

**Please cite the Published Version**

Kendrick, Connah (2018) Markerless facial motion capture: deep learning approaches on RGBD data. Doctoral thesis (PhD), Manchester Metropolitan University.

Downloaded from: <https://e-space.mmu.ac.uk/622357/>

Usage rights:  Creative Commons: Attribution-Noncommercial-No Derivative Works 4.0

**Enquiries:**

If you have questions about this document, contact [openresearch@mmu.ac.uk](mailto:openresearch@mmu.ac.uk). Please include the URL of the record in e-space. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from <https://www.mmu.ac.uk/library/using-the-library/policies-and-guidelines>)

# Markerless Facial Motion Capture: Deep learning approaches on RGBD data

by

Connah Kendrick

A thesis submitted in partial fulfilment of the  
Manchester Metropolitan University for the degree of  
Doctor of Philosophy

Faculty of Science and Engineering  
School of Computing, Mathematics and Digital Technology

**MANCHESTER METROPOLITAN UNIVERSITY**

November 2018

# *Abstract*

Facial expressions are a series of fast, complex and interconnected movement that causes an array of deformations, such as stretching, compressing and folding of the skin. Identifying expression is a natural process in human vision, but due to the diversity of faces, it has many challenges for computer vision. Research in markerless facial motion capture using single Red Green Blue (RGB) camera has gained popularity due to the wide access of the data, such as from mobile phones. The motivation behind this work is much of the existing work attempts to infer the 3-Dimensional (3D) data from 2-Dimensional (2D) images, such as in motion capture multiple 2D cameras are calibration to allow some depth prediction. Whereas, the inclusion of Red Green Blue Depth (RGBD) sensors that give ground truth depth data could gain a better understanding of the human face and how expressions are visualised.

The aim of this thesis is to investigate and develop novel methods of markerless facial motion capture, where the focus is on the inclusions of RGBD data to provide 3D data. The contributions are: A tool to aid in the annotation of 3D facial landmarks; A novel neural network that demonstrate the ability of predicting 2D and 3D landmarks by merging RGBD data; Working application that demonstrates complex deep learning network on portable handheld devices; A review of existing methods of denoising fine detail in depth maps using neural networks; A network for the complete analysis of facial landmarks and expressions in 3D.

The 3D annotator was developed to overcome the issues of relying on existing 3D modelling software, which made feature identification difficult. The technique of predicting 2D and 3D with auxiliary information, allowed high accuracy 3D landmarking, without the need for full model generation. Also, it outperformed other recent techniques of landmarking. The networks running on the handheld devices show as a proof of concept that even without much optimisation, a complex task can be performed in near real-time. Denoising Time of Flight (ToF) depth maps, showed much more complexity than the tradition RGB denoising, where we reviewed and applied an array of techniques to the task. The full facial analysis showed that when neural networks perform on a wide range of related task for auxiliary information allow for deep understanding of the overall task.

The research for facial processing is vast, but still with many new problems and challenges to face and improve upon. While RGB cameras are used widely, we see the inclusion of high accuracy and cost-effective depth sensing device available. The new devices allow better understanding of facial features and expression. By using and merging RGB data, the area of facial landmarking, and expression intensity recognition can be improved.

# *Acknowledgements*

Throughout my research and preparation of this thesis, many people have guided my understanding and knowledge that allowed me to form the work presented. Firstly, I would like to thank my Director of Studies, Dr. Kevin Tan, for his understanding, support and overall confidence in my abilities from the very beginning. I would also like to express my utmost gratitude to my other supervisor Dr. Moi Hoon Yap, for her kindness, patience, guidance and continuous support through talks and opportunities. I also want to express my utmost gratitude to Image Metrics, Dr. Kevin Walker and Dr. Tomos Williams who all provided their own unique perspective in different aspects of my work and contributing greatly when I was in need.

I have received help and advice, directly or indirectly, from my peers and colleagues throughout my study. These people motivated me to share ideas and build relationships to further expand my knowledge. From MMU, Dr. Daniel Leightley, Dr. Choon Ching Ng, Dr. Brett Hewitt, Dr. Ezak Fadzrin, Dr. Adrian Davision, Dr. Darren Dancey, Mr. Sean Maudsley Barton, Ms. Jhan Saad A Alarifi, Mr. Manu Goyal, Mr. Guido Ascenso and everyone who kindly agreed to participate in my data collection experiment to form the Kinect One Expressional Dataset ([KOED](#)) dataset.

I would like to thank the research administration roles Ms. Megan Schofield, Ms. Francesca Robinson and Ms. Kristina Ganchenko. As well as always being available to help with a multitude of requests, they were key to helping with the success and progression of this PhD. Also, for there support and success with allowing me to help with the Science and Engineering Research Symposium.

I would like to thank my friends From both Leigh and Manchester for always being there whenever I needed advice through the high and low points. I would also like to thank the pool team i am part of for there continued support and patience throughout the studies.

---

My most sincere thanks goes to my family, for always believing I could succeed, even when I did not. To my Grandmother, Ms. Brenda Richardson, who have always been available if I need her, and always help me to think positively. To my Mother, Ms. Lisa Kendrick, I am unable to express how much I want to thank her for always listening and defending me. For always making me laugh, through the difficult times we both have had in the past.

I am grateful for MMU and Image Metrics for providing the studentship for me to complete this thesis, and provide valuable experience in teaching at a university level.

# Contents

<b>Abstract</b>	<b>i</b>
<b>Acknowledgements</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>ii</b>
<b>List of Figures</b>	<b>viii</b>
<b>List of Tables</b>	<b>xi</b>
<b>List of Abbreviations</b>	<b>xii</b>
<b>List of Publications</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 Motivation . . . . .	2
1.3 Problem Statement . . . . .	3
1.4 Aim and Objectives . . . . .	4
1.5 Contributions . . . . .	5
1.6 Thesis Organisation . . . . .	5
<b>2 Literature Review</b>	<b>8</b>
2.1 Introduction . . . . .	8
2.2 Facial Structure . . . . .	9
2.3 Emotion Representation . . . . .	10
2.4 Facial Expression Analysis and Landmarking . . . . .	12
2.4.1 Facial Expression Analysis and Intensity Representation . . . . .	13
2.4.2 Face Localisation and Pre-Processing . . . . .	14
2.4.3 Facial Data Extraction and Representation . . . . .	15
2.5 Landmarking methods . . . . .	16
2.6 Expression Recognition . . . . .	17
2.7 Multi-Output Neural Networks . . . . .	18
2.8 Machine Learning . . . . .	19
2.9 Deep Learning . . . . .	19
2.10 Facial Animation set-up . . . . .	20

---

2.10.1	Model Generation . . . . .	21
2.10.2	Rigging/ Parametrization . . . . .	22
2.10.3	Blendshapes . . . . .	22
2.10.4	Animation and Facial Animation Units (FAU) Generation . . . . .	23
2.11	Depth Sensing . . . . .	24
2.11.1	Structured Light . . . . .	25
2.11.2	Time of Flight . . . . .	26
2.12	Depth based facial landmarking and animation . . . . .	27
2.13	Research Direction . . . . .	28
2.14	Summary . . . . .	29
<b>3</b>	<b>Theories and Techniques</b>	<b>30</b>
3.1	Introduction . . . . .	30
3.2	Camera Settings . . . . .	31
3.2.1	3D Viewing Frustum . . . . .	31
3.2.2	Alternative Viewing Spaces and Metrics of Multi-Sensory Devices . . . . .	32
3.3	Face Processing . . . . .	33
3.3.1	Face Detection . . . . .	33
3.3.2	Landmarking . . . . .	34
3.3.3	Denoising . . . . .	35
3.3.4	Automated Processing . . . . .	35
3.3.5	Depth Processing and Normalisation . . . . .	36
3.4	Deep Learning . . . . .	37
3.4.1	Background . . . . .	38
3.4.1.1	Convolutions . . . . .	39
3.4.1.2	Activation Functions . . . . .	40
3.4.1.3	Pooling . . . . .	41
3.4.1.4	Fully Connected Layers . . . . .	42
3.4.1.5	Output . . . . .	42
3.4.2	Loss . . . . .	42
3.4.3	Optimiser . . . . .	44
3.5	Performance Metrics . . . . .	46
3.5.1	Loss Functions . . . . .	47
3.5.2	Statistical Analysis . . . . .	49
3.5.2.1	T-Test . . . . .	49
3.5.2.2	Standard Deviation . . . . .	51
3.5.2.3	Confusion Matrix . . . . .	52
3.5.3	Distance Metrics . . . . .	53
3.6	Summary . . . . .	54
<b>4</b>	<b>Preliminary Research</b>	<b>55</b>
4.1	Introduction . . . . .	55
4.2	3D Data Annotation . . . . .	56

---

4.2.1	Related Work . . . . .	57
4.2.2	Proposed Method . . . . .	60
4.2.3	Results . . . . .	67
4.2.4	Discussion . . . . .	73
4.3	Kinect One Expression Dataset (KOED) . . . . .	74
4.3.1	Experimental Protocol . . . . .	74
4.3.1.1	Emotional Replication Training . . . . .	75
4.3.1.2	Ethics . . . . .	75
4.3.1.3	Equipment and Experimental Set-Up . . . . .	75
4.3.2	Dataset Ground Truth Acquisition . . . . .	76
4.3.3	Dataset Analysis . . . . .	78
4.3.3.1	Demographic Breakdown . . . . .	78
4.3.3.2	Comparison with Current Datasets . . . . .	78
4.4	Summary . . . . .	81
<b>5</b>	<b>Real-Time Facial Landmarking using Neural Networks</b>	<b>82</b>
5.1	Introduction . . . . .	82
5.2	Related Works . . . . .	84
5.2.1	Facial Landmarking with Neural Networks . . . . .	85
5.2.2	Neural Network Compression . . . . .	87
5.2.3	Merging Visual and Depth . . . . .	89
5.3	Proposed Methods . . . . .	90
5.3.1	Facial Landmarking for Mobile Applications . . . . .	90
5.3.2	3D landmarking with auxiliary information . . . . .	95
5.4	Results . . . . .	97
5.5	Discussion . . . . .	110
5.6	Summary . . . . .	113
<b>6</b>	<b>Depth Data Denoising</b>	<b>114</b>
6.1	Introduction . . . . .	114
6.2	Related work . . . . .	116
6.3	Proposed Methods . . . . .	121
6.3.1	Proposed Data Synthesizing method . . . . .	121
6.3.2	Generation of noise upon depth images . . . . .	124
6.3.3	Experimental Setup . . . . .	125
6.4	Results . . . . .	127
6.5	Discussion . . . . .	132
6.6	Summary . . . . .	133
<b>7</b>	<b>Multi-Tasking Neural Network for facial landmarking</b>	<b>135</b>
7.1	Introduction . . . . .	135
7.2	Related Works . . . . .	137
7.3	Proposed Method . . . . .	138
7.4	Results . . . . .	140
7.5	Discussion . . . . .	146



---

7.6	Summary	146
<b>8</b>	<b>Conclusion</b>	<b>148</b>
8.1	Introduction	148
8.2	Research Findings	149
8.3	Future Work	151
8.3.1	Neural Network with Object Detection Integration	152
8.3.2	General Adversarial Networks based networks	152
8.3.3	Dataset Expansion and Improvement	153
8.3.4	Denoising Expansion	153
8.3.5	Integration of Improved Depth Sensing Technology	154
8.4	Concluding Remarks	155
<b>A</b>	<b>Comparison of 3D landmarks</b>	<b>156</b>
<b>B</b>	<b>Expression Network Activation</b>	<b>158</b>

# List of Figures

1.1	A visual guide to the structure and flow of the thesis. . . . .	6
2.1	A visualisation of the seven universal emotions. . . . .	11
2.2	Example of an Autodesk model with a library Blendshapes (Shape-points in Blender) on the right. . . . .	24
2.3	Example of an image used by a structured light projector, some sensors employ variations with different colours or specular patterns to increase accuracy. . . . .	25
2.4	Example of the image from Fig 2.3 projected into a 3D scene, notice how the straight lines curve, change direction and thickness when interacting with objects in the scene. . . . .	25
3.1	An example of a 3D viewing frustum. . . . .	31
3.2	Illustration of different viewing spaces from the Kinect depth sensor. . . . .	33
3.3	This illustrates how changing the input $x$ to function $f$ , affects the output value, creating a ‘u’ shape. . . . .	39
3.4	An illustration of different filters that can be used to highlight features. . . . .	40
3.5	This image shows Rectified Linear Unit (ReLU) (left) activation vs sigmoid (right), notice how sigmoid normalises the range, but ReLU allows an infinite range, but without negative values. . . . .	41
3.6	An example of a max-pooling window the network focus on the highest values. . . . .	42
3.7	This image shows how changing the loss function affects the result of image de-noising. Even though the network is trained to optimise Mean Squared Error (MSE), it still achieves a better Mean Average Error (MAE) score than the MAE based network. The network was trained on the Morph dataset. . . . .	43
3.8	This image shows the effect of the denoisers when trained on different losses. . . . .	44
3.9	An illustration of a deep learning gradient; the loss can start at any point along the line. Blue points show local minimums, in which the network can appear to have converged but have not fully. The green point is the global minimum at which we want our network to reach. Red points indicate local maximums, which we want avoid. . . . .	45
4.1	Example of two face models of the same individual taken from [1]. The neutral face (left) and the eyebrows raised (right). . . . .	57

