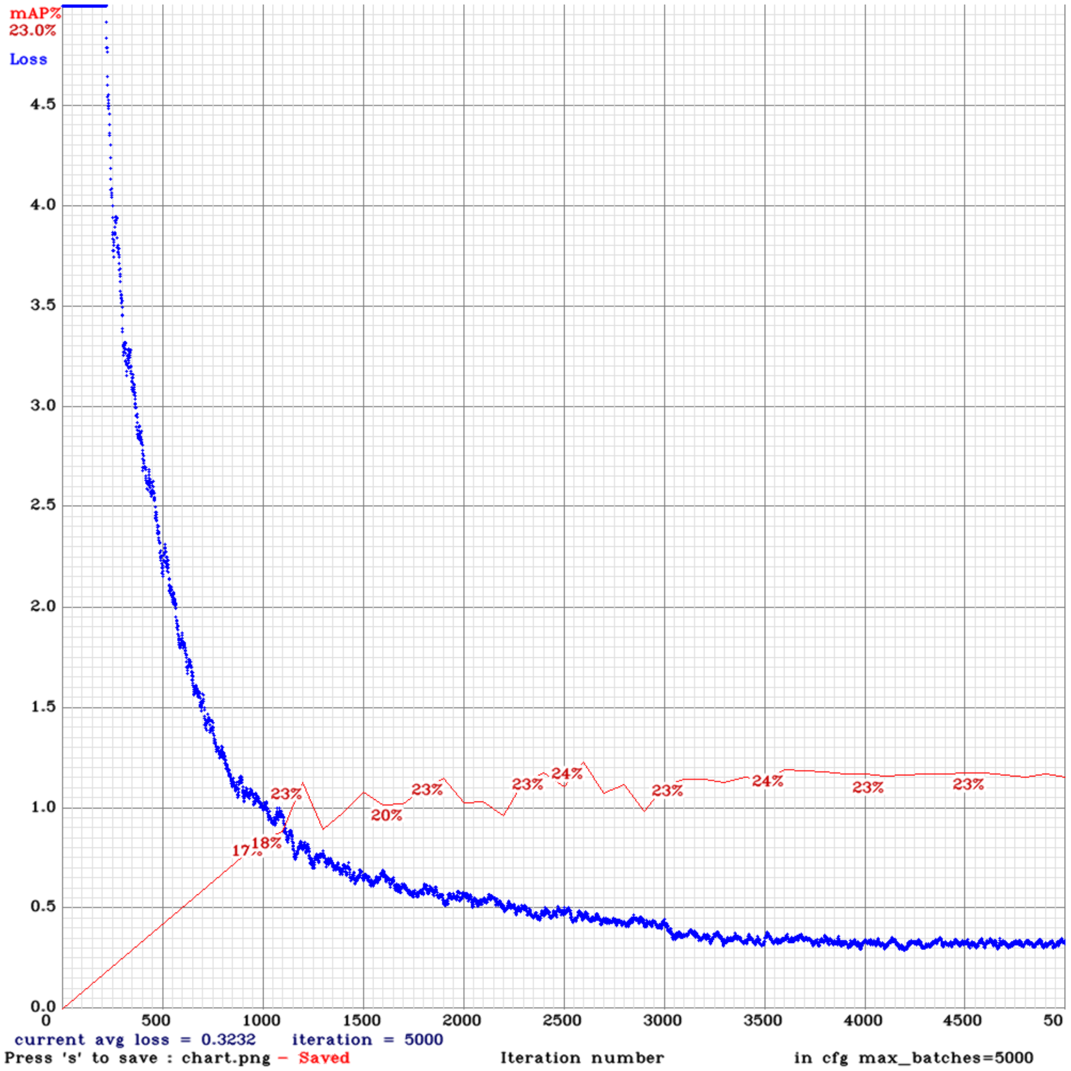


Figure 5.26: First training process of driving database

### 5.3.6 Experimental Results and Performance Evaluation

As can be seen from Figure 5.26, there is an overall mean average precision (mAP) of around 23%. This is not high because there are very few annotated examples of bus and person, which makes this result is not that meaningful. Besides, the algorithm used here has been developed for regular camera images, so the result is less than that would have been achieved with the same data with a regular camera.

The second training has dropped the person and bus examples from the dataset. The H3D driving database has 1145 frames in the train set, 359 frames in the validation set, and around 400 frames were excluded from this result because they contained either buses or persons. The training process is shown in Figure 5.27.

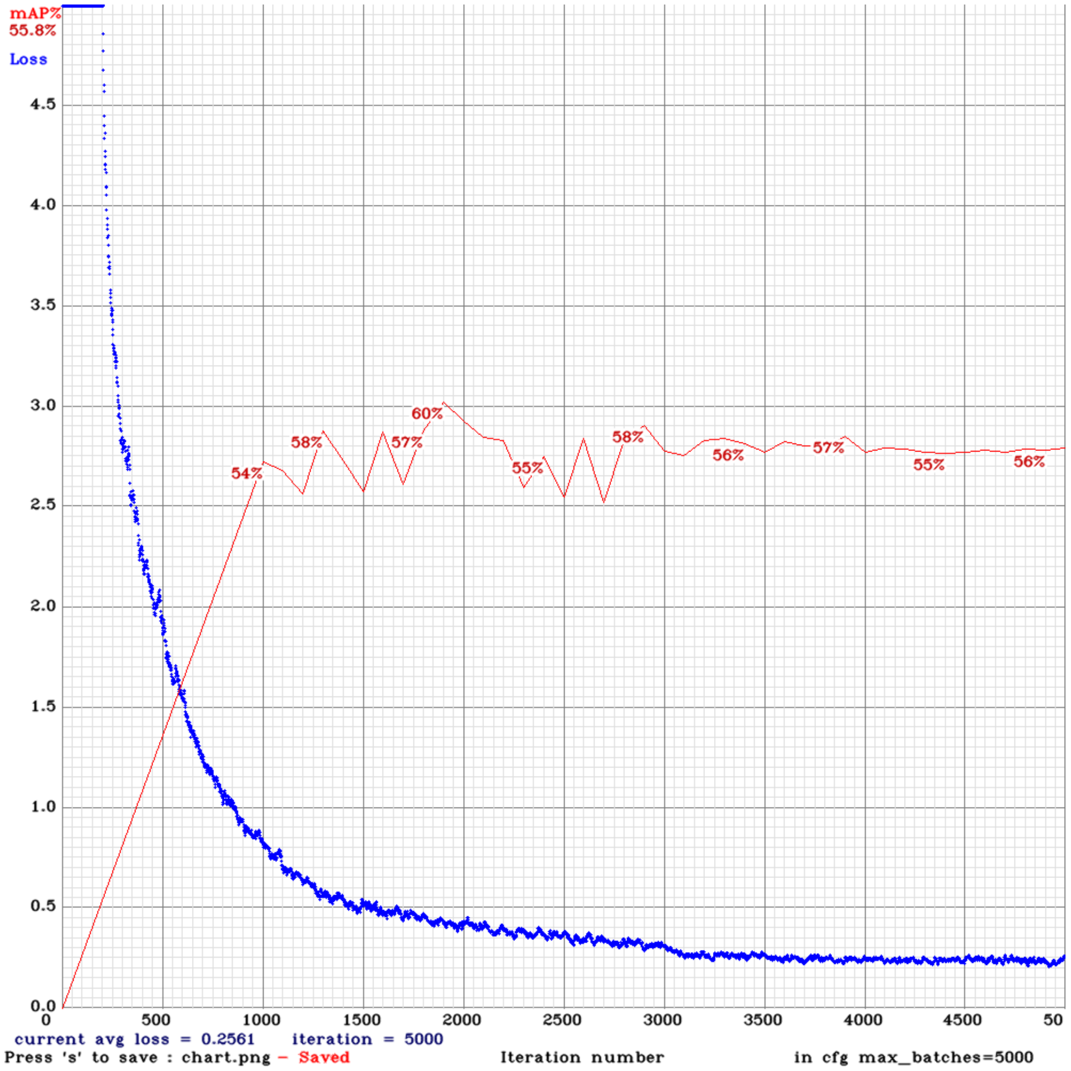


Figure 5.27: Second training process of driving database

With H3D frames of only cars from the train set and test set, the overall AP has increased to 56%. This result is not bad for baseline, however getting a really high score would probably require a more fundamental paradigmatic shift.

In order to compare the performance with 2D database, a training has been proceeded using a small car-only subset of BDD100K[196] containing only cars size-matched to the H3D dataset. In this training, the same number of frames as in H3D for the train set and around twice as much in the validation set are used, which gives a higher confidence in the statistical significance of the result.

The AP of BDD100K has achieved 66%, as H3D database achieved 56%, the score is fairly high and that means the network learned H3D database well. It has been considered to be an interesting preliminary or indicative result. The best way to get a higher statistical confidence in the comparison, would be to do more annotation on H3D data. Moreover, repeating the training again several times with different split ratio of train / validation set, would be helpful to achieve a higher score.

Table 5.2 shows the comparison between the results of car-only frames of 2D database and H3D database. A confidence threshold of 25% and an IoU threshold of 50% are used.

Table 5.2: Comparison between 2D and H3D driving database

| | 2D | H3D |
|---|---|---|
| AP | 66% | 56% |
| Precision | 0.72 | 0.70 |
| Recall | 0.65 | 0.52 |
| F1 | 0.68 | 0.60 |
| Average IoU | 53% | 52% |
| Training frames | 1145 | 1145 |
| Validation frames | 359 | 622 |

The main difference is a lower recall for H3D, which is what drags down the F1 and the AP. As the test sets are different, this comparison can only be considered to be very loosely indicative. One possibility is that the training resolution is too low to be able to capture the spatially dense information that H3D images contain - the training images are at a resolution of 3840 × 2160 - which we are shrinking down by a factor of 4.8 to 800 × 448 which is on the large side for a real-time CNN.

Training and testing at the full 3840 × 2160 would require many GPUs working in tandem and would take a very long time to train and be extremely expensive and slow to run, although down sampling H3D image is not ideal. The training at a higher resolution of 1024 × 576 is currently running again but hardware requirements increase exponentially with network resolution and AP may not increase or may actually go down due to decreased batch size.

Due to time limitation, only very small portion of holoscopic 3D content was prepared and carried out the initial training and the same process was designed for the 2D content to compare the DL algorithm performance. The experiment outcome shows the DL algorithm learns holoscopic 3D content a lot faster than 2D content and surprisingly, there is less recall compared to 2D processing. This shows that the ML/DL algorithm is the appropriate approach for reliable visual perception if the training is extended with more data with parameter adjustment.

## 5.4 Summary

The two main mission of H3D perception platform for autonomous vehicles are scene classification and object detection, in this chapter, these two experiments of H3D perception platform have been conducted on both RC car and real car. In order to face variety of surrounding circumstances, RC car is suited to not complicated environments such as indoor shooting and sidewalks, and the real car is suited to multiple vehicle streams and pedestrians on main street.

With the H3D scene database collected by RC car and H3D camera, the training using the pre-trained deep learning network - AlexNet - for scene classification finally comes out an outstanding accuracy above 96%. It is a high performance compared with similar size 2D scene database. The deep convolutional neural network of AlexNet displays its superiority on H3D scene database and it also demonstrated that H3D scene database has a remarkable performance on deep learning.

With the H3D driving database collected by real car and H3D camera, the training using YOLOv3 for object recognition finally comes out the mAP of 56% with H3D frames of only cars, which is not bad for baseline. The best way to get a higher score would be to do more annotation on H3D data, and repeating the training again several times with different split ratio of train / validation set. Due to the large resolution of H3D training images, shrinking down by a factor of 4.8 to 800 × 448 is possibly too low to be able to capture the spatially dense information that H3D images contain. Although down sampling H3D image is not ideal, training and testing at the full resolution of 3840 × 2160 would require many GPUs working in tandem and would take a very long time to train, and be extremely expensive and slow to run.

# Chapter 6

# Conclusion and Furture Work

## 6.1 Conclusion

Autonomous vehicles and sensing technologies are fast growing area due to a huge industrial needs for effective transportation thus there have been a huge investment and research on AV technologies includes its autonomous drive and control, 3D mapping and localisation as well as perception and situation awareness. State of the art AV technologies uses sensor fusion of RADAR, LiDAR and Camera sensors for perception to detect and recognise objects around the vehicle however due to different sensors complexity in its nature of different frame rate, type and perspective, it is a great challenge to do a holistic data fusion. As a result, there is a need of simplistic approach for AV perception in order to deliver safety and robustness as well as cost of different type of sensors.

As a result, this PhD research investigated innovative approach of using AI with a rather unique light field imaging principle to address the AV perception challenge. This research contributed in innovative design of holoscopic 3D sensing technology with machine learning approach, in particular deep learning algorithm for holoscopic 3D object detection and recognition. This thesis has contributed an innovative method of holoscopic 3D camera for environment perception with AI algorithms to observe scene and road objects for autonomous vehicles. The sensor of H3D camera provides rich 3D information of the environment with depth information to imitate the real world. The H3D perception platform for autonomous vehicles perceiving the environment through the H3D camera sensor mount on the vehicle and acting on the data it received by making sense of the environmental and surroundings, in the ways of classifying the scene and detecting the objects on road with AI algorithms reliably.

In order to enable this research, a specialist holoscopic 3D camera sensor is designed and prototype which is retrofitted to the vehicle for holoscopic 3D content acquisition of the vehicle drive. In the camera design phase, the component of micro-lens holder has been manufactured by 3D printer which is the key component to ensure the steady status of micro-lens array during the car driving. The 3D printer's tolerance and the accuracy of micro-lens holder ensures the micro-lens array to fit in the holder perfectly. The calibration process is the most vital part before every car driving happened. This will directly affect the quality of H3D images and videos.

Holoscopic 3D content type is evaluated using face object as face detection is hugely research and develop therefore it shall be a known area of object to evaluate the holoscopic 3D sensor to ensure its an appropriate approach for complex object detection. Therefore the H3D face database has been a great attempt of using H3D database to train neural network. With the first training on BPNN and then training on AlexNet, the accuracy of H3D face recognition has been raised using deep learning neural network. Normal 2D face database has been simulated by viewpoint images of H3D images, and with comparison of the result of 2D face database, H3D face database has illustrated its good performance on deep learning neural network training for its rich information contained. This has made the groundwork for the following experiments of H3D perception platform on car.