---

4.2	The same models as Fig. 4.1, but with the vertex locations displayed.	58
4.3	An example of the Yaw bar calculation.	64
4.4	A comparison of the number of iterations.	66
4.5	The total Maya times (left) compared to the proposed method (right)	71
4.7	The points placed back on the model similar to Fig. 4.6, but aligned with an image.	72
4.6	The results of the current tests the green points are the proposed annotator and red points, Maya.	72
5.1	A visualisation of the basic network used for porting to mobile.	93
5.2	A visualisation of the basic network used for the auxiliary information experiment. The networks was adapted from 5.1, having fewer pooling layer to preserve the data.	95
5.3	A visualisation of the merged network used for this experiment	96
5.4	The networks accuracy, both training and validation.	99
5.5	The networks loss, both training and validation.	100
5.6	The networks MAE, both training and validation.	100
5.7	Example output of the android application	101
5.8	The MSE of the UV Only networks validation over 100 epochs.	103
5.9	The MSE of the XYZ Only networks validation over 100 epochs.	104
5.10	The MSE of the UVXYZ networks validation over 100 epochs.	105
5.11	A visual comparison of the results from the 2D and 3D networks.	106
5.12	A visual comparison of the output of the final convolutional filter for each type of network prediction on the RGB Images. The third column illustrates the feature maps for UVXYZ prediction, the best performance with auxiliary information.	109
5.13	The result of the Greyscale ( $G_s$ ) UVXYZ trained network and the appropriate model from the same input depth map. The model is transparent to show the geometry coordinates of the facial landmarks.	110
5.14	Example output of the android application in a controlled environment(bottom) and wild environment (top), with both male and female faces.	111
5.15	Same Scenario as Fig 5.14, but with glasses.	111
6.1	A comparison of different noise generation methods applied to the depth data and then render into models.	115
6.2	An example of an integrated model (left) and a single frame model (right).	116
6.3	A comparison of the different values of sigma for Block Matching 3D (BM3D). Traditionally sigma is increased with noise, but for depth data, we found reducing sigma increases performance.	117
6.4	The 3D viewing frustum (perspective mode) used in many 3D applications.	124

---

6.5	An example of a clean (ideal) model versus a noisy model. The image illustrates the type of noise we must simulate, such as the wave type effect over the skin and the self-occlusion noise of the positional difference between the infra-red (IR) transmitter and receiver. . . .	126
6.6	This figure represents a visual comparison of the denoised 3D models using normalised data. The clean shows the ground truth, BM3D is state of the art. BaseNet1 and BaseNet32 are our proposed methods; VAE1 and VAE32 are the networks based on Variational Auto Encoder (VAE). . . . .	128
6.7	A visual comparison of the denoised 3D models using raw depth data. The clean show the ground truth model; BaseNet1 and BaseNet32 are our proposed methods; VAE1 and VAE32 are the networks based on VAE. . . . .	128
6.8	The Peak Signal to Noise Ratio (PSNR) Loss of networks the trained networks. . . . .	129
6.9	The MAE Loss of networks the trained networks. . . . .	130
6.10	The MSE Loss of networks the trained networks. . . . .	131
6.11	The Loss of the depth networks. . . . .	131
6.12	The ground truth clean (left), the noisy input (middle) and the network result (right). . . . .	132
6.13	The Loss of the depth networks. . . . .	132
7.1	A visualisation of the network used for this experiment . . . . .	139
7.2	The MSE and MAE loss for the 2D landmark output . . . . .	141
7.3	The MSE and MAE loss for the 3D landmark output . . . . .	141
7.4	The MSE and MAE loss for the gender output . . . . .	142
7.5	The MSE and MAE loss for the expression output . . . . .	142
7.6	The MSE and MAE loss for the combined outputs . . . . .	143
7.7	Outputs of the final convolution layer . . . . .	144
7.8	A comparison of a definite male and female prediction. . . . .	145
7.9	A comparison of definite expression predictions. This figure illustrates a subsection of the graph, a full version is available in the Appendix B Fig B.1 . . . . .	145
A.1	The MSE scores of the network during training . . . . .	156
A.2	The MAE scores of the network during training . . . . .	157
B.1	The full graph of the expression networks activation . . . . .	158

# List of Tables

4.1	Table of $t$ -test results comparing the facial landmark accuracy. . . .	68
4.2	Table of the overall distance in millimetres. . . . .	69
4.3	Table of the $t$ -test results for the time comparison. . . . .	70
4.4	Table of the proposed methods average alignment and annotation times (minutes) with Standard Deviation (SD). . . . .	71
4.5	Table of Mayas average alignment times with SD. . . . .	71
4.7	Table of the overall performance . . . . .	77
4.6	Truth table of the Dlib face detection . . . . .	77
4.8	Table showing break down of overall participants . . . . .	78
4.9	Table Comparing available datasets . . . . .	81
5.1	A comparison of the accuracy and loss for three types of network. . .	92
5.2	Table of the result file sizes . . . . .	98
5.3	Table of the testing set evaluations on MSE. . . . .	107
5.4	Table of the testing set evaluations on MAE. . . . .	108
6.1	The results based on raw depth data. The mean and SD are obtained for PSNR and Time. The model indicates if the method can generate a 3D model. . . . .	127
6.2	The results based on normalised data. The mean and SD are obtained for PSNR and Time. The Model indicates if the method can generate a 3D model. . . . .	127
7.1	The results of the network on the testing set. . . . .	144
8.1	Research objectives and outcomes. . . . .	150
A.1	A comparison of the test score of the networks and ours. . . . .	157

# List of Abbreviations

**2D** 2-Dimensional

**3D** 3-Dimensional

**3DMM** 3D Morphable Model

**AAM** Action Appearance Model

**ASM** Active Shape Model

**FAU** Facial Animation Units

**API** Application Program Interface

**APK** Android Package

**BM3D** Block Matching 3D

**CNN** Convolutional Neural Networks

**CLM** Constrained Local Model

**CSV** Comma Separated Value

**CPU** Central Processing Unit

**DCNN** Deconvolution Neural Network

**DDnSRCNN** Deep Denoising Super Resolution Convolutional Neural Network

**DDE** Displaced Dynamic Expression

**DEM** Dynamic Expression Model

---

**AU** Action Unit

**FACS** Facial Action Coding System

**FRGC** Facial Recognition Grand Challenge

**FPS** Frames Per Second

**FOV** Field of View

**GAN** General Adversarial Networks

**GPU** Graphical Processing Unit

**G<sub>s</sub>** Greyscale

**G<sub>s</sub>D** Greyscale Depth

**HMD** Head Mounted Display

**RAM** Random Access Memory

**RGB** Red Green Blue

**RGBA** Red Green Blue Alpha

**RGBD** Red Green Blue Depth

**RNN** Recurrent Neural Network

**ReLU** Rectified Linear Unit

**SD** Standard Deviation

**ICP** Iterative Close Point

**ToF** Time of Flight

**HOG** Histogram of Orientated Gradients

**IR** infra-red

**KOED** Kinect One Expressional Dataset

---

**LBP** Local Binary Pattern

**LSTM** Long Short-Term Memory

**MSE** Mean Squared Error

**MAE** Mean Average Error

**MLP** Multi-Layered Perceptron's

**PCA** Principal Component Analysis

**PDM** Point Distribution Model

**PSNR** Peak Signal to Noise Ratio

**SVM** Support Vector Machine

**VAE** Variational Auto Encoder



# List of Publications

This thesis is based on material from the following publications:

1. Connah Kendrick, Kevin Tan, Tomos Williams, Moi Hoon Yap (2017), “An Online Tool for the Annotation of 3D Models”, The 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), pp. 362-269, IEEE.
2. Connah Kendrick, Kevin Tan, Kevin Walker, Moi Hoon Yap (2018), “The Application of Neural Networks for Facial Landmarking on Mobile Devices”, Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4, ISBN 978-989-758-290-5, pages 189-197. DOI:10.5220/0006623101890197.
3. Connah Kendrick, Kevin Tan, Kevin Walker, Moi Hoon Yap (2018), “Towards Real-Time Facial Landmark Detection in Depth Data Using Auxiliary Information ”, *Symmetry*, 10(6):230.

# Chapter 1

## Introduction

*This chapter introduces the concepts, techniques and theories of motion capture and animation. The terminology is shown in the background alongside the problem statement, thesis contributions and thesis structure.*

### 1.1 Background

Facial motion capture is a well-established field with many commercial applications/software available [2, 3]. Furthermore, facial animations a widely researched field [4, 5, 6] with many techniques used to track the face and determine the expression. At its core facial motion capture relies on the work performed in image processing to track points on the face, whether its optical markers or marker-less. The motion capture is used to aid animators it creating realistic motion on a 3D model. Many of the modern facial capture systems only implement 2D techniques and attempt to infer both the 3D points and a 3D representation of the actor's face. However, depth data can provide a complete view of the movements, such as how the wrinkle crease the skin and changes in depth like pouting. The research focus of this work is to incorporate depth data into the facial motion capture pipeline.

---

## 1.2 Motivation

When the Kinect version 1 for the Xbox 360 was released, it signalled the beginning of readily available low-cost sensors and caused massive interest in 3D from both publicly and commercially. During this time there was a drastic increase in research on how 3D data could improve many fields of research. However, devices, such as the Kinect were designed for full-body tracking and show to improve research in these areas, but in other areas the Kinect, such as facial identification and expression recognition, the methods used were slow and hampered by noise. Because of the limitations of the Kinect, research in this area is limited. However, the technology was still in its infancy, and newer depth sensing capture both higher resolution depth maps with a fraction of the noise. With the technological improvements, and with improved data merging methods could have a beneficial impact on for facial processing.

A method that is capable of retrieve structural information in real-time would provide aid in creating and tracking facial movements. Unfortunately, there are many current restrictions, such as datasets are limited in the motions they are targeting, and the complexity of faces increased when viewing 3D structure, in comparison to 2D. The task of 3D landmarking has restrictions as many of the high accuracy scanners cannot function in real-time, and real-time sensor suffers from a significant degree of noise.

Many methods try to incorporate RGBD prefer to use the depth data only for model construction, making the depth data a drain on resources after model construction. Additionally, as they only use RGB for landmarking, the movements are restricted, due to RGB only being able to identify landmarks in highly detailed facial areas.

---

## 1.3 Problem Statement

Currently the use of consumer based depth sensors focus on capture the 3D positions of large movements, such as limbs where it has made significant improvements to how we approach analyse of human movement. However, for facial analysis the devices are used significantly less, due to the large degree of noise that covers the facial features. We perform our work to analyse the capability of low cost depth sensors, to perform the acquisition of 3D facial landmarks.

Outside of commercial production companies, the use of 3D technologies is uncommon, and when coupled with the costly equipment and difficultly set-up small production or individual consumers cannot fully access the improved results. Additionally, recent technological improvements and innovations, such as the Kinect have the potential to overcome the issues of using multiple cameras to create stereography depth images. Furthermore, the existing methods use a complex pipeline to produce their results, normally the traditional machine learning is used for locating landmarks and assessing expressions.

Many of the problems come from the ability to retrieve 3D landmarks coordinates in real-time when considering both, low powered consumer devices and the level of noise found in depth maps. Further, the amount of datasets is limited as many prefer the use of traditional RGB data, where 3D landmarks are not accessible.

The logical progression of developing and improving methods of detecting landmarks for expression recognition is to extend the work from the 2D viewpoint and statistical facial models for 3D shape prediction, to the inclusion of accurate depth data that can generate a near-identical 3D model of the users face from single frame depth maps. Unfortunately, this is not a simple task. Traditional machine learning techniques focus on small set of 3D landmarks, and the majority of the focus is using a predefined 3D model or curvature analysis, which is a processor intensive task. An additional difficulty comes from the diversity in faces when moving into 3D, where the differences in a facial structure become more

---

apparent and defined. Neural networks provide a way of quickly accessing facial images that can efficiently process both **RGB** and depth images simultaneously. Also, to retrieve landmarks in **2D** and **3D**.

This thesis addresses these limitations and proposes some new methods to build the research gap. Firstly, we need to know the ground truth **3D** points are accurate before we begin training or the systems will not be adequate. We then demonstrate the use of neural networks on portable mobile devices, to illustrate neural networks application to consumers. We validate our choice of the stream, to ensure the most effective and efficient network is implemented into our project. As noise can be an issue with depth maps, we evaluate methods of real-time denoising.

## 1.4 Aim and Objectives

The primary aim of this research is to propose markerless facial motion capture methods using deep learning approaches on **RGBD** data. The improvements will aid current computer systems to gain a thorough knowledge of the face and its **3D** structure. The following objectives have been designed to achieve the aim:

- Identify the research gap and create resources with ground truth for validation.
- Propose new methods to detect **2D** and **3D** facial landmarks in real-time, with a demonstration on portable devices.
- Investigate the ability of current **RGB** image denoising techniques and design Convolutional Neural Networks (**CNN**)s for depth data denoising.
- Improve the performance of facial landmarking for accurate expression intensity detection.

---

## 1.5 Contributions

The main contributions of this thesis are as follows:

- A novel techniques of aligning 3D geometry to a 2D reference image without calibration data, for 3D model data annotation. A new dataset, namely KOED that contains synchronous RGB and depth data.
- Novel neural networks for the combination of RGB and depth data for real-time 3D facial landmarking using auxiliary information, with a competitive analysis of stream impact on network performance. We also demonstrate real-time mobile based neural networks performing typically computationally expensive tasks in real-time.
- An evaluation of existing denoising neural networks re-targeted for the use of depth data, with a solution to the lack of ground truth, not noisy, ToF depth data.
- A neural network for facial landmarking and the estimation of expression intensity.

## 1.6 Thesis Organisation

Fig. 1.1 illustrates the structure of the thesis, where there are three main segments:

1. Chapters 1 - 3: Introductory chapters describe the topic of the thesis, the existing work done and any terminology used.
2. Chapters 4 - 7: Contribution chapters explain each of the contributions made in the thesis, their underlying novelty and how they work.
3. Chapter 8: Conclusion chapter of the thesis, this will provide a summary of the thesis and outline future work.

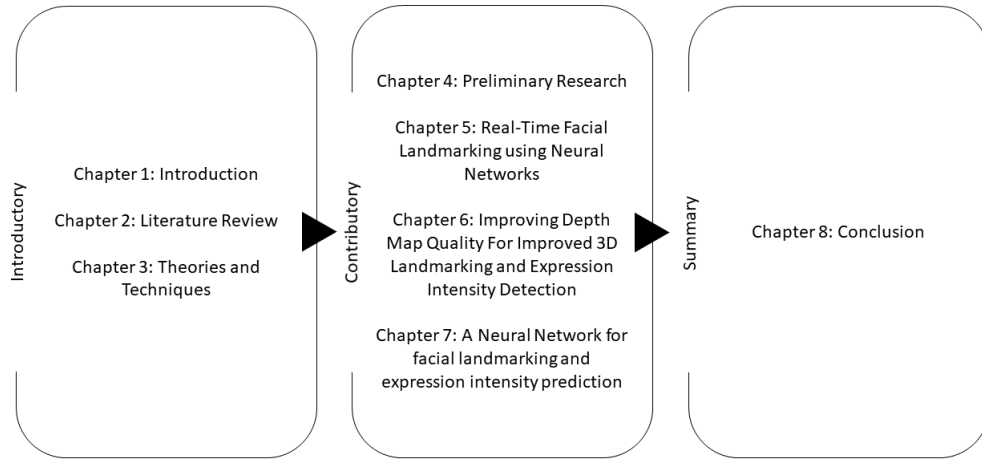


FIGURE 1.1: A visual guide to the structure and flow of the thesis.

A brief overview of each chapter follows:

Chapter 1: The current chapter is a guideline to the contents of the thesis providing the background information of the problem statement, the motivation to solve the issues and the contributions made to resolve the problems faced.

Chapter 2 presents the fundamental knowledge and provides a review of the current papers in facial motion capture and animation. As this subject falls into multiple categories, information on these subjects is provided.

Chapter 3 provides the technical knowledge required for understanding [3D](#) motion capture and animation techniques, including the computer vision methods used to process input video data.

Chapter 4 introduces a developed tool to aid with the annotation of [3D](#) models with an evaluation of its performance with the traditional methods. Additionally, we introduce a new Kinect based dataset.

---

Chapter 5 reviews the effectiveness of a neural network is at merging RGB and depth data for facial landmarking. We validate the method on our a Kinect One based facial animation dataset. The use of the neural network on consumer devices is demonstrated.

Chapter 6 presents an evaluation of methods capable of real-time denoising of Kinect based ToF depth maps. We overcome the issue of ground truth data by generating a synthetic depth dataset.

Chapter 7 uses the knowledge gained from the previous chapters to design and implement an all-in-one neural network that focuses on improved facial landmarking and facial expressions intensity prediction.

Chapter 8 concludes the thesis contributions and provide a summary of future work.



# Chapter 2

## Literature Review

*This chapter introduces and examines existing methods and techniques available in this field of study.*

### 2.1 Introduction

In this chapter, an overview of the current literature in facial analysis and the different aspects required for facial landmarking. There is a large amount of research into facial animation and its different aspects, such as detection, landmarking and expression identification. Although using human motion for animation purposes date back to 1917 [7], a method for facial motion capture only introduced in 1988 [8], which employed a helmet device that would be attached to a section of the face and use potentiometers to track its motion. A more complete and detailed view and history of motion capture can be found in [9, 10].

Since the early days of motion capture, the methods have improved significantly, and with technological advancements in a mobile device, the ability for anyone to try motion capture is capable. However, the methodology is still sophisticated and requires multiple sections to work together to create a smooth experience just for standard 2D tracking. The focus of tracking movements differs from other fields as the focus is on gaining an understanding of the facial

---

movements and representing the expression on a 3D models. The current focus on consumer-based programs is to work with 2D data, where we aim to extend the capabilities of methods to 3D.

There is a limited number of datasets in this domain that annotate both landmarks, other than the five common points, and facial expression. The lack of available data has kept research in its infancy as many research use their datasets or have to merge with different datasets to produce their results. The problem of data deficiency is increase when trying to perform 3D facial landmarking and expression recognition.

This section will give an outline of the current chapter. Firstly, we begin with an overview of the face, its structure and facial expressions. Followed by the categorisation of expressions objectively. After presenting a method of categorising expression, we break down the current pipeline for processing image for animating a virtual avatar. Furthermore, this introduces how we capture the motions with the process of traditional machine learning which many of the current techniques implement. Finally, the chapter leads to the current use of depth-sensing devices for facial animation.

## 2.2 Facial Structure

The face constructed of three base layers: the skull, the muscles and the skin. Attached to the skull is a series of muscles, the muscle can contract and relax to change their shape which directly affects the skin, a highly deformable surface. The face contains multiple muscles that control many different aspects of facial movements, such as separate muscles for either raising or lowering the lip corners. Expressions performed by activating or relaxing one or more muscles in tandem. The final layer is the skin, which is a highly deformable layer it covers the head region and links to the muscle on the face, when muscle contract they stretch and crease the skin, creating the appearance of expressions.

---

Facial muscles are a series of fast-moving fibres that can move in less than 20 milliseconds, which includes the time from the central nervous system sending the signal [11]. The speed of expressions means a natural expression can change in rapid succession with ease, without a human observer noticing. However, these subtle expressions can add detail and realism to the characters.

The human visual cortex is a vital part of understanding how we interoperate expressions. The human eye processes on around 10-12 images a second, with light pooling lasting around 15 milliseconds. The speed of the processing of images and light pooling indicates movement at 1/25th a second is near the limit of human perception, which is one of the reasons many programs broadcast at 25 Frames Per Second (FPS).

## 2.3 Emotion Representation

Expression recognition has many different techniques available that provide high accuracy results. However, the question of how expression varies among different people through environmental and cultural influence is vital, as if different cultures recognise expressions differently the method must be adjusted for different places. The universality of emotion was a commonly debated study from researchers, such as Darwin [12] who first examined expression in humans and animals that were universal. Later, the work was further expanded by Tomkins et al. [13] in which he determined there are eight easily recognisable expressions. However, certain gestures can have different meanings in different places around the world.

Ekman et al. [14] performed a series of tests around the world, including tribes outside of modern influence. During the tests, they asked the tribes to perform a series of expressions that relate to certain emotions. They found that the people of the tribe performed certain expressions the same as people from all around the world. These expressions are known as universal emotions:

- *Surprise*: A sudden or unexpected event

- 
- *Fear*: The threat of harm
  - *Anger*: Interference with goals
  - *Disgust*: Offensive in nature
  - *Contempt*: Immoral action
  - *Sadness*: Loss of a valued object or person
  - *Happiness*: Pleasure

Figure 2.1 illustrates the neutral face and seven universal emotions defined by Ekman et al. [15]. They also describe how emotions are caused by a two staged method:

- *Trigger*: this is the person, object or event that causes the emotion.
- *Impulse*: This is an uncontrollable event which can induce one of the seven universal emotions.

When an impulse occurs, facial muscle moves in tandem to perform the expression. However, after the initial movement, people may attempt to conceal their emotions. Although the trigger may be identical, such as a dog running towards someone, the impulse may be different people with dogs, may become happy, but those afraid may show fear instead.

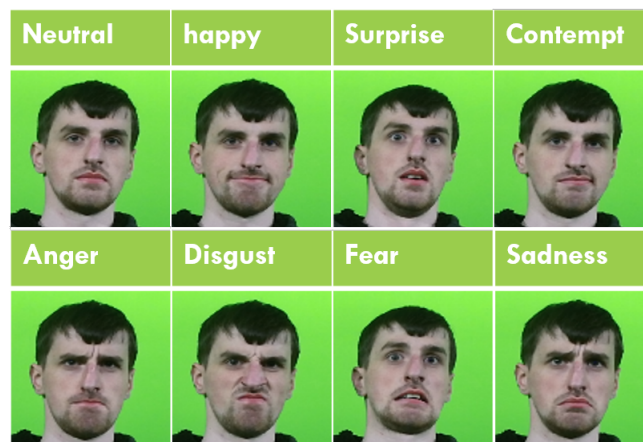


FIGURE 2.1: A visualisation of the seven universal emotions.

---

After showing the universality of expression, Ekman et al. [16] continued to produce a method of both annotating facial movement and intensity, namely Facial Action Coding System (FACS). FACS is a method of objectively classifying any humanly possible facial movement and the intensity of the movement. FACS also provide no emotional attachment to movements, just how the move directly affects facial appearance.

Each separate muscle movement can be categorised into a unique Action Unit (AU) [16], and because of the universality of expressions, each expression can be broken down into a set of AU. The method is capable of recording human motion accurately even with the variations between individuals, such as wrinkle, fat and bone structure because the underlying muscles remain similar. FACS has firm guidelines on how to categories movements and how to interoperate multiple facial muscles moving at the same time. Also, to score FACS movement, the intensity is given a value between A-E, where A is a minor activation and E is the highest intensity possible.

There are many challenges and limitations in the current work. Many researchers focus on the realistic reconstruction of faces, including unique features, such as wrinkles, which are difficult for consumer based sensors due to noise and with monocular cameras. Furthermore, this extends to creating real-time methods that incorporate 3D data. Many of the methods focus on the refinement of models over the tracking of the faces, preferring either a 2D tracking, which lacks information but is quick. In contrast a full 3D model approach, that has more information but is slower and hampered by noise.

## 2.4 Facial Expression Analysis and Landmarking

The human face is exceptionally complex providing many deformations that change the appearance of the face. The changes in appearance give insight into the emotions individuals are experiencing, but many of the existing automatic techniques did not include intensity as a measure [17]. In production, the implementation

---

of facial landmarking to allow facial features and movement tracking with high precision. Furthermore, they realistically represent motion by intensity calculation from the expression peak to the current allowing realistic change of facial expressions.

### 2.4.1 Facial Expression Analysis and Intensity Representation

Facial expression analysis is a widely researched area with many available techniques for estimating facial expressions using computer vision. The areas of research include expression classification, human-computer interaction and virtual avatars. For an in-depth evaluation, the reader is advised to read [18, 19].

The pipeline for facial analysis divides into three key segments:

- *Face Detection and Pre-Processing*: This is the processes of face localisation in an image. The face can then be cropped out of the image and resized for further processing if required.
- *Feature Extraction*: Traditionally is performed into two main ways, such as appearance-based with Principal Component Analysis (PCA) or Histogram of Orientated Gradients (HOG) and feature-based using canny edge detection or Sobel. For problems such as facial recognition [20], the use of appearance-based methodology is more common, but for landmarking, implementation of feature-based methods target specific facial features. Features can be represented differently in images, such as edges, points and blobs, but commonly referred to as “interesting” sections of an image, as they provide key details of the face. In newer methods that employ deep learning, the use of machine defined features are implemented. An algorithm decides machines defined features at the time of training; this means it could use shape features. Shape features, look at whole shape, such as the lips and chin, these give a fuller view of the face and the expressions being performed.

- 
- *Processing Output*: The final stage the estimation of both the landmarks and performed expression.

The above three steps are performed in many of the current state of the art methods [21, 18]. Although, the methods can adjust and add steps to improve their performance. The focus on the output is a single task, such as landmarks or expression, but extensions can be made to suit multiple outputs.

## 2.4.2 Face Localisation and Pre-Processing

The first stage of an algorithm is to localise the face and crop out the face. Viola and Jones [22] developed a robust object detection algorithm that runs in real-time. They used haar features as a core section of its processing, which is a series of sliding windows that perform a series of feature selection methods. The haar sequence was improved by Lienhart et al. [23] that allows for an increased amount of identified features. However, like all object detectors, the effectiveness of the method depends on the training data. Many face detectors train on frontal face detection only, thus cannot successfully detect heads at extreme angles.