The H3D scene database collected on RC car has performed a quite high accuracy of over 96% on the deep learning network AlexNet for scene classification. It is believed that by training with different slit ratio of train / validation set, and with tuning the training parameters, it could achieve a higher accuracy. Even compared with similar size 2D scene database, the network of AlexNet has learned the H3D database better. The deep convolutional neural network of AlexNet displays its superiority on H3D scene database and it also demonstrated that H3D scene database has an outstanding performance on deep learning.

Due to limited time for holoscopic 3D data acquisition and preparation, the initial experiment was carried out on small portion of holoscopic 3D content with low image resolution because training process was taking more a few weeks of time. A comprehensive like-2-like comparison was carried out on both 2D images and holoscopic 3D images, the outcome has been a shock as the AI algorithm learns holoscopic 3D content a lot faster than the traditional 2D content. The H3D driving database collected on real car has came out the mAP of 56% with H3D frames of only cars training on YOLOv3 for object recognition, which is also a good result for baseline. Due to the algorithm of YOLOv3 is developed for normal 2D content, the training need to be repeated for several times with different split ratio of train / validation set with tuning of parameters. More annotation on H3D data is also needed to acquire more annotated objects in more frames. Large resolution H3D driving database contains a lot of spatial information so shrinking down the images may lose the information. Due to lack of computing power and in order to speed up the learning and experiment, training and testing at full resolution could be sacrificed and the experiment of a higher resolution can be proceeded for better result. Currently, the second experiment is ongoing for larger dataset with higher resolution images that shall be ready later for the viva time.

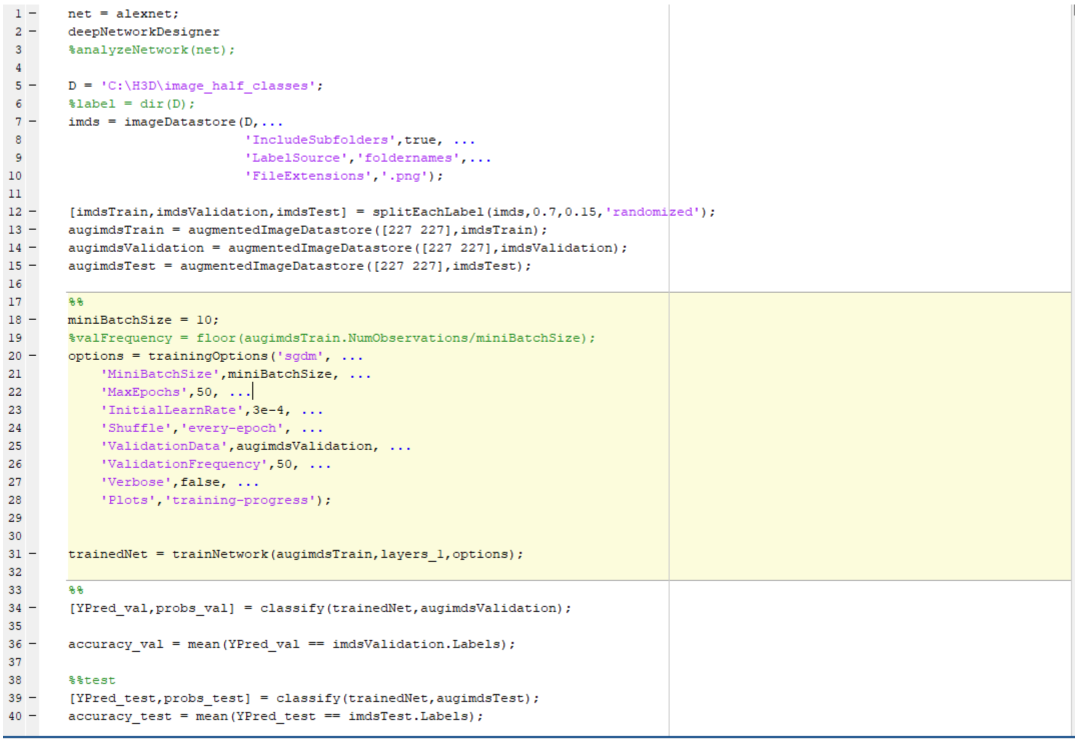
## 6.2 Future Work

Due to time limitation and lack of equipment, there are some aspects worth improving listed below to take the work further in future research.

- In terms of data preparation, due to time limitation and lack of equipment such as high performance computing and resource for 3D data preparation of over 200,000 video images which only around 2000 video images are prepared and validated. This work could be proceeded in future research.
- In terms of hardware and equipment, holoscopic 3D camera lens parameters could be studied for different focal length and aperture to identify the effect and improvement which could contribute for reliable and robust perception based object detection.
- The holoscopic 3D scene database can be prepared to extend the holoscopic 3D object detection to holoscopic 3D scene detection which is continues video of a meaningful scene such as the Brunel University roundabout based on semantics which shall support the navigation and localisation as well as the ML/DL algorithm shall be further explored with other relevant technique, e.g. ResNet as well as viewpoint based approach compared to elemental images.
- For holoscopic 3D driving database, more content shall be prepared to enlarge the dataset for future research to make the unique dataset available and enable the international research to address the global challenge of AV.