Once the face is detected, post-processing on the facial image to collect features can begin. Post-processing usually includes cropping the face and resizing to suit the next stage of processing. However, other methods of post-processing can include, converting to  $G_s$  and using image processing techniques, such as Sobel or tone mapping. Another technique includes the use of affine transformations [24] to align facial geometry. However, this requires accurate landmarking as if the landmarks position is incorrect the alignment will not be correct; the method also struggles if the geometry is hidden, preventing face alignment at extreme angles.

Another popular method for aligning images but extends to 3D geometry is provided through the use of Iterative Close Point (ICP) [25], which over a number of iterations refining the position closer to the target. ICP has rigid, does not deform the target, and non-rigid versions which allow for precision translation of 2D and 3D geometry. Whereas affine transformation requires pre-existing correlation,

---

[ICP](#) attempts to predict the correlation automatically; the method can work well, even with noise data but can suffer from unnatural results, where it fails to match correct sections of the face.

Other methods for aligning face can integrate unsupervised method such as 3D Morphable Model ([3DMM](#)) [26]. [3DMM](#) is a statistical shape method trained on both scanned [3D](#) model and reference images; this builds a statistical model of faces, by implementing [PCA](#), appearance-based features are learned. By providing the trained [PCA](#) model with a new image, the [PCA](#) model can generate [3D](#) models that resemble a user unknown to the training data. [3DMM](#) has also been expanded over the years to include expressions and faces large rotations to the camera.

### 2.4.3 Facial Data Extraction and Representation

As mentioned, there are two types of image data extraction methods, feature-based and appearance-based. The feature-based method focuses on localising key “interesting” section of an image, this highlight facial features, such as the nose and eyes. Whereas, appearance-based focus on the unique features of each face, should as the wrinkles, colour and texture. For our work, we focus on both landmarking which is a feature based method and the facial shape.

In early works, the use of optical flow and base feature tracking can cause instabilities [27], due to inaccuracies in the landmarks, such as jittering. However, many of the newer works, such as Weise et al. [4] who uses a depth camera (Kinect 360) for facial animation and Cao et al. [28] who uses a single image to regress [3D](#) landmarks for animation, the use of facial shape regression is implemented [29]. Methods, integrating [PCA](#) [30] considers facial shape when predicting expression, while still using the popular methods, such as Action Appearance Model ([AAM](#)) [31] and Active Shape Model ([ASM](#)) [32] to predict facial landmarks, to segment the face for shape analysis.



---

## 2.5 Landmarking methods

In traditional methods, the use of machine learning and statistical models provide accurate landmarking for facial features. This section will break down the history of landmarking. The first breakthrough in automated facial landmarking was through a statistical model based method called [AAM](#). In which, could reliably detect facial landmarks through a variety of pose, both extreme and straightforward. [AAM](#) works by first fitting a deformable base model to a reference image, the base fitting to the image implements [PCA](#) and eigenvectors to represent the transformations. The base fit, will not be entirely accurate the goal of the algorithm is then to deform the initial fit to the object in the image. The algorithm performs the merging by repeating a series of warping function and feature extraction. [AAMs](#) required training data as they are based upon the techniques of Point Distribution Model ([PDM](#)) as part of the initial fit, but this allows adaptability for any number of landmarks and different objects. Other statistical model methods, such as Constrained Local Model ([CLM](#)) have been used to give high accuracy result in facial landmarking.

Traditional machine learning for landmarking facial features is a vast area of research. Kazemi et al. [33] showed a real-time high accuracy facial landmarking method that provides excellent results even on extreme facial poses. They used an ensemble of regression trees that use pair pixel intensity as feature values. By using a simple, feature the algorithm can achieve fast processing times but can become unstable under varying lighting conditions.

In recent years, the highest accuracy methods for facial landmarking integrate deep learning methodology. Early methods, such as Sun et al. [34], proposed a fully end-to-end [CNN](#) network that follows a standard network architecture of convolutions and max-pooling before outputs, which gave a high performance but only could predict five facial landmarks in clear areas of the face. Zhou et al. [35] and Liu et al. [36] used facial feature detectors to detect and crop facial features, such as eyes and nose, to feed into smaller neural networks. However, Zhou et al. [35] included a pre-processing stage to align the face regions. By using a series

---

of smaller neural networks, the network focus on the key attributes, instead of generalising to all the facial features. Lia et al. [37] implemented a much larger scale network that uses convolution and deconvolution to collect image features. The network uses a series of Long Short-Term Memory (LSTM) [38] layer to predict and refine landmark layers. By using deconvolution layers, they overcome previous issues of over-pooling the images, that shrinks the image too much of expression to be visible. However, they can only predict five facial landmarks.

## 2.6 Expression Recognition

Expression recognition is an extensive wide research area, and that splits into two separate categories. The first category is from the emotion perspective; the goal is to infer a person's emotion, such as happy or sad. This research is essential in fields, such as conversational agents or learning agents, where a system needs to engage with the user to work effectively. The second category is the objective measure of facial muscle movements by using FACS. The use of FACS has much more application and more commonly used in animation as separate facial movement can aid in transfers not just expressions, but personality traits.

Expression recognition, illustrated in much of the research performed currently [19, 39]. Wiese et al. [5] presented a method of expression animation based on depth data only. To do this, they built a series of expression models off-line and a rigid tracking mask. The rigid tracking mask is used to track the face through different positions and rotations using ICP. An actor specific PCA model of expressions is calculated, allowing for comparison of input frames for expression prediction. By using the expression model and performing PCA on input, frames allows accurate tracking of an actor expression. However, this required a high-resolution 3D scanner, whereas noisy low-resolution sensors produce less accurate results and the technique could only function at 15 FPS. Weise et al. [4] expanded their work by using less accurate consumer based scanners. Due to the less accurate depth data, colour image data is used to aid in expression prediction. These

---

methods perform expression based tracking by comparing pre-recorded footage of the person performing the expressions and then calculates which expression is the closest. Cao et al. [29] expanded on the work, by only using a 2D web camera, the method automatically detects, and furthermore it also allow users to adjust the 2D landmark positions. Their method then performs 3D shape regression, to predict the 3D facial model and landmarks.

## 2.7 Multi-Output Neural Networks

Many newer methods of expression recognition are based upon multiple outputs to overcome many difficult tasks at the same time, such as facial landmarking and facial direction [40]. Methods that perform multiple tasks may take longer to process however, since they are forced to predict additional relevant auxiliary information, during the training stage, the algorithm receives detailed feedback of error allowing in-depth learning of both the face and features. By having an improved knowledge of inputs, the accuracy of this method shows better results than the single focused networks [24]. For multiple outputs, we focus primarily on the use of deep learning. The work of using auxiliary information primarily came from Zhang et al. [40] where they investigated the effect of auxiliary information on network performance. To perform the experiment, they developed a series of small neural networks, the only difference being after convolutions the data would go to multiple fully connect layers depending on the output. The outputs consisted of both generally easy challenges, such as the five basic landmarks and gender, but also contain difficult tasks, face direction and age. The author's experiment showed that even when adding difficult tasks, by training to predict other relevant features, the network could learn significantly better, receiving improved accuracy. The authors expanded their method in [41]. Jourabloo et al. [42] experimented on this work with a focus on refining a 3D model. Ranjan et al. [43] produced an all-in-one network that can detect faces in images, and produce landmarks, gender and pose with high accuracy.

---

## 2.8 Machine Learning

Traditional machine learning is a method of getting a computer to produce a program that can replicate the desired output. However, it still requires humans in some aspects, such as determining the feature selection method. Features are interesting sections in an image, for example, expression detection a feature selector that provides insight into the lip shape is critical for certain expressions. Feature selection method can include, Local Binary Pattern ([LBP](#)), [HOG](#) and [PCA](#) these are known as human defined features as a person has designed and made the method they use to select features. Features can then be processed and shaped into a vector for processing by a machine.

After pre-processing the input images for feature selection, there are a variety of different machine learning techniques, in the aspect of face, Random forests [[44](#)] is typically used, as it allows a diverse set of features, accurate and high-speed results, as shown by Kezemi et al. [[33](#)], where they performed facial landmarking in a millisecond.

Traditional machine learning works as it is capable of determining many linear problems with high accuracy. Machine learning can quickly find correlations, even with difficult non-linear challenges using higher dimensional data. However, a limitation of machine learning is the reliance on human defined features and require the person's core understanding of the data.

## 2.9 Deep Learning

Deep learning with [CNNs](#) is a method of solving complex non-linear relationships between input data and output. While machine learning is an open box, deep learning networks are more of a black box as the whole network is trained rather than just the prediction section, such as Support Vector Machine ([SVM](#)) in machine learning.

---

Deep learning uses a neural network which is designed at the start and is vastly interchangeable, with researchers capable of creating a network structure to suit the problem faced. The first stage of CNNs is the use of convolutional layers, which act as the feature selection stage of a neural network. Convolutions act as sliding windows, performing basic matrix operation to decide the value of a single matrix cell, which is similar to a traditional kernel-based method, such as Sobel. After each convolution, the use of a linear activation unit normalises the values from convolutions. A neural network can optionally apply pooling which down-samples the feature maps. The convolutions and pooling can continue for many iterations through the network, as the network becomes deeper more in-depth and descriptive features are identifiable in images, but at the cost of increased processing requirements.

The final stage of a neural network is the fully connected layers or Multi-Layered Perceptron's (MLP). The MLP is a series of interconnected neurons, each neuron in one layer is connected to each neuron in the next, hence fully connect layer. The connection applies a weight the values in each neuron and uses the new value to determine the activation of the neuron. The output of the final layer is the network's prediction of results.

Although deep learning has shown vastly improved accuracy compared to traditional machine learning [45], the method is limited as it requires a vastly increased amount of training data. The increased data requirement is due to the full training requirements, such as learning the kernels for feature selection and ensuring the trained networks can work for a diverse amount of ethnicities.

## 2.10 Facial Animation set-up

In high-end commercial animation productions, the implementation of facial landmarking can track the expression and also the intensity of the movements [21, 46, 47]. Landmarks also allow the presence of multiple expressions or non-universal deformation. As a result, this research will target on facial landmarking [48, 49, 50].

---

For different methods of motion capture, facial models require unique preparation, such as blendshape library for shape interpolation, or the use of control points for model deformation. This section will discuss the available methods of generating 3D facial models and the techniques used to animate the models.

### 2.10.1 Model Generation

Generating a 3D model of a user's face is a difficult task, as for animations the model should be anatomically correct, realistic and as close to the actor's features as possible. In film and television industry, they can use professional 3D modellers to build a realistic model of a user's face. However, a commonly used and efficient method, but highly expensive is the use of a lightbox [51]. Lightboxes work by having the participant sat in the middle of a frame covered in controllable lights and light receivers; a system then activates different lights and measure both the diffuse and specular light bouncing off the individual's features with high accuracy. The participants will sit in the box and record a series of expressions to create a blendshape library. Unfortunately, due to the required expertise and cost of a professional modeller or a lightbox, these methods are not applicable to consumer bases.

A popular method that applies to consumers is the use of a 3DMM [26] to generate models of a user's face. Once trained, 3DMM can produce a wide array of 3D models from single images without the need of any depth data. 3DMM builds a statistical model of an object, and stores parameters that define the shape, in our case this could be landmarks. By using the model trained on a wide variety of faces, inputting of facial parameters can create a realistic representation of a user the trained model has not seen before, as many people share similar features. However, constrained by the ranges in the training set. 3DMM is commonly used in many different aspects of facial animation as it can also be used to perform some expressions. Also, 3DMM is constrained as from 2D data it cannot collect in detailed 3D features of a face, meaning the model may fit the appearance, but may not have to true depth structure of the individuals.

---

When depth data is available, most researchers implement the use of Deformable Models [52] but this method requires some pre-processing. Firstly, generating a 3D model depth data is performed; this means the participant must hold an expression while rotating their head to build a full model. Then a template model is deformed to the rigid model creating a zero-hole model. This process is slow and required in every expression the model. However, future work using Dynamic Expression Model (DEM) [4] and Displaced Dynamic Expression (DDE) [29] a single capture overcomes the issue, but still requires the pre-processed single model, other methods allow for the models to be built during run-time and refines them throughout the performance [6].

### 2.10.2 Rigging/ Parametrization

Using pre-built models for animation requires the ability to deform its shape to replicate an expression. The model does this by moving the vertices that make up the model's structure. However, a face model is just a visual representation of the facial skin, with no information of how the underlying bones and muscles in a real person deforms the features. The process of rigging is to replicate the skeletal and muscle features of a human on a 3D model and then weight the vertices to the bones to push and pull the mesh. Weighting the bones to the skin is key, as it simulates the muscle skin relationship in real faces.

### 2.10.3 Blendshapes

Blendshapes illustrate in Fig 2.2, are a library of deformations on a 3D model. The method works by deforming a model vertices structure to replication facial movement; done through rigs and vertex manipulation. Each of the libraries deformations can be interpolated to replicate the stored deformations, but this also allows interpolation of multiple deformations at the same time. Blendshape libraries can contain hundreds of different expressions that can be interpolated to

---

either alone or in combination. Blendshapes are the most common method used for facial animation, as it can be used to display realistic expressions in real-time.

In existing works by Weise et al. [5, 4], models are generated as a pre-processing step. To generate a model, the user must perform and hold an expression in front of a depth camera while rotating their head. While holding and rotating their face in front of the sensor, the technique performs a series of tasks:

- Firstly, segmentation of the face from the depth map and generates a 3D model on per frame basis.
- The generated model aligned with a rigid ICP mask, which all previous frames have also aligned, allowing for smooth integration.
- Model deformation is then performed to deform and expand the previous iteration to build a complete view of the face and expression.
- Finally, a template model that contains no holes, then deforms to the integrated model produces the completed mask.

Weise et al. is an effective method but has its limitations, it can effectively build a repository of user-defined blendshapes, but relying on a user's ability to perform and hold expressions is difficult without training and practice. Also, to deform the template mesh, the user must manually annotate to facial feature on the 3D model, which is difficult process even for trained annotators on a 3D model as we demonstrate in chapter 4. Cao et al. [28, 6] expand on this by building an automated method that recognises when a model requires refinement and when a suitable pose is ready. Therefore, unlike Weise et al. because of the automated refinement of pre-set motions, new expression can not be added.

#### 2.10.4 Animation and FAU Generation

For real-time animation, the calculation of FAUs drives a library of Blendshapes. FAUs is a series of values, between 0-1 but in cartoon characters can extend beyond



---

1, that instructs how to deform the character mesh. As the face is complex and Blendshape libraries can be different in scales; there are different standards for AUs. **FAUs** can be coded to follow differently available deformation descriptors, such as:

- **FACS**: [16] An anatomically correct description of facial movements that uses two values muscle moved and intensity of movement.
- **MPEG-4**: [53] Is the current standard in facial animation using **FACS** but tailored towards animation.
- **Custom encoding**: For some methods, such as games and on mobile devices custom encoding methods are defined as full MPEG-4 is not possible with the restrictions placed on models.



FIGURE 2.2: Example of an Autodesk model with a library Blendshapes (Shape-points in Blender) on the right.

## 2.11 Depth Sensing

Depth Sensing in real-time can be done effectively by two existing methods, structured light and **ToF** sensing. This section will evaluate both of the methods and give reasoning to the sensor type we implement.

---

### 2.11.1 Structured Light

Structured light [54] is an older method of predicting the depth of a scene; it projects a pre-made image into the scene, such as Fig 2.3. The device then takes an image of the scene with the pattern projected. As illustrated by Fig 2.4 the image is warped by objects in the scenes as different depths causing the evenly spaced lines to go closer, further and change direction or shape when they encounter an object in the scene. As a result of the warping and with the original image we can use these distortions to estimate how far away an object in the scene is.



FIGURE 2.3: Example of an image used by a structured light projector, some sensors employ variations with different colours or specular patterns to increase accuracy.



FIGURE 2.4: Example of the image from Fig 2.3 projected into a 3D scene, notice how the straight lines curve, change direction and thickness when interacting with objects in the scene.

Although this is an effective method, there is some limitations to this method:

- 
- As illustrated in Fig 2.4 the projection causes shadows, and glare causing some areas to lose track.
  - As the projector and image cannot be at the same location, occlusions due to the viewing differences will be present.
  - The technique can only track areas where the image has contrast, as a plain surface has no information to show depth change.
  - Overly complicated images can confuse the method making it unable to get accurate depth details.

With the first Kinect, implementing a Structured light scanner that projected an IR specular pattern into the scene to calculate depth. However, this was a low-resolution image containing noise artefacts.

### 2.11.2 Time of Flight

ToF similar to Structured Light requires a projector. However, it is more common for ToF to implement IR technology. The sensor blasts IR light into the scene and then uses an IR receiver to collect the burst of lights. As IR light travels at a constant speed by recording the time from projection to the time of return, the distance tracked reliably in real-time. The sensor does suffer from some limitations, such as:

- Occlusion from the projector and receiver being at different positions.
- Differing light absorption through different colours and textures.
- Natural decay and disruption in received IR light.

With the Xbox One Kinect, implements a ToF sensor, providing a higher resolution depth image than the original, but still contains some noise artefacts.

---

## 2.12 Depth based facial landmarking and animation

As shown, in high-end production companies depth data for animation is highly desired especially with the face, where equipment is made to fit the face holding multiple cameras [55]. The current commercial systems, use the multiple cameras and image stereography techniques to “stitch” images together, allowing prediction of some depth information. They also implement visual markers for aid in automatic landmark detection, these markers allow for high accuracy tracking, in all areas of the face, but requires a long set-up time, prone to human error and markers can fall off. The markers, when combined with multiple cameras allows high, but limited range accuracy of 3D landmarks with the calibrated cameras. However, with depth sensors, we can capture a realistic 3D face model in real-time or high-speed (90FPS) with newer sensors, such as the real-sense D435 [56]. A single depth sensor could retrieve millimetre precise facial structure. Also, this would reduce both the weight and complexity of a Head Mounted Display (HMD), but to improve the model, it must have clearer details required for accurate 3D landmarking.

The research in landmarking from 3D data is extremely limited; most work focuses on the pre-built 3D model rather than the depth map. Methods, such as 3DMM [26] and other 3D face reconstruction methods [57, 58, 59] and other statistical model-based methods used to generate models from 2D images, by using a statistical model the vertices are always in predetermined iterations, they can be used to retrieve landmark coordinates. However, even in newer research the models, they look like the individual they are still struggling with some expressions and areas of low detail. Other methods to landmark use raw 3D models generated from depth maps, which follow work on deformable face models, where a template facial mesh is deformed to the structure of the depth map generated 3D model, and a similar method to 3DMM to retrieve landmarking. Another notable method is the use of curvature analysis [60]. However, this only work for landmarks in areas of high curvature change and with low noise models.

---

Rendering and processing on 3D models is a difficult and processor intensive task, a solution to this is to implement computer vision techniques onto the depths maps, which contains the critical 3D information, while in an easily processable format for real-time use. Our work focuses primarily on the use of depth maps to produce accurate 3D facial landmarks, without the need for a 3D model generation or approximation.

## 2.13 Research Direction

With much of the available work focusing on model generation and using tradition machine learning algorithms getting good results, but deviating significantly from how the start of the art methods function. State of the art techniques employs the use of high accuracy landmarking to identify and interpolate FAUs. This method achieves the incredible results seen in many modern day film productions. The issue with state of the art is the use of singular or multiple visual cameras, and retro-reflective markers to identify the key tracking locations, which allows accurate tracking and some 3D prediction possible.

Existing research follow the process of generating or predicting full 3D models for prediction when a much smaller amount of points for accurate facial expression determination can be integrated. The expressions can then transfer to a pre-existing 3D models that have been set-up for animations, in a similar manner as the current research. This help in many aspects as the research in this area does not show how accurate 3D prediction of data is when using 2D data [29], while also avoid complex processor intense model build processes, which require storage in memory.

Detection of 3D landmarks has many challenges that can prevent an accurate method from being deployed. Based on the currently available research there are many challenges to be faced, the complexity of the face, the speed of movement and the noise present in consumer-based depth data. The use of machine learning and statistical models have previously implemented for animation purposes, with

---

the authors created a previously available commercially software [2]. The problem of detecting accurate 3D landmarks is still a common research problem and then determining how many points is sufficient for FAU prediction. Furthermore, whilst machine learning has provided a solution for these problems, the use of deep learning is not used in this field, in which it could provide vastly improved accuracy [18].

## 2.14 Summary

This chapter introduced the theory of expressions and facial movements and why they are required to infer expressions. It then moves on the detection and analysis of facial features, for both expressions and landmarks. Analysing and predicting these point is common in 2D but is much more difficult in 3D, even for trained annotators. However, this field is steadily growing with many researchers targeting the alternative parts required for face representation in 3D. We still have many improvements to make before this research is available for high-end and high accuracy performance.

The work in this area focuses mainly on generating and predicting high resolution and realistic facial models, which is an intensive task and in prediction make assumptions to the facial structure. However, in reality, only the key tracking points are required for the transference of expression to a 3D model. The key position can be generated and synthesis in a similar way to the 3D model, but with less processing power due to the reduced amount of points for generation and the need for triangulation removed. This chapter has described the problem of retrieving the landmarks in 3D and the issues with prediction and generating of 3D models for use in this field.

# Chapter 3

## Theories and Techniques

*This chapter explains the theories and reasoning of the techniques being used to solve the problem statements. The section will explain in detail the inner working of each of the techniques implemented in the thesis.*

### 3.1 Introduction

The literature shown in the previous chapter shows that the research in this area is widespread, covering many different aspects. Furthermore, we are focusing on landmarking facial features in depth data for expression recognition and animation generation. The current section of the thesis we describe the theories and techniques used to solve the issues outlined in chapters 1 and 2 of the thesis. This chapter describes the camera settings, perspectives and different aspects of a depth sensor, such as how the **RGB** image is recorded differently to the depth map. Then, the techniques used in face image processing are described, both **RGB** and depth. Finally, the use of deep learning in face processing and relevance performance metrics are presented.

---

## 3.2 Camera Settings

The camera used is a crucial section of the facial analysis, in our work we focus on the Kinect version 2 as this is used in face research [61]. As the Kinect is a multi-sensory device with 3D capabilities, background knowledge of the camera spaces and 3D viewing frustum is required.

### 3.2.1 3D Viewing Frustum

In 3D environments, virtual cameras are used to view the scenery, to decide which sections of the environment are rendered virtual cameras implement viewing frustums that both identify and rendered the 3D geometry in view; This section will explain and describe the different setting of the frustum. The primary setting is the view mode, orthographic or perspective which drastically changes the image generated by:

- Orthographic cameras use a box-shaped frustum to choose which objects to render. However, unlike perspective cameras, the box-shape removes the perception of depth.
- Perspective cameras, unlike orthographic, uses a pyramid-shaped frustum as illustrated in Fig 3.1. The pyramid-shape allows the perception of depth in a 3D scene.

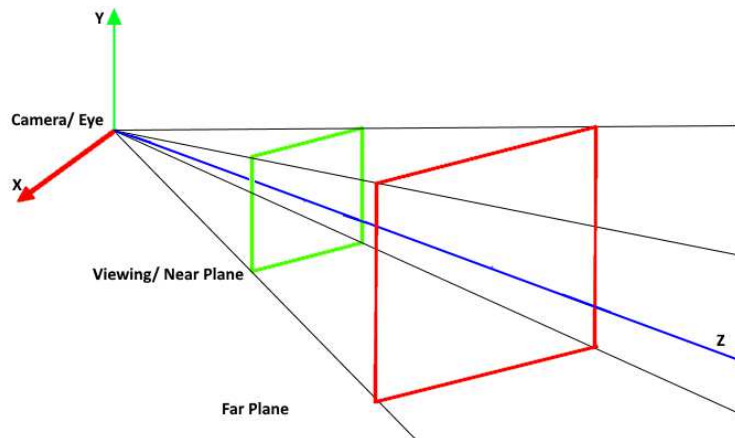


FIGURE 3.1: An example of a 3D viewing frustum.



---

As shown in Fig 3.1, the camera acts as the eye in the scene. The near and far viewing planes are used to cull and crop objects that are not in the view of the camera, this reduces the processing requirements before rendering. The final setting is the Field of View (FOV), which determines the height and width of the camera's view.

### 3.2.2 Alternative Viewing Spaces and Metrics of Multi-Sensory Devices

Multi-Sensory devices, such as the Kinect uses different metrics for the different views as shown in Fig 3.2. Each sensor produces images. Due to different FOV, the resolution and information captured by the RGB and depth cameras are varied, as illustrated in Fig 3.2. Knowledge of the different spaces is required to use multi-sensor devices effectively. The Kinect has three different viewing spaces:

- *Image Space*: This is the space of the colour image and measured in pixels, as standard, the images are full high definition (1920\*1080) in blue, green, red and alpha format.
- *Depth Space*: This is the space of the depth image, where the colour image stores the colour intensity at each pixel, depth image store the distance of an object from the Kinect in millimetres.
- *Camera Space*: This is a middle space it contains the information of the sensor position in relation to each other, with the distance measured in meters.

Because of the different available spaces, the device requires calibration to allow coordinates to map between the different views.

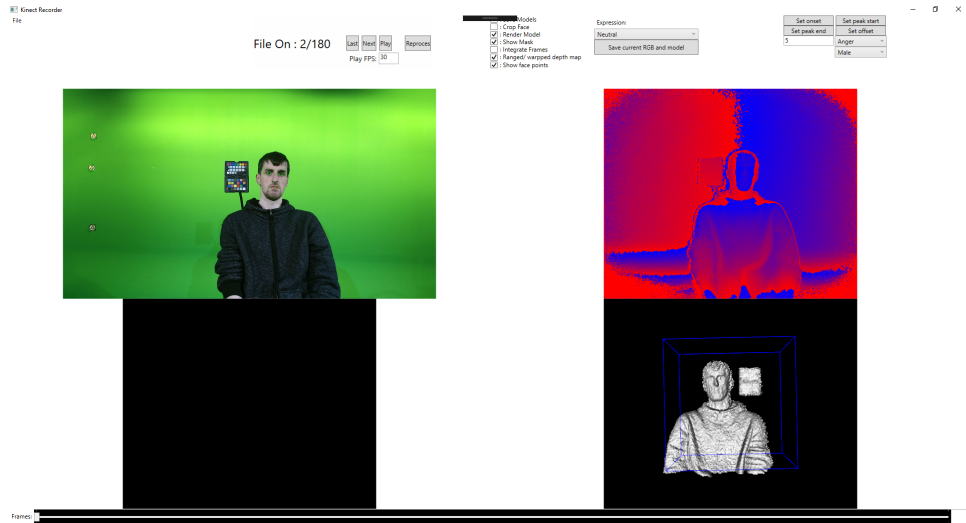


FIGURE 3.2: Illustration of different viewing spaces from the Kinect depth sensor.