# Appendix A

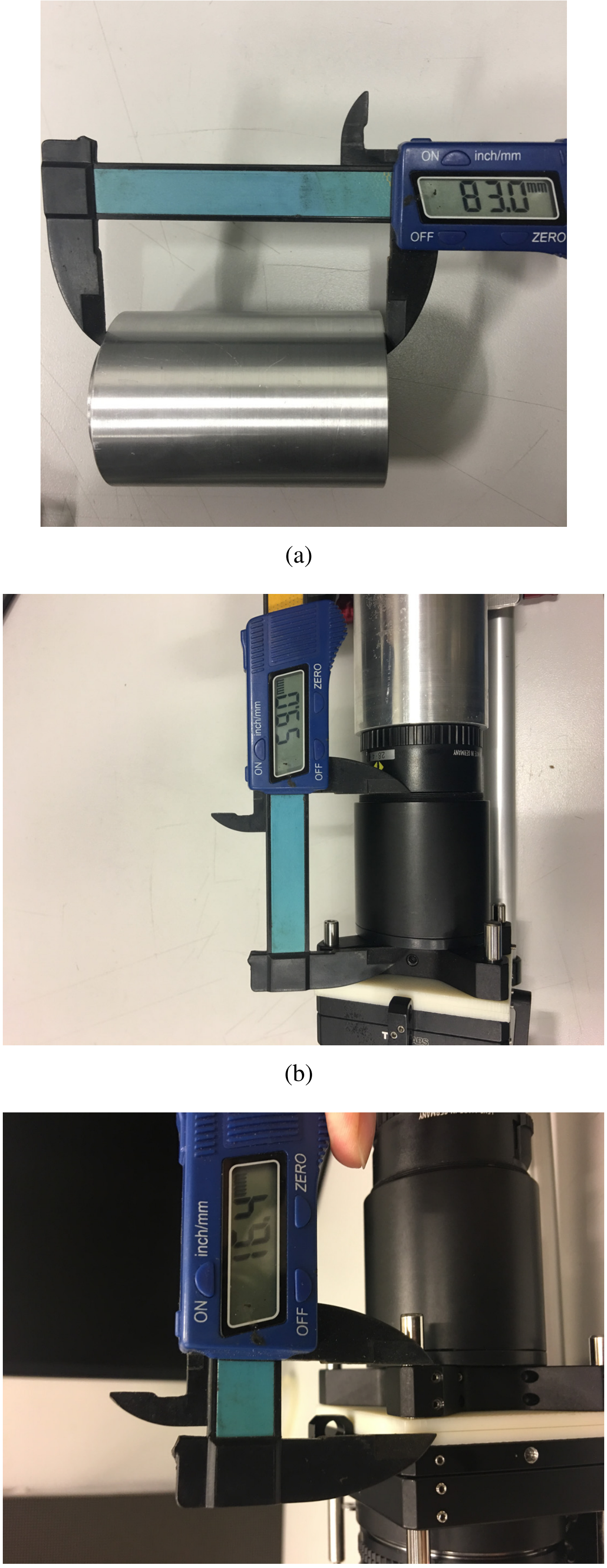
Figure A.1 displays the MATLAB code for H3D scene classification training with AlexNet.



```
net = alexnet;
deepNetworkDesigner
%analyzeNetwork(net);

D = 'C:\H3D\image_half_classes';
%label = dir(D);
imds = imageDatastore(D,...
                      'IncludeSubfolders',true, ...
                      'LabelSource','foldernames',...
                      'FileExtensions','.png');

[imdsTrain,imdsValidation,imdsTest] = splitEachLabel(imds,0.7,0.15,'randomized');
augimdsTrain = augmentedImageDatastore([227 227],imdsTrain);
augimdsValidation = augmentedImageDatastore([227 227],imdsValidation);
augimdsTest = augmentedImageDatastore([227 227],imdsTest);

%%
miniBatchSize = 10;
%valFrequency = floor(augimdsTrain.NumObservations/miniBatchSize);
options = trainingOptions('sgdm', ...
    'MiniBatchSize',miniBatchSize, ...
    'MaxEpochs',50, ...
    'InitialLearnRate',3e-4, ...
    'Shuffle','every-epoch', ...
    'ValidationData',augimdsValidation, ...
    'ValidationFrequency',50, ...
    'Verbose',false, ...
    'Plots','training-progress');


trainedNet = trainNetwork(augimdsTrain,layers_1,options);

%%
[YPred_val,probs_val] = classify(trainedNet,augimdsValidation);

accuracy_val = mean(YPred_val == imdsValidation.Labels);

%%test
[YPred_test,probs_test] = classify(trainedNet,augimdsTest);
accuracy_test = mean(YPred_test == imdsTest.Labels);
```

Figure A.1: MATLAB code for H3D scene classification

Figure A.2 displays some measurements of the H3D camera components.



(a)

(b)

(c)

Figure A.2: H3D camera components measurements

# Bibliography

[1] Christopher W. Wells. *Car Country: An Environmental History*. University of Washington Press, 2013. URL: https://www.ebook.de/de/product/24786579/christopher_w_wells_car_country.html.

[2] John A. Jakle and Keith A. Sculle. *Lots of Parking: Land Use in a Car Culture (Center Books)*. University of Virginia Press, 2004.

[3] International Organization of Motor Vehicle Manufacturers(OICA). 2018 world motor vehicle production statistics. Technical report, OICA, 2018. URL: http://www.oica.net/category/production-statistics/2018-statistics/.

[4] Paul Treffner, Rod Barrett, and Andrew Petersen. Stability and skill in driving. *Human movement science*, 21(5-6):749–784, 2002.

[5] Dennis J Crouch David L Strayer, Frank a Drews. Fatal distraction? a comparison of the cell-phone driver and the drunk driver. In *Proceedings of the 2nd International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design: Driving Assessment 2003*. University of Iowa, 2005. doi:10.17077/drivingassessment.1085.

[6] Warren Brodsky. The effects of music tempo on simulated driving performance and vehicular control. *Transportation research part F: traffic psychology and behaviour*, 4(4):219–241, 2001.

[7] Marcel Bahro, Earle Silber, Paulette Box, and Trey Sunderland. Giving up driving in alzheimer's disease - an integrative therapeutic approach. *International Journal of Geriatric Psychiatry*, 10(10):871–874, oct 1995. doi:10.1002/gps.930101010.

[8] NHTSA's National Center for Statistics and Analysis. 2017 fatal motor vehicle crashes: Overview. Technical Report DOT HS 812 603, National Highway Traffic Safety Administration, 2018. URL: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812603.

[9] NHTSA's National Center for Statistics and Analysis. Critical reasons for crashes investigated in the national motor vehicle crash causation survey. Technical Report DOT HS 812 115, National Highway Traffic Safety Administration, 2015. URL: `https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115`.

[10] Ravi Shanker, Adam Jonas, Scott Devitt, Katy Huberty, Simon Flannery, William Greene, Benjamin Swinburne, Gregory Locraft, Adam Wood, Keith Weiss, et al. Autonomous cars: Self-driving the new auto industry paradigm. In *Morgan Stanley blue paper*, pages 1–109. Morgan Stanley & Co. LLC, 2013.

[11] Panos J Antsaklis, Kevin M Passino, and SJ Wang. An introduction to autonomous control systems. *IEEE Control Systems Magazine*, 11(4):5–13, 1991.

[12] Katie Burke. *How Does a Self-Driving Car See?-Camera, radar and lidar sensors give autonomous vehicles superhuman vision.* URL: `https://blogs.nvidia.com/blog/2019/04/15/how-does-a-self-driving-car-see/`.

[13] Daniel J. Fagnant and Kara Kockelman. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice*, 77:167–181, jul 2015. `doi:10.1016/j.tra.2015.04.003`.

[14] Hubert Igliński and Maciej Babiak. Analysis of the potential of autonomous vehicles in reducing the emissions of greenhouse gases in road transport. *Procedia engineering*, 192:353–358, 2017.

[15] Felix Steck, Viktoriya Kolarova, Francisco Bahamonde-Birke, Stefan Trommer, and Barbara Lenz. How autonomous driving may affect the value of travel time savings for commuting. *Transportation research record*, 2672(46):11–20, 2018.

[16] Danny Shapiro. *Eyes on the Road: How Autonomous Cars Understand What They're Seeing*, 2016. URL: `https://blogs.nvidia.com/blog/2016/01/05/eyes-on-the-road-how-autonomous-cars-understand-what-theyre-seeing/`.

[17] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.

[18] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, and Lipika Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416–442, 2015.

[19] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *IJCAI*, pages 1151–1156, 2003.

[20] Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6):403–429, 2006.

[21] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[22] ChunHong Wu, Malcolm McCormick, Amar Aggoun, and Sun-Yuan Kung. Depth mapping of integral images through viewpoint image extraction with a hybrid disparity analysis algorithm. *Journal of Display technology*, 4(1):101–108, 2008.

[23] Hong-xia Wang, Zhi-li Xu, Zi-ping Li, and Chun-hong Wu. 3d reconstruction from integral images based on interpolation algorithm. In *5th International Symposium on Advanced Optical Manufacturing and Testing Technologies: Advanced Optical Manufacturing Technologies*, volume 7655, page 76552Z. International Society for Optics and Photonics, 2010.

[24] Mariusz Bojarski, Philip Yeres, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Lawrence Jackel, and Urs Muller. Explaining how a deep neural network trained with end-to-end learning steers a car. *arXiv preprint arXiv:1704.07911*, 2017.