### 3.3 Face Processing

Facial analysis is a vast research area covering but not limited to face detection, segmentation and landmarking. For face processing, much of the research is extended and requires other techniques to be used as pre-processing steps, such as landmarking methods as standard use face detection to crop out the face before landmarking. This section will provide an overview of the different stages of facial analysis.

#### 3.3.1 Face Detection

The purpose of face detection is to localise a face and produce a bounding box for the face. Once the face has a bounding box, it can be cropped and resized for further processing. The Viola and Jones [22] is a commonly used method of object and face detection as it allows fast image processing with high detection rates. The algorithm splits into three main processes:

- 
- First the collection of Haar-like features with an integral image. The integral image is a summed area table that uses rectangle features, produced by the Haar Features.
  - Secondly, The AdaBoost [62] learning algorithm is used to increase training times and select the best features for the classifiers.
  - Finally, the Viola and Jones use a series of weak classifiers in tandem to produce a reliable result.

However, new techniques employing deep learning networks have been implemented [63] and have vastly improved on accuracy under a wide range of head poses. The deep learning network attempt to predict the top left and bottom right corner of each face allowing a bounding box to be inferred. However, more recent work implements complex multiple process networks, such as Mask-RCNN [64] and Hyperface [43] to detect an object, crop, resize and process the output further. The ability to detect and crop a face inside a network allows all-in-one systems to be designed.

### 3.3.2 Landmarking

Facial landmarking is the process of identifying and localising critical locations on a face image. The location of the facial landmarks is shown in pixel position in relation to the input image size. In a traditional system, the use of machine learning techniques with random regression forests is employed [65] or statistical shape models. As with face detection, deep learning has improved the result of landmarking.

Accurate landmark localisation is a key for many systems, such as facial alignment. In facial animation the landmarks can be used in two ways:

- 
- Pre-recorded expressions: commonly used in high-end productions, recording expressions before performance allows references to be made. After the pre-recorded motions, new sets of landmarks can be compared for accurate facial animation inference.
  - Model-based inference: by normalising the facial landmarks with a neutral frame. A trained model can take input landmarks and estimate an expression or FAU.

### 3.3.3 Denoising

Denoising is a large area of research in 2D image processing [66]. The use of methods, such as BM3D [67] allows for the noisy images to be restored. However, for denoising depth images, RGB data is used to inform the system, to remove and “smooth” noise from an image. However, the RGB systems can struggle in areas of high contrast change, such as text on a shirt can cause inconsistencies with results.

The use of neural networks on this problem has been widely researched, with many networks achieving results, near indistinguishable to humans [68, 69, 70]. However, this method has not been extended to depth data. For the denoising networks to be effective, a large scale dataset is required, but for depth data, this is currently not available.

### 3.3.4 Automated Processing

For our work, we use the Dlib [71] package to detect faces in an image. Dlib combines multiple methods to achieve high accuracy and reliable face detection. Dlib uses HOG as a feature descriptor and a linear classifier to determine a face. To aid in different face scales and image resolution, Dlib also uses the Image Pyramid technique and the traditional sliding window. Another solution is the CLM-framework [72], also known as the Cambridge facial tracker, which provides

---

an extended amount of tracking features, such as head orientation and AU prediction. However, there are license restrictions on its use and due to the extended features requires more time to process. The CLM-framework also relies on Dlib for its core components. The developers of CLM have switched focus to the new projects of OpenFace Toolkit version 1 and 2 [73, 74] which incorporate deep learning algorithms for prediction of landmarks, FAUs and head pose, improving both processing time and accuracy of the toolkit. Similarly, commercially available tools are available such as Face++ [75] which incorporate high accuracy deep learning models for high-end results, but cannot function in real-time as data must process on their servers. As shown many effective systems have implemented Dlib as a base for their systems, as it is an effective method public available, and due to time restrictions, we use this as the core automation program.

For landmarking, Dlib uses a method by Kazemi et al. [33], that allows for high accuracy landmarking of facial images to be performed within a millisecond. They propose an approach that uses multiple regression trees working in tandem. The decision is made using intensity threshold differences at the pixel level. In current works, the use of CNNs is state of the art achieving high accuracy scores.

### 3.3.5 Depth Processing and Normalisation

Depth data has a variety of ways to be stored, and some researchers prefer to render and only store 3D models. In our work, we store our depth maps similar to images. We flatten images into 1-dimensional arrays and save as a 16-bit unsigned short integer. Using 16-bit unsigned short integer allow for values outside normal image range, 0-255, as the Kinect uses a millimetre metric, with a maximum range of 8 meters, giving us a range between 0-8000.

As we store the image as 1-dimensional arrays once loaded and reshaped back into 2D matrix the structure, similar to images. However, there are some considerations to be made before processing:

- 
- Before processing, any values outside the reliable range requires culling, Kinect reliable range 0-8000. The value that is not in the reliable range resets to 0.
  - Many kernel methods use the fact that images have colour information that can smoothly blend or sharp edges. Depth has both of these but contains a server amount of noise which can affect many traditional methods negatively.

Depth data can be treated both as normal raw data, but some methods may require normalisation for depth data we use the following equation:

$$\text{Norm} = \frac{\text{Input} - \text{Min}}{\text{Max} - \text{Min}} \quad (3.1)$$

where:

- Norm: is the normalised value.
- Max: is the maximum value, 8000 for Kinect.
- Min: is the Minimum value, with the Kinect values under 400 are considered unreliable, but as unknown and unreliable values reset to 0. Using the value 400 as the minimum would cause a physical wall to appear in generated geometry.

## 3.4 Deep Learning

Deep learning with CNNs is a subset of traditional machine learning, but in recent years has significantly improved the accuracy and reliability of methods. Deep learning can be applied to many linear and non-linear challenges. The principal behind Deep Learning with CNNs is that the computer learns both the features to extract and how to interpret those features. We give an overview of how deep learning works and the background details on how a network begins to learn. After the background, we show each stage of a traditional CNN and describe the inputs and the process of each section.

---

### 3.4.1 Background

The theory behind the use of machine learning is the use of derivatives. Derivatives have two forms, but for deep learning, we use them to determine the gradient of a line at a certain point. The slope predictions are useful in deep learning during the training stage of a network, as the network learns processes examples and compares to the ground truth the optimiser can plot the errors as described by the loss function. Furthermore, when the optimiser adjusts the parameters to improve data prediction, it creates additional plots. By using derivative functions, we can determine the gradient; this allows us to determine if the network is performing better or worse. Derivatives are especially useful with deep learning as it can model non-linear functions as linear.

An example of how derivatives are calculated is provided. If we use the function illustrated in Fig 3.3, where:  $f(x) = x^2 + 5$ , we can calculate the slope between two points with equation 3.2. However, this requires two points, to solve this and calculate the derivative, we modify the input function to add a minuscule value, such as 0.00001. By adding a small value, we can calculate the slope at a specific point.

$$slope = \frac{y_2 - y_1}{x_2 - x_1} \quad (3.2)$$

where:

- $y$  is a 2D vector
- $x$  is a 2D vector

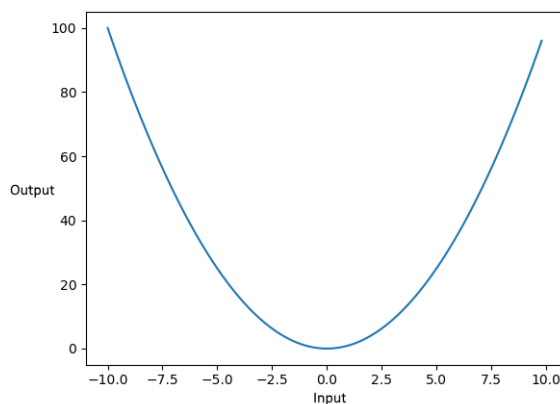


FIGURE 3.3: This illustrates how changing the input  $x$  to function  $f$ , affects the output value, creating a ‘u’ shape.

### 3.4.1.1 Convolutions

A convolution is a kernel-based method to detect features in an image, it implemented a small sliding matrix (kernel) over a matrix (input image) and applying basic matrix operands on the values in the image, by the values provided in the kernel. The kernels can perform a wide array of tasks, such as edge detection and direction. In a [CNN](#), a convolution layer has adjustable parameters, such as:

- *Kernel Size*: This value determines the kernels height and width in pixels. Adjusting the filter size influences the networks ability to determine specific features in an image.
- *Stride*: This value determines how much the kernel moves after each calculation. By increasing the stride, due to the nature of a kernel calculation, the image is also down-sampled as it begins to skip pixels.
- *Padding*: Due to the nature of kernels where it used to calculate a single value, for the pixel in the middle of the kernel, edges are usually skipped, reducing the image size. Padding allows for the images to have zeros placed around the edge, so the first pixels fit into the image edge, maintaining image size.



- *Filter Count*: This is how many kernels are used on the input image, each kernel is running on the raw image and gives a separate output image. By using an array of kernels at each layer, a single layer can identify a multitude of different features as each kernel can focus on highlighting alternative features, as shown in Fig 3.4.

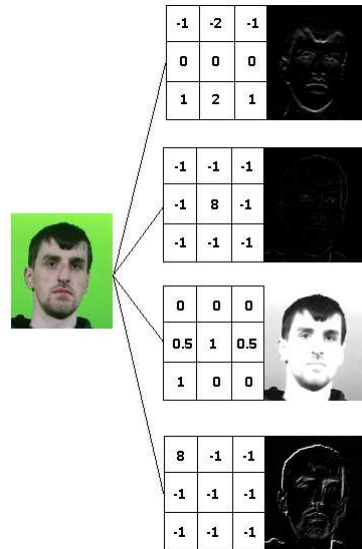


FIGURE 3.4: An illustration of different filters that can be used to highlight features.

### 3.4.1.2 Activation Functions

Activation functions take in values, processes them through a non-linear function and use this value to determine if the neuron is activated. A commonly used activation function is the [ReLU](#), as it can aid increase convergence speed compared to more complex sigmoid and tanh functions, but is limited as some neurons can ‘die’, due to the inability to use negative values.

In our work, we focus on the use of [ReLU](#) as other methods, such as sigmoid, illustrated in equation 3.3, normalises the data between the range of 0-1. Furthermore, for landmarking, this reduces the capability of the method as it cannot return a pixel value, visualised in Fig 3.5. Whereas [ReLU](#) allows for an infinite range of output values that are non-negative, this allows pixel positions for any scale of images to be trained.

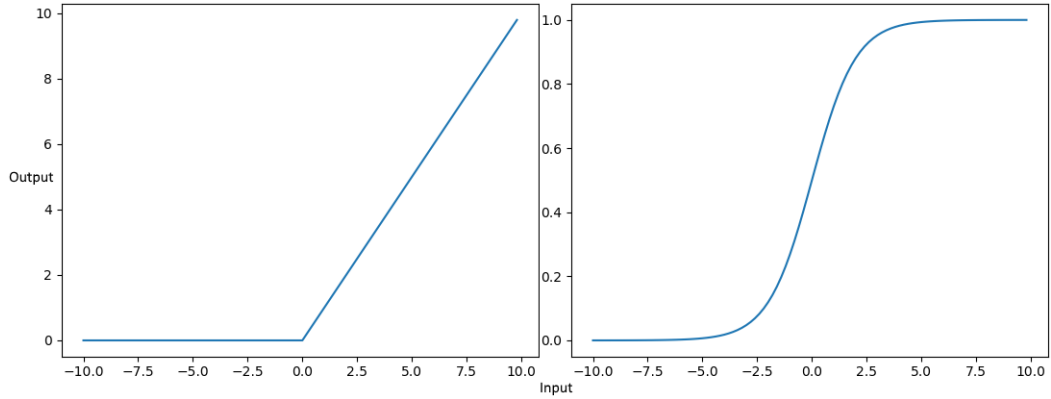


FIGURE 3.5: This image shows **ReLU** (left) activation vs sigmoid (right), notice how sigmoid normalises the range, but **ReLU** allows an infinite range, but without negative values.

$$f(z) = \frac{1}{1 + e^{-z}} \quad (3.3)$$

where:

- $z$  is the input to a function
- $e$  is an exponential function

$$f(z) = \max(0, z) \quad (3.4)$$

where:

- $z$  is the input to a function
- $\max$  returns the value of  $z$  if it is greater than 0, else it returns 0.