[25] Kevin Lim, Paul Treitz, Michael Wulder, Benoît St-Onge, and Martin Flood. LiDAR remote sensing of forest structure. *Progress in Physical Geography: Earth and Environment*, 27(1):88–106, mar 2003. `doi:10.1191/0309133303pp360ra`.

[26] Sebastian Thrun. *TED2011 Google's Driverless Car*, 2011. URL: `https://www.ted.com/talks/sebastian_thrun_google_s_driverless_car#t-92794`.

[27] Albie Jarvis. *Renovo Brings the World's Most Advanced LiDAR Sensor from Velodyne LiDAR to its AWare Automated Mobility Ecosystem*, 2018. URL: `https://velodynelidar.com/press-release/renovo-brings-the-worlds-most-advanced-lidar-sensor-from-velodyne-lidar-to-its-aware-automated-mobility-ecosystem/`.

[28] Velodyne. *Alpha Prime*. URL: `https://store.clearpathrobotics.com/products/alpha-prime`.

[29] Shawn CARPENTER. Autonomous vehicle radar: Improving radar performance with simulation. *ANSYS [online]. Canosburg: ANSYS [cit. 2019-05-01]. Dostupné z: https://www. ansys. com/about-ansys/advantage-magazine*, 12, 2018.

[30] Daniel Göhring, Miao Wang, Michael Schnürmacher, and Tinosch Ganjineh. Radar/lidar sensor fusion for car-following on highways. In *The 5th International Conference on Automation, Robotics and Applications*, pages 407–412. IEEE, 2011.

[31] Martin Schneider. Automotive radar-status and trends. In *German microwave conference*, pages 144–147, 2005.

[32] Katie Burke. *Perception Matters: How Deep Learning Enables Autonomous Vehicles to Understand Their Environment*, 2018. URL: `https://blogs.nvidia.com/blog/2018/08/10/autonomous-vehicles-perception-layer/`.

[33] Christian Häne, Torsten Sattler, and Marc Pollefeys. Obstacle detection for self-driving cars using only monocular cameras and wheel odometry. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5101–5108. IEEE, 2015.

[34] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun. Towards fully autonomous driving: Systems and algorithms. In *Proc. IEEE Intelligent Vehicles Symp. (IV)*, pages 163–168, June 2011. `doi:10.1109/IVS.2011.5940562`.

[35] C. Stiller and J. Ziegler. 3D perception and planning for self-driving and cooperative automobiles. In *Proc. Signals Devices Int. Multi-Conf. Systems*, pages 1–7, March 2012. `doi:10.1109/SSD.2012.6198130`.

[36] Erico Guizzo. *How Google's Self-Driving Car Works*, 2011. URL: `https://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works`.

[37] Jonathan Petit, Bas Stottelaar, Michael Feiri, and Frank Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and lidar. In *Black Hat Europe*, 11/2015 2015. URL: `https://www.blackhat.com/docs/eu-15/materials/eu-15-Petit-Self-Driving-And-Connected-Cars-Fooling-Sensors-And-Tracking-Drivers-wp1.pdf`.

[38] Uwe Voelzke Gert Rudolph. *Three Sensor Types Drive Autonomous Vehicles*, 2017. URL: `https://www.fierceelectronics.com/components/three-sensor-types-drive-autonomous-vehicles`.

[39] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016.

[40] Andrew Howard. Real-time stereo visual odometry for autonomous ground vehicles. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3946–3952. IEEE, 2008.

[41] J. Kim, J. Yoo, and J. Koo. Road and lane detection using stereo camera. In *Proc. IEEE Int. Conf. Big Data and Smart Computing (BigComp)*, pages 649–652, January 2018. `doi:10.1109/BigComp.2018.00117`.

[42] Simon Hecker, Dengxin Dai, and Luc Van Gool. End-to-end learning of driving models with surround-view cameras and route planners. In *Proceedings of the european conference on computer vision (eccv)*, pages 435–453, 2018.

[43] G. H. Lee, F. Fraundorfer, and M. Pollefeys. Structureless pose-graph loop-closure with a multi-camera system on a self-driving car. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 564–571, November 2013. `doi:10.1109/IROS.2013.6696407`.

[44] G. H. Lee, F. Faundorfer, and M. Pollefeys. Motion estimation for self-driving cars with a generalized camera. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 2746–2753, June 2013. `doi:10.1109/CVPR.2013.354`.

[45] Stanford Artificial Intelligence Laboratory. *Introduction to Computer Vision*, 2015. URL: `https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html`.

[46] A. Rosenfeld. Computer vision: basic principles. *Proceedings of the IEEE*, 76(8):863–868, August 1988. `doi:10.1109/5.5961`.

[47] Charles Wheatstone. XVIII. contributions to the physiology of vision. —part the first. on some remarkable, and hitherto unobserved, phenomena of binocular vision. *Philosophical Transactions of the Royal Society of London*, 128:371–394, dec 1838. `doi:10.1098/rstl.1838.0019`.

[48] Mahmoud Afifi. *Basics of stereoscopic imaging in virtual and augmented reality systems*, 2013. URL: `https://medium.com/@mahmoudnafifi/basics-of-stereoscopic-imaging-6f69a7916cfd`.

[49] Lin Edwards. *Active Shutter 3D Technology for HDTV*, 2009. URL: `https://phys.org/news/2009-09-shutter-3d-technology-hdtv.html`.

[50] E. Dubois. A projection method to generate anaglyph stereo images. In *Proc. (Cat. No.01CH37221) and Signal Processing 2001 IEEE Int Conf. Acoustics, Speech*, volume 3, pages 1661–1664 vol.3, May 2001. doi:10.1109/ICASSP.2001.941256.

[51] B. Toperverg, O. Nikonov, V. Lauter-pasyuk, and H.j. Lauter. Towards 3d polarization analysis in neutron reflectometry. *Physica B: Condensed Matter*, 297(1-4):169–174, mar 2001. doi:10.1016/s0921-4526(00)00866-8.

[52] 3d Vision Blog. *Anaglyph, Shutter, Polarized Glasses or Autostereoscopic 3D Solution*, 2010. URL: https://3dvision-blog.com/4124-anaglyph-shutter-polarized-glasses-or-autostereoscopic-3d-solution/.

[53] M. Kurc, K. Wegner, and M. Domański. Transformation of depth maps produced by tof cameras. In *Proc. Int. Conf. Signals and Electronic Systems (ICSES)*, pages 1–4, September 2014. doi:10.1109/ICSES.2014.6948713.

[54] Dipl-Ing Bianca Hagebeuker and Product Marketing. A 3d time of flight camera for object detection. *PMD Technologies GmbH, Siegen*, 2007.

[55] Antonio Medina. Three dimensional camera and range finder, January 14 1992. US Patent 5,081,530.

[56] R. Lange and P. Seitz. Solid-state time-of-flight range camera. *IEEE Journal of Quantum Electronics*, 37(3):390–397, March 2001. doi:10.1109/3.910448.

[57] T Ringbeck, T Möller, and B Hagebeuker. Multidimensional measurement by using 3-d pmd sensors. *Advances in Radio Science: ARS*, 5:135, 2007.

[58] J. Steinbaeck, N. Druml, A. Tengg, C. Steger, and B. Hillbrand. Time-of-flight cameras for parking assistance: A feasibility study. In *Proc. 12th Int. Conf. Advanced Semiconductor Devices and Microsystems (ASDAM)*, pages 1–4, October 2018. doi:10.1109/ASDAM.2018.8544683.