### 3.4.1.3 Pooling

Pooling is used to downsample an input, by focusing on certain values in the image depending on the type of pooling used, such as in Max pooling. Max pooling splits the input into separate non-overlapping segments and then for each window

---

returns the highest values in each window, as illustrated in Fig 3.6. In pooling, the size of the windows that split the images is controllable. Max-pooling similar to convolutions can have variable sizes for the pooling window.

123	45	20	253
85	35	134	242
56	24	43	23
10	80	200	111

=

123	253
80	200

FIGURE 3.6: An example of a max-pooling window the network focus on the highest values.

#### 3.4.1.4 Fully Connected Layers

Fully connected layers can only take single dimensional data, and each neuron connects to all values in the previous layer. Each connection has a weight applied, with basic matrix operation with the addition of a bias value. These layers are commonly referred to as the [MLP](#). As the input to neural networks for our work is images, which consists of two or more dimensions based upon the channels available in the final layer, matrix reshaping has to occur. As the matrix is reshaped, the network suffers from loss of spatial data.

#### 3.4.1.5 Output

The output is the last layers of the neural network, regarding regression the final fully connected layers, can be used as output. Whereas, in classification specialised layers are required, such as softmax. Regression works natively with neural networks as the [MLP](#) performs weighting on the output neurons as it is processed, these weight act as regressors and refine the values to their required values.

### 3.4.2 Loss

The loss is the error the network makes during training, at each stage of the network, such as convolutions, we can calculate how far from the result we want

the network is. The loss, used in determining the effectiveness of a network as high loss indicates the networks is deviating from the wanted results by a high amount. Loss can be calculated, in many ways, such as [MSE](#) and [MAE](#).

The loss has a significant impact on the performance of the network by affecting how it will learn. As shown in Fig 3.7, when de-noising images the type of loss drastically affects the results of the cleaning. In Fig 3.7 we show overall, the traditional method achieves better results than [MAE](#) trained network in, [MAE](#), [MSE](#), accuracy and [PSNR](#), in which we discuss further in Chapter 6.



FIGURE 3.7: This image shows how changing the loss function affects the result of image de-noising. Even though the network is trained to optimise [MSE](#), it still achieves a better [MAE](#) score than the [MAE](#) based network. The network was trained on the Morph dataset.

The difference, highlighted in Fig 3.8. [MSE](#) allows the hair to become better defined and the bottom eyelids clearer.
















	Clean	Noisy	MSE	MAE	PSNR
Example 1					
Example 2					
Example 3					

FIGURE 3.8: This image shows the effect of the denoisers when trained on different losses.

### 3.4.3 Optimiser

Optimisers are used exclusively in the training stage of the network; they update the weights of the network by using the information gathered from the loss stage. The purpose of the optimiser is to guide the weights while avoiding the “man on the hill” problem: If a blindfolded man is stood on top of a hill, and needs to get to the bottom how does he get down? The problem seems simple, as he slowly walks down the hill, the moment the incline stops the man has reached the bottom. However, as shown in Fig 3.9, in reality, there are many stages in which the gradient evens out, but has not reached the lowest point. The sections in the gradient that even out are referred to as local-minimums and the optimisers, try to avoid these to best perform the task they are learning.

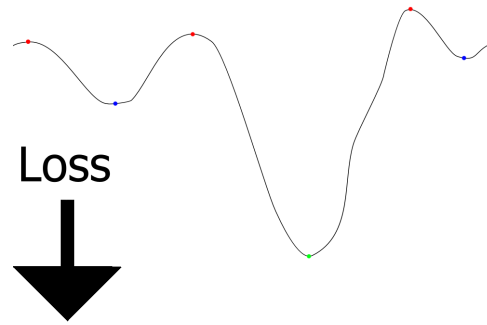


FIGURE 3.9: An illustration of a deep learning gradient; the loss can start at any point along the line. Blue points show local minimums, in which the network can appear to have converged but have not fully. The green point is the global minimum at which we want our network to reach. Red points indicate local maximums, which we want avoid.

A [CNN](#) works by combining these different sections into one complete network; this section will show how a network is formed and trained. A standard [CNN](#) network takes a single image as input; input image is fixed size meaning images are resized to fit the required dimensions. The image is processed using convolutions with activations, to reduce over-fitting and processing requirement pooling is integrated [\[76\]](#), as it maintains a high-value feature set. After a series of convolutions, activations and pooling the outputs a flattened into a single dimension, losing spatial information. The fully connected layer weight the features using matrix operators with the inclusion of learnt biases, to determine the logical output.

A [CNN](#) requires training to provide good results; this requires a large amount of labelled training data to ensure high accuracy results. The network learns by processing the training data in batches; larger batches allow the network to become more generalisable. Each batch of training images are processed by the network with the loss being calculated at each stage until the output has been produced, this is forward propagation. The next stage uses feedback from the loss and the optimiser to adjust the weights of the neural network based upon the learning weight value, this is known as backpropagation.

An example of the method Adam [\[77\]](#) or Adaptive Moment Estimation is given as this is the loss function performs well in this field. Adam is an improved

---

method that learns from previous methods, such as AdaDelta, by remembering previous past gradients, as shown in equation 3.5. Then, the weights are then updated throughout the network with equation 3.6.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (3.5)$$

where:

- $\hat{m}_t$  is a vector of means gradients
- $\hat{v}_t$  is a vector of the gradient variance
- $\beta_1$  is 0.9, given by the author of the method
- $\beta_2$  is 0.999, given by the author of the method

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \exp} \hat{m}_t \quad (3.6)$$

where:

- $\hat{m}_t$  and  $\hat{v}_t$  are the values shown in equation 3.5
- $\exp$  equals  $10 \times \exp(-8)$  as given by the author of the method

## 3.5 Performance Metrics

In this section, we describe and explain the different type of metrics we use throughout the thesis to evaluate the work performed. We begin with the performance metrics, which provide views of how the method is performing, such as [MSE](#) for the distance from the correct answer. Then, the statistical methods used for evaluating techniques and comparing with others are presented.

---

### 3.5.1 Loss Functions

As the problems of landmarking is a regression-based issue, the use of [MSE](#) as a loss function is implemented. [MSE](#), as shown in equation 3.7 used in many deep learning architectures.

$$MSE = \sum_{i=0}^n \frac{(y_i - y'_i)^2}{n} \quad (3.7)$$

where:

- $n$  is the number of samples in the training batches.
- $y_i$  is the ground truth for the training image.
- $y'_i$  is the predicted output for the training image.

As shown in equation 3.7, by squaring the value the function focuses on large deviations from the ground truth. The focus on large error allows a network to avoid major mistakes, and if an error does occur it is naturally small. However, this does mean [MSE](#) commonly overlooks small errors.

Additionally, the use of [MAE](#), illustrated in equation 3.8, can be used as a loss function. [MAE](#) allows equal weighting to all errors. By allowing equal weighting, the method can be more robust but is more prone to significant errors.

$$MAE = \sum_{i=0}^n \frac{|y_i - y'_i|}{n} \quad (3.8)$$

where:

- $n$  is the number of samples in the training batches.
- $y_i$  is the ground truth for the training image.
- $y'_i$  is the predicted output for the training image.



---

For classification based problems different evaluation techniques are required, compared to regression problems. Classification methods produce a yes or no result, but regression can produce a wide range of values. We use classification in our methodology. Thus we show accuracy metrics. Classification with deep learning can be performed in two ways:

- One-hot: A binary method of determining a class, for each output there is one output, but only one can be “hot” equal to one. As shown in equation 3.9, binary accuracy as default in Keras, rounding is performed allowing for a 0.5 error margin.
- Categorical: This works in single output networks with multiple classes, the output of the method is a range, such as 1-5. The focus is on when output is in the same range as the class. As shown in equation 3.10 categorical accuracy uses the maximum predicted values, which is why value in the range predicts as true.

$$BinaryAccuracy = \frac{X}{i} \quad (3.9)$$

where:

- $X$  is the number of correctly predicted classes when the predicted is rounded to the nearest class.
- $i$  is the number of images in prediction.

$$CategoricalAccuracy = \frac{X'}{i} \quad (3.10)$$

where:

- $X'$  is the maximum value of correctly predicted classes when the predicted is rounded to the nearest class.

- 
- $i$  is the number of images in the prediction

As the aim of Deep Learning is to reduce the loss, some metrics cannot be integrated as a loss function, such as [PSNR](#) illustrated in equation 3.11. [PSNR](#) is used as a guide to how well denoising algorithms are performing, but the higher the value, the better the algorithm has performed, meaning a network would naturally train to get worse results, rather than better.

$$\text{PSNR} = 20 \times \log_{10}(\text{MAX}) - 10 \times \log_{10}(\text{MSE}) \quad (3.11)$$

where:

- MAX is the maximum possible value in the ground truth, for the depth data, this is 8000.
- MSE is equation 3.7.

As shown in equation 3.11, [PSNR](#) derives partly from MSE, but also consider the maximum possible value to identify how much noise in relation to a signal exists.

## 3.5.2 Statistical Analysis

### 3.5.2.1 T-Test

The t-test is a method of determining if there is a significant difference between the means of two populations. The t-test is used to highlight both the difference, t-score, and the chances that the data occurred randomly with the p-value. However, the t-tests have two different variants:

- Standard t-test: is used for comparing two separate populations, such as in healthcare where there is a control group with placebos and the test group. In the standard t-test, no participant can exist in both populations.

- 
- Paired t-test: is used for comparing one group before and after, such as improvements after training or for putting two populations through the same procedure, such as vehicle safety check were all manufacturers have to go through the same process.

$$t = \frac{(\sum D)/n}{\sqrt{\frac{\sum D^2 - ((\sum D)^2/n)}{(n-1)(n)}}} \quad (3.12)$$

where:

- $\sum D$  is the sum of the difference between each of the participant's results
- $\sum D^2$  is the sum of the squared differences between each of the participant's results.
- $(\sum D)^2$  is the sum of the difference between each of the participant's results, squared
- $n$  is the number of participant's
- $t$  is the t-score

In our experiments, we use a paired t-test. The t-score is calculated using equation 3.12. The equation uses versions of the sum of the difference between participants before and after results, divided by how many participants there are at different stages.

The second stage of the algorithm is to determine if the results are by chance using the p-value. The p-value is predetermined and set by the user, the lower the p-value, the more chance of a null hypothesis, but shows the significance of the result better. Many tests employ a p-value of 0.05 for experiments. To calculate this, we use  $1 - n$  and look up the value in a t-distribution table. By comparing the value in the t-table to the t-score, ignoring  $\pm$  as it works both directions, with a set p-value a calculation to perform the significance of the results is possible.

---

### 3.5.2.2 Standard Deviation

The **SD** calculation, illustrated in equation 3.13 is used to determine by how much the results deviate from the mean. By calculating the deviation, it allows tracking of the consistency of the data and how far out of the standard range value is. The lower the deviation, the more consistent the results are. Whereas, the higher the score, the less consistent and harder to predict the results will be.

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n (X_i - \mu)^2} \quad (3.13)$$

where:

- $\sigma$  is the **SD**
- $n$  is the number of samples
- $X$  is the input number
- $\mu$  is the mean of the input numbers

As shown in equation 3.13, the deviation is calculated by subtracting the mean from all the input values and squaring the result. By squaring all results of the calculation are positive. The next stage is to average out the squared values and find the square root. The result is the deviation  $\pm$  from the mean value. For example, if we were determining the time to perform a task in minutes, and the mean time was an hour with an **SD** of 5, the task would take between 55-65 minutes to complete.

The **SD** also only works for the population trained. For example, if the deviation calculates for height in a specific country, it could not be used for other countries as the deviations would be different.

---

### 3.5.2.3 Confusion Matrix

To determine the effectiveness of face detection methods, the implementation of a confusion matrix gives valuable insight to the performance. The matrix consists of 5 values:

- The total inputs for evaluating the method.
- True positives, the number of values that correctly classified as true.
- True negative, the number of values that correctly classified as false.
- False positive, the number of values that incorrectly classified as true.
- False negative, the number of values that incorrectly classified as false.

The matrix gives a general view of how the method is performing. However, from a confusion matrix, we can calculate a series of values, such as accuracy, precision, specificity and recall, which give in detail views on the performance.

Accuracy, as shown in equation 3.14, shows how well the technique is performing by correctly classifying items.

$$Accuracy = \frac{(TP + TN)}{Total} \quad (3.14)$$

where:

- TP is the number of true positives
- TN is the number of true negatives
- The total is the number of inputs in the matrix

Specificity as shown in equation 3.15, shows when the method accurately predicts false, to check against false positives.

$$Specificity = \frac{TN}{Total} \quad (3.15)$$

---

where:

- TN is the number of true negatives
- The total is the number of inputs in the matrix

Precision as shown in equation 3.16, shows when the method accurately predicts true, to check the true positive rate.

$$Precision = \frac{TP}{Total} \quad (3.16)$$

where:

- TP is the number of true positive
- The total is the number of inputs in the matrix

### 3.5.3 Distance Metrics

Distance is a key aspect of understanding facial structure and its deformities allowing accurate measurement of when features move in relation to each other. The two commonly used methods are L1, Manhattan and L2, Euclidean distances.