[59] Stephen Hsu, Sunil Acharya, Abbas Rafii, and Richard New. Performance of a time-of-flight range camera for intelligent vehicle safety applications. In *Advanced Microsystems for Automotive Applications 2006*, pages 205–219. Springer, 2006.

[60] Omar Elkhalili, Olaf M Schrey, Wiebke Ulfig, Werner Brockherde, Bedrich J Hosticka, P Mengel, and L Listl. A 64× 8 pixel 3-d cmos time of flight image sensor for car safety applications. In *2006 Proceedings of the 32nd European Solid-State Circuits Conference*, pages 568–571. IEEE, 2006.

[61] Sebastian Anthony. *Kinect for Xbox One: An always-on, works-in-the-dark camera and microphone. What could possibly go wrong?*, 2013. URL: https://www.extremetech.com/gaming/156515-kinect-for-xbox-one-an-always-on-works-in-the-dark-camera-and-microphone-what-could-possibly-go-wrong.

[62] Eric Walz. *Autoliv to Acquire LiDAR and Time of Flight Camera Expertise From Fotonic*, 2017. URL: https://m.futurecar.com/1490/Autoliv-to-Acquire-LiDAR-and-Time-of-Flight-Camera-Expertise-From-Fotonic.

[63] G. Lippmann. Épreuves réversibles donnant la sensation du relief. *Journal de Physique Théorique et Appliquée*, 7(1):821–825, 1908. doi:10.1051/jphystap:019080070082100.

[64] A. Aggoun, E. Tsekleves, M. R. Swash, D. Zarpalas, A. Dimou, P. Daras, P. Nunes, and L. D. Soares. Immersive 3D holoscopic video system. *IEEE MultiMedia*, 20(1):28–37, January 2013. doi:10.1109/MMUL.2012.42.

[65] Amar Aggoun. 3d holoscopic imaging technology for real-time volume processing and display. In *High-Quality Visual Experience*, pages 411–428. Springer, 2010.

[66] Ju-Seog Jang and Bahram Javidi. Formation of orthoscopic three-dimensional real images in direct pickup one-step integral imaging. *Optical Engineering*, 42(7):1869–1871, 2003.

[67] Makoto Okui and Fumio Okano. 3d display research at nhk. In *Workshop on 3D Media, Applications and Devices*, 2009.

[68] Manuel Martínez-Corral, Bahram Javidi, Raúl Martínez-Cuenca, and Genaro Saavedra. Formation of real, orthoscopic integral images by smart pixel mapping. *Optics Express*, 13(23):9175–9180, 2005.

[69] Bahram Javidi, Raúl Martínez-Cuenca, Genaro Saavedra, and Manuel Martínez-Corral. Orthoscopic long-focal-depth integral imaging by hybrid method. In *Three-Dimensional TV, Video, and Display V*, volume 6392, page 639203. International Society for Optics and Photonics, 2006.

[70] Eman Alazawi. *Holoscopic 3D image depth estimation and segmentation techniques*. PhD thesis, Brunel University London, 2015.

[71] George Stockman Linda G. Shapiro. *Computer Vision*. Pearson Education (US), 2001. URL: https://www.ebook.de/de/product/3246508/linda_g_shapiro_george_stockman_computer_vision.html.

[72] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

[73] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach (2nd Edition)*. Pearson, 2011. URL: `https://www.amazon.com/Computer-Vision-Modern-Approach-2nd/dp/013608592X?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=013608592X`.

[74] Gaurav Kumar and Pradeep Kumar Bhatia. A detailed review of feature extraction in image processing systems. In *2014 Fourth international conference on advanced computing & communication technologies*, pages 5–12. IEEE, 2014.

[75] CVI-IITM. *Summer School Session 3: Features - Corner and Edge Detection*, 2018. URL: `https://iitmcvg.github.io/summer_school/Session3/`.

[76] Gaurav Y Tawde and Jayshree Kundargi. An overview of feature extraction techniques in ocr for indian scripts focused on offline handwriting. *International Journal of Engineering Research and Applications*, 3(1):919–926, 2013.

[77] Maya Nayak and Bhawani Sankar Panigrahi. Advanced signal processing techniques for feature extraction in data mining. *International Journal of Computer Applications*, 19(9):30–37, 2011.

[78] Robert Fisher, Simon Perkins, Ashley Walker, and Erik Wolfart. *Hough Transform*, 2000. URL: `https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm`.

[79] Chengjun Liu and Harry Wechsler. Independent component analysis of gabor features for face recognition. *IEEE transactions on Neural Networks*, 14(4):919–928, 2003.

[80] Bai Li and Yihui Liu. When eigenfaces are combined with wavelets. In *Applications and Innovations in Intelligent Systems IX*, pages 213–220. Springer, 2002.

[81] Jian Huang Lai, Pong C. Yuen, and Guo Can Feng. Face recognition using holistic fourier invariant features. *Pattern Recognition*, 34(1):95–109, 2001.

[82] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[83] Andrej Karpathy. *Convolutional Neural Networks for Visual Recognition*, 2016. URL: `http://cs231n.github.io/classification/`.

[84] Abdellatif Abdelfattah. *Image Classification using Deep Neural Networks — A beginner friendly approach using TensorFlow*, 2017. URL: `https://medium.com/@tifa2up/image-classification-using-deep-neural-networks-a-beginner-friendly-approach-using-tensorflow-94b0a090ccd4`.

[85] Siddhartha Sankar Nath, Girish Mishra, Jajnyaseni Kar, Sayan Chakraborty, and Nilanjan Dey. A survey of image classification methods and techniques. In *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, pages 554–557. IEEE, 2014.

[86] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[87] Honglak Lee. *Unsupervised feature learning via sparse hierarchical representations*, volume 20. Stanford University, 2010.

[88] Sahar Muneam, Mohammad Q. Jawad, and Dina Hassan. Survey and comparison of different classification techniques for select appropriate classifier of image. *Periodicals of Engineering and Natural Sciences*, 7(3):1396–1404, 2019.

[89] Dengsheng Zhang and Guojun Lu. Review of shape representation and description techniques. *Pattern recognition*, 37(1):1–19, 2004.

[90] Thair Nu Phyu. Survey of classification techniques in data mining. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, pages 18–20, 2009.

[91] Gongde Guo, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. Knn model-based approach in classification. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 986–996. Springer, 2003.

[92] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[93] Vikramaditya Jakkula. Tutorial on support vector machine (svm). *School of EECS, Washington State University*, 37, 2006.

[94] K.-K. Sung and Tomaso Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):39–51, 1998.

[95] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–761, 2011.

[96] P. N. Druzhkov and V. D. Kustikova. A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, 26(1):9–15, jan 2016. doi:10.1134/s1054661816010065.

[97] Sharif Elfouly. *Part 1: R-CNN (Object Detection)- A beginners guide to one of the most fundamental concepts in object detection.*, 2019. URL: https://towardsdatascience.com/r-cnn-3a9beddfd55a.

[98] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678, 2014.

[99] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[100] Paul Viola and Michael J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[101] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017.

[102] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2722–2730, 2015.

[103] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

[104] Jason Brownlee. *A Gentle Introduction to Object Recognition With Deep Learning*, 2019. URL: https://machinelearningmastery.com/object-recognition-with-deep-learning/.

[105] Z. Zhao, P. Zheng, S. Xu, and X. Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, November 2019. doi:10.1109/TNNLS.2018.2876865.

[106] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[107] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[108] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *Proceedings. international conference on image processing*, volume 1, pages I–I. IEEE, 2002.

[109] Yoav Freund and Robert E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

[110] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[111] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[112] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[113] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[114] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2147–2154, 2014.

[115] Mahyar Najibi, Mohammad Rastegari, and Larry S. Davis. G-cnn: an iterative grid based object detector. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2369–2377, 2016.

[116] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[117] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[118] Huiyu Zhou, Abdul H. Sadka, Mohammad R. Swash, Jawid Azizi, and Umar A. Sadiq. Feature extraction and clustering for dynamic video summarisation. *Neurocomputing*, 73(10-12):1718–1729, 2010.

[119] Marcos Vinicius Mussel Cirne and Helio Pedrini. A video summarization method based on spectral clustering. In *Iberoamerican Congress on Pattern Recognition*, pages 479–486. Springer, 2013.

[120] Jiang Peng and Qin Xiao-Lin. Keyframe-based video summary using visual attention clues. *IEEE MultiMedia*, 17(2):64–73, 2009.

[121] Michael A. Smith and Takeo Kanade. Video skimming and characterization through the combination of image and language understanding. In *Proceedings 1998 IEEE International Workshop on Content-Based Access of Image and Video Database*, pages 61–70. IEEE, 1998.

[122] Yihong Gong and Xin Liu. Video summarization using singular value decomposition. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 2, pages 174–180. IEEE, 2000.

[123] Denis Simakov, Yaron Caspi, Eli Shechtman, and Michal Irani. Summarizing visual data using bidirectional similarity. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[124] Yu-Fei Ma, Lie Lu, Hong-Jiang Zhang, and Mingjing Li. A user attention model for video summarization. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 533–542, 2002.

[125] Giles M Foody and Ajay Mathur. Toward intelligent training of supervised image classifications: directing training data acquisition for svm classification. *Remote Sensing of Environment*, 93(1-2):107–117, 2004.

[126] Jong-Sen Lee, Mitchell R Grunes, Thomas L Ainsworth, Li-Jen Du, Dale L Schuler, and Shane R Cloude. Unsupervised classification using polarimetric decomposition and the complex wishart classifier. *IEEE Transactions on Geoscience and Remote Sensing*, 37(5):2249–2258, 1999.

[127] M. Guillaumin, J. Verbeek, and C. Schmid. Multimodal semi-supervised learning for image classification. In *Proc. IEEE Computer Society Conf. Computer Vision*

*and Pattern Recognition*, pages 902–909, June 2010. doi:10.1109/CVPR.2010.5540120.

[128] Andrew G. Barto Richard S. Sutton. *Reinforcement Learning*. The MIT Press, 1998. URL: https://www.ebook.de/de/product/3643662/richard_s_sutton_andrew_g_barto_reinforcement_learning.html.

[129] Sovan Lek and J. F. Guégan. Artificial neural networks as a tool in ecological modelling, an introduction. *Ecological Modelling*, 120(2-3):65–73, aug 1999. doi:10.1016/s0304-3800(99)00092-7.

[130] Awan-Ur-Rahman. *What is Artificial Neural Network and How it mimics the Human Brain?*, 2019. URL: https://medium.com/analytics-vidhya/what-is-artificial-neural-network-and-how-it-mimics-the-human-brain-f92c45564e20.

[131] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, jan 2015. doi:10.1016/j.neunet.2014.09.003.

[132] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, oct 1986. doi:10.1038/323533a0.

[133] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[134] A. Olgac and Bekir Karlik. Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence And Expert Systems*, 1:111–122, 02 2011.

[135] Jun Han and Claudio Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *Lecture Notes in Computer Science*, pages 195–201. Springer Berlin Heidelberg, 1995. doi:10.1007/3-540-59497-3_175.

[136] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[137] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, feb 2005. doi:10.1007/s10479-005-5724-z.

[138] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, mar 1951. doi:10.1214/aoms/1177729694.

[139] Michael Nielsen. *Why are deep neural networks hard to train?*, 2019. URL: http://neuralnetworksanddeeplearning.com/chap5.html.

[140] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[141] Jon Shlens. *Train your own image classifier with Inception in TensorFlow*, 2016. URL: https://ai.googleblog.com/2016/03/train-your-own-image-classifier-with.html.

[142] Christopher Olah. *Understanding Convolutions*, 2014. URL: http://colah.github.io/posts/2014-07-Understanding-Convolutions/.

[143] Stanford Vision Lab. *ImageNet*, 2016. URL: http://www.image-net.org/.

[144] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *Procedings of the British Machine Vision Conference 2015*. British Machine Vision Association, 2015. doi:10.5244/c.29.41.

[145] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998. doi:10.1109/5.726791.

[146] Maurice Peemen. *Convolutional Neural Network (CNN)*, 2013. URL: https://sites.google.com/site/5kk73gpu2013/assignment/cnn.

[147] Mohammad Swash et al. *Holoscopic 3D imaging and display technology: Camera/processing/display*. PhD thesis, Brunel University London., 2013.

[148] Sony. *alpha 7R E-mount Camera with Full Frame Sensor*. URL: https://www.sony.co.uk/electronics/interchangeable-lens-cameras/ilce-7r.

[149] Nikon. *Nikkor AF-D 35mm f/2.0*. URL: https://www.photospecialist.co.uk/nikkor-af-d-35mm-f-2-0?gclid=EAIaIQobChMIOIfqto_y7QIVlvlRCh0d6AxHEAQYAiABEgKPD_D_BwE.

[150] Rodenstock. *User manual for Rodenstock 50mm f/2.8 APO-Rodagon N Enlarging Lens 452340*. URL: https://www.pdf-manuals.com/rodenstock-50mm-f-2-8-apo-rodagon-n-enlarging-lens-452340-115079-manual.

[151] Changwon Jang, Keehoon Hong, Jiwoon Yeom, and Byoungho Lee. See-through integral imaging display using a resolution and fill factor-enhanced lens-array holographic optical element. *Optics Express*, 22(23):27958, nov 2014. `doi:10.1364/oe.22.027958`.

[152] Alex Pentland. Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Transactions on Pattern analysis and machine intelligence*, 22(1):107–119, 2000.

[153] Jie Pan, Xue-Song Wang, and Yu-Hu Cheng. Single-sample face recognition based on lpp feature transfer. *IEEE Access*, 4:2873–2884, 2016.

[154] H Chan and WW Bledsoe. A man-machine facial recognition system: some preliminary results. *Panoramic Research Inc., Palo Alto, CA, USA1965*, 1965.

[155] Gerald J Kaufman and Kenneth J Breeding. The automatic recognition of human faces from profile silhouettes. *IEEE Transactions on systems, Man, and Cybernetics*, SMC-6(2):113–121, 1976.

[156] Leon D Harmon and Willard F Hunt. Automatic recognition of human face profiles. *Computer Graphics and Image Processing*, 6(2):135–156, 1977.

[157] LD Harmon, MK Khan, Richard Lasch, and PF Ramig. Machine identification of human faces. *Pattern Recognition*, 13(2):97–110, 1981.

[158] Lahoucine Ballihi, Boulbaba Ben Amor, Mohamed Daoudi, Anuj Srivastava, and Driss Aboutajdine. Boosting 3-d-geometric features for efficient face recognition and gender classification. *IEEE Transactions on Information Forensics and Security*, 7(6):1766–1779, dec 2012. `doi:10.1109/tifs.2012.2209876`.

[159] Jian-Gang Wang and Eric Sung. Facial feature extraction in an infrared image by proxy with a visible face image. *IEEE Transactions on Instrumentation and Measurement*, 56(5):2057–2066, oct 2007. `doi:10.1109/tim.2007.904567`.

[160] Federico M. Sukno, John L. Waddington, and Paul F. Whelan. 3-d facial landmark localization with asymmetry patterns and shape regression from incomplete local features. *IEEE Transactions on Cybernetics*, 45(9):1717–1730, sep 2015. `doi:10.1109/tcyb.2014.2359056`.

[161] Bo-Gun Park, Kyoung-Mu Lee, and Sang-Uk Lee. Face recognition using face-ARG matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1982–1988, dec 2005. `doi:10.1109/tpami.2005.243`.

[162] Alan L Yuille, Peter W Hallinan, and David S Cohen. Feature extraction from faces using deformable templates. *International journal of computer vision*, 8(2):99–111, 1992.

[163] P. Mohanty, S. Sarkar, and R. Kasturi. From scores to face templates: A model-based approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2065–2078, dec 2007. `doi:10.1109/tpami.2007.1129`.

[164] Anil Kumar Sao and B. Yegnanarayana. Face verification using template matching. *IEEE Transactions on Information Forensics and Security*, 2(3):636–641, sep 2007. `doi:10.1109/tifs.2007.902920`.

[165] Xiaoguang Lu and A.K. Jain. Deformation modeling for robust 3d face matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8):1346–1357, aug 2008. `doi:10.1109/tpami.2007.70784`.

[166] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[167] Jun Zhang, Yong Yan, and M. Lades. Pace recognition: eigenface, elastic matching, and neural nets. *Proceedings of the IEEE*, 85(9):1423–1435, 1997. `doi:10.1109/5.628712`.

[168] Martin Lades, Jan C Vorbruggen, Joachim Buhmann, Jörg Lange, Christoph Von Der Malsburg, Rolf P Wurtz, and Wolfgang Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on computers*, 42(3):300–311, 1993.

[169] Ho-Chul Shin, Jae Hee Park, and Seong-Dae Kim. Combination of warping robust elastic graph matching and kernel-based projection discriminant analysis for face recognition. *IEEE Transactions on Multimedia*, 9(6):1125–1136, oct 2007. `doi:10.1109/tmm.2007.898933`.

[170] C. Kotropoulos, A. Tefas, and I. Pitas. Frontal face authentication using morphological elastic graph matching. *IEEE Transactions on Image Processing*, 9(4):555–560, apr 2000. `doi:10.1109/83.841933`.

[171] Hugo Proença and Juan C. Briceño. Periocular biometrics: constraining the elastic graph matching algorithm to biologically plausible distortions. *IET Biometrics*, 3(4):167–175, dec 2014. `doi:10.1049/iet-bmt.2013.0039`.

[172] Ara V Nefian and Monson H Hayes. Hidden markov models for face recognition. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 5, pages 2721–2724. IEEE, 1998.

[173] Jen-Tzung Chien and Chih-Pin Liao. Maximum confidence hidden markov modeling for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):606–616, apr 2008. `doi:10.1109/tpami.2007.70715`.

[174] M.A. Mohamed and P. Gader. Generalized hidden markov models. II. application to handwritten word recognition. *IEEE Transactions on Fuzzy Systems*, 8(1):82–94, 2000. `doi:10.1109/91.824774`.

[175] Johan Lim and Kyungsuk Pyun. Cost-effective hidden markov model-based image segmentation. *IEEE Signal Processing Letters*, 16(3):172–175, mar 2009. `doi:10.1109/lsp.2008.2008586`.

[176] Guangshu Hu. Digital signal processing: theory, algorithm and implementation. *Publisher ofQingHua University, Beijing*, 2003.

[177] Stanford Vision Lab. *Large Scale Visual Recognition Challenge 2012 (ILSVRC2012)*, 2012. URL: `http://www.image-net.org/challenges/LSVRC/2012/results.html`.

[178] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[179] Hao Gao. *A Walk-through of AlexNet*, 2017. URL: `https://medium.com/@smallfishbigsea/a-walk-through-of-alexnet-6cbd137a5637`.

[180] Sunita Nayak. *Understanding AlexNet*, 2018. URL: `https://neurohive.io/en/popular-networks/https://www.learnopencv.com/understanding-alexnet/`.

[181] Muneeb ul Hassan. *AlexNet – ImageNet Classification with Deep Convolutional Neural Networks*, 2018. URL: `https://neurohive.io/en/popular-networks/alexnet-imagenet-classification-with-deep-convolutional-neural-networks/`.

[182] Jerry Wei. *AlexNet: The Architecture that Challenged CNNs*, 2019. URL: `https://towardsdatascience.com/alexnet-the-architecture-that-challenged-cnns-e406d5297951`.

[183] P Jonathon Phillips, Sandor Z Der, Patrick J Rauss, and Or Z Der. *FERET (face recognition technology) recognition algorithm development and test results*. Army Research Laboratory Adelphi, MD, 1996.

[184] A Georghiades, P Belhumeur, and D Kriegman. Yale face database. *Center for computational Vision and Control at Yale University, http://cvc. yale. edu/projects/yalefaces/yalefa*, 2:6, 1997.

[185] Benjamin Weyrauch, Bernd Heisele, Jennifer Huang, and Volker Blanz. Component-based face recognition with 3d morphable models. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on*, pages 85–85. IEEE, 2004.

[186] Mohammad Rafiq Swash, Amar Aggoun, O Abdulfatah, B Li, JC Fernandez, E Alazawi, and Emmanuel Tsekleves. Pre-processing of holoscopic 3d image for autostereoscopic 3d displays. In *3D Imaging (IC3D), 2013 International Conference on*, pages 1–5. IEEE, 2013.

[187] Chengjun Liu and Harry Wechsler. Independent component analysis of gabor features for face recognition. *IEEE transactions on Neural Networks*, 14(4):919–928, 2003.

[188] Bai Li and Yihui Liu. When eigenfaces are combined with wavelets. In *Applications and Innovations in Intelligent Systems IX*, pages 213–220. Springer, 2002.

[189] Jian Huang Lai, Pong C Yuen, and Guo Can Feng. Face recognition using holistic fourier invariant features. *Pattern Recognition*, 34(1):95–109, 2001.

[190] Jason Brownlee. *What is a Confusion Matrix in Machine Learning*, 2016. URL: `https://machinelearningmastery.com/confusion-matrix-machine-learning/`.

[191] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2009. `doi:10.1109/cvpr.2009.5206537`.

[192] Aude Oliva. *Indoor Scene Recognition - Database*, 2009. URL: `http://web.mit.edu/torralba/www/indoor.html`.

[193] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[194] Ayoosh Kathuria. *What's new in YOLO v3?*, 2018. URL: `https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b`.

[195] Python Lessons. *YOLO v3 theory explained*, 2019. URL: `https://medium.com/analytics-vidhya/yolo-v3-theory-explained-33100f6d193`.

[196] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5), 2018. `arXiv:http://arxiv.org/abs/1805.04687v1`.