Manhattan distance, as illustrated in equation 3.17, was created for travel in places, such as New York where the distance can have obstructions, so will not always be a straight line, the algorithm has to consider building and corners.

$$Manhattan = \sum_{i=0}^n |X_i - Y_i| \quad (3.17)$$

where:

- X is the starting point vector
- Y is the ending point vector

---

For deep learning, Manhattan has little use, as even in 3D we only require the distance at a straight line.

Euclidean distance, as illustrated in equation 3.18, calculates the distance in a straight line. Calculating the exact distance for a point is useful for error calculation.

$$Euclidean = \sum_{i=0}^n \sqrt{(X_i - Y_i)^2} \quad (3.18)$$

where:

- $X$  is the starting point vector
- $Y$  is the ending point vector

## 3.6 Summary

This section has given an in-depth view of the techniques and performance metrics used within the thesis. The camera used in the experiments was explained in detail, including the alternative camera spaces and the metrics used. We showed the traditional method of face detection and landmarking. The landmarking is a core section of our methodology, and this gives insight into facial topography and indications of occurring movements. After understanding how to track the movements, the evaluation metrics for the features and why they are used. As the work uses 3D methodology a description of the 3D viewing frustum is provided. The viewing frustum helps to understand how the interpretation of 3D data is performed. It also leads to the use of multi-sensory devices and the different viewing spaces available. Deep learning is used to process the data from the sensor and is much of the focus of the thesis is using the techniques made available through this process. Then, we demonstrate deep learning, both at a consumer base and for accurate 3D landmark determination.

# Chapter 4

## Preliminary Research

*This chapter analyses the current method of 3D ground truth annotation and presents a new tool to overcome issues in existing datasets. An in-house RGBD dataset is created and shared with the research community.*

### 4.1 Introduction

From the chapter 2 literature review, the limitations of the existing datasets were summarised as:

- Annotation had no procedures or conventional method.
- Datasets lacked a large quantity of simultaneous RGB and depth capture.

This chapter will address these limitations by providing solutions to bridge the gaps.

The annotation of 2D face data has become a widely researched area of image processing, with a vast amount of tools to aid in the annotation of images [78, 79, 80]. With the recent interest in deep learning, Face++ [75] has been used in many research [81, 82] for facial landmark annotation in 2D. The annotation software speeds up the process of annotation by predicting points, predicts future



---

video frame landmarks and automatically outputs the points to remove human error from the manual annotation. In 3D annotation, there is a reliance on 3D modelling software, such as Maya [83]. However, Maya is a 3D model creation software and outputting the vertex data requires in-depth knowledge of the Maya Application Program Interface (API).

Moreover, the system cannot naturally generate an output file containing the vertex locations. Other 3D annotators exist in Faceshift’s [2] built-in annotation tool, but this does not use texture information. Hence, the identification of points is difficult. Furthermore, Faceshift cannot output the annotated points and relies on real-time capture from the Kinect (Xbox 360 version).

The purpose of annotating data is to create an effective training set for machine or deep learning. Mathias et al. [84] showed that using a strict annotation regime can produce classifiers using machine learning with small-scale datasets, that can match the accuracy of a classifier trained on a large-scale dataset. The ability to train on smaller datasets benefits 3D data, as there are less publicly available 3D datasets.

In 2D data features such as the eyebrows and lip edges were used to track movement, but in 3D without texture, these points are difficult to identify. As the lips become fused (one surface) when the mouth is closed, and the lack of surface detail for the eyebrow makes the region difficult to identify. Although, the facial features are hard to recognise without texture, on clean models the use of curvature analysis can identify and highlight features on a clean model.

## 4.2 3D Data Annotation

In this section, we introduce the methodology of the created tool to overcome the restrictions in annotating data. The tool uses novel alignment techniques allowing a 3D model to be aligned with a reference image, relative to a virtual camera, without the need for calibration data. The work was accepted and presented at

---

the 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017).

### 4.2.1 Related Work

This section will review the available methods to determine the efficiency of the methods and suitability for facial annotation. There are three commonly used methods to annotate 3D models:

- *3D modelling toolkit*: Software, such as Maya allows the use of its built-in vertex manipulation tools and the MEL script interface to retrieve individual vertex coordinates. However, this cannot output to a file, without the user having expert knowledge in the Maya API, MEL and python. The method of using 3D modelling toolkits is slow, requiring experience with the software. Maya offers the best interface as it allows the creation of user-defined scripts, which can add features to the system allowing for an annotation interface. Using 3D software to annotate also suffers from identical limitations as Faceshift, where the points are difficult to identify. However, the user interface allows for intuitive control over the model's position, rotation and scale to aid in locating the model features.



FIGURE 4.1: Example of two face models of the same individual taken from [1]. The neutral face (left) and the eyebrows raised (right).

---

Fig. 4.1 illustrates the difficulty in identifying some facial features, such as the eyebrows on a model. The issue of identifying points on a 3D model increases when the vertex locations are displayed (Fig. 4.1).

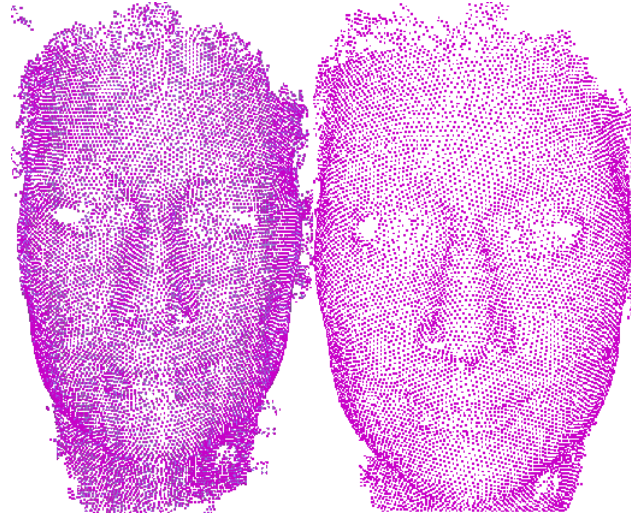


FIGURE 4.2: The same models as Fig. 4.1, but with the vertex locations displayed.

Using 3D modelling software for annotation provides a method of retrieving vertex locations on a 3D model. However, the whole process is very time consuming and prone to human error.

- *2D image interpolation*: Image interpolation [85] requires the depth sensor has a built-in RGB camera, which simultaneously captures both images and models. If the depth sensor has a built-in RGB camera, the system will have the capability to generate three types of coordinates:
  - *Colour space*: The coordinates of the image produced in pixels.
  - *Camera space*: This is the cameras location and the position of the available cameras/sensors.
  - *Depth space*: The distance of the object(s) scanned from the sensor. The distance metric set by the sensor during calibration; as standard, it is millimetres for the Kinect [86, 87].

By using a coordinate mapper to map between the different spaces, the identified 2D (Colour space) landmarks can infer to the 3D (depth space)

---

position. However, with cameras where the sensor and camera remain relative, giving an ideal setting for texture alignment, the texture can still be incorrectly aligned. Another issue with interpolation is that it relies on the available 2D methods to identify landmarks. The use of 2D landmark detection restricts the number of landmarks the system can place, as only features that 2D systems can accurately identify can be used. Furthermore, this method relies on machine learning to annotate data for training. As 2D landmark detection is not perfect, any errors from the 2D location of landmarks will be used to train the new 3D system.

- *Shape-based*: The most recent method by Zulqarnain et al. [88] uses shape-based analysis to predict the facial landmarks. This method relies on the facial curvature to identify the landmarks. The method works well on clean data when identifying areas of high curvature change, such as the eyes and lips, but in lower change areas, the system struggles to identify the landmarks. Also, noise severely hampers this method's ability to detect 3D based landmarks. Overall, this is an effective method of determining landmarks on clean models. Although it uses a process of refinement to identify landmarks, which requires time to process.

Many systems do not, by default, texture the model that the depth sensor produces. To texture, a 3D model post-processing of the images and models is required. Texture generation can be performed if the camera and depth sensor remain relative or the camera calibration data is available. By calculating the colour, camera and depth space, a texture transfers between the image and model. However, due to noise, glare and distance, even with relative sensors (such as Kinect), the texture mapping method can be inaccurate as the texture information might 'bleed' into other objects. For example, a red cup on a table captured the sensor can cause the cup's texture to bleed onto the nearby table structure, this is known as texture misalignment. Because of the texture misalignment and the inherent difficulty of annotating the 3D models, the misaligned texture can cause incorrect identification of landmarks. Franken et al. [89] create a method of producing

---

accurate UV maps. However, this method similar to the annotator produced uses points placed on the texture and model to calculate texture coordinates. This method uses a multitude of points and multiple images. Whereas this proposed method uses fewer points and a single image, but if some of the landmarks are occluded realignment with other images is possible.

The existing 3D annotators are arduous and prone to human errors. Furthermore, without the texture information, identifying facial landmarks is difficult. As the landmarks are difficult to identify, locating the facial points in Maya is ineffective for many facial features. The proposed method will resolve the issues of identifying facial features, by using the 2D image data for identification of the facial movement. The proposed method uses a series of intuitive user-interfaces to speed up the time required to annotate models and output the landmark coordinates.

## 4.2.2 Proposed Method

The proposed method requires a thorough knowledge of the viewing frustum, which is the primary method of viewing a 3D scene. The viewing frustum (Fig. 3.1) is built up of many components:

- *The camera:* The camera acts as the eye in the scene.
- *The near and far viewing planes:* These are used to crop objects that are too close or far from the camera. Cropping the object means it will not process for screen rendering.
- *FOV:* The FOV controls the height and width of the viewing plane. The FOV settings can be used to replicate real-world cameras.

Images loaded into the 3D scene must retain their native aspect ratio. For example, an image of aspect ratio 5:3 placed into a frame using the ratio of 16:9 will stretch, increasing the difficulty in alignment. Without the image in the correct

---

aspect ratio, alignment without non-uniformed scaling will not be possible, the proposed method uses uniformed scaling to achieve alignment. Uniform scaling reduces the amount of scaling errors considerably as early iterations may not match rotation and will cause unnatural scaling of the face model to occur.

This proposed solution relies on:

$$\overrightarrow{AB} = B - A \quad (4.1)$$

where  $\overrightarrow{AB}$  is the direction and magnitude between the position vectors  $B$  and  $A$ . By using the vector result of 4.1, we can increase the magnitude through objects in the same direction as the vector. By extending the vector, if we have a 2D image and a 3D model behind the image we can calculate the point at which the beam would interact or ‘hit’ with the model. By using this equation if a model and image are aligned, annotating the 2D image could yield the 3D results. To interact with an object in the scene, the method used is as:

$$\overrightarrow{R} = V - C \quad (4.2)$$

where  $\overrightarrow{R}$  is equal to the coordinates on the viewing plane where the cursor is present  $V$  minus the camera position  $C$  (Fig. 3.1 Dashed purple vector), giving a directional vector into the 3D scene. Furthermore, by extending the magnitude of the vector, we can calculate the exact position which the vector would collide with an object, the face model, in the scene. The extended vector is used as a base method to place the annotation and alignment points in the scene.

Now the base equation to interact with the 3D scene is known, the next stage will discuss the proposed algorithm. The proposed method uses 5 points on both the image and the model in corresponding locations. Using the corresponding reference points, the position, rotation and scale of the model can be calculated in relation to the image. Although, this method can be replicated to fit an image to the model, to allow 2D techniques to be implemented the model fits the image. The proposed method requires the points to remain corresponding to each other

---

on the image and the model, meaning if the image is not from the same time as the model, the differences in expressions can cause misalignment.

The first step of this algorithm is to align the position of the model, relative to the image. For this algorithm, the pivot point is the nose tip. The pivot point is also used to pivot the object during the rotation stage of the algorithm. To align a point from both the image and model, the following equation is implemented:

$$\hat{H} = H + ((\overrightarrow{I - C}) * (\log(I_z - C_z)^{(M_z - C_z)} + \alpha) + (\overrightarrow{C - M})) \quad (4.3)$$

where:

- $\hat{H}$  is a position where if the model  $H$  moves to this position, the image and model pivot point will be aligned.
- $H$  is a vector of the original vector position of the model we want to align
- $I$  equals a vector of an image point position
- $M$  equals a vector of a model point position
- $C$  equals a vector of the camera position
- $\alpha$  is a floating-point distance value. By increasing the value of  $\alpha$  the distance between the model and image increases, this prevents the model piercing the image. By having the alpha value, it allows objects where the pivot point is not the closest to the image to be used, for example, if the person's hair/fringe is sticking out, without an increased  $\alpha$  value, the hair would 'seep' through the image.
- $C_z$  is the z (far) position of the virtual camera
- $I_z$  is the z position of the nose point on the image
- $M_z$  is the z position of the nose point on the model

---

The first part of equation 4.3,  $(\overrightarrow{I - C})$  creates a vector from the camera to the image pivot point. Equation 4.3 then calculates a scalar using a logarithm, to move the model to the image point by adjusting the  $\overrightarrow{I - C}$  vector. The  $\alpha$  value pushes the model away from the image while following the direction of the vector, preventing the model ‘seeping’ through the image. The next section of the equation 4.3 uses the vector of the camera to the model’s pivot point  $(\overrightarrow{C - M})$ , and the model’s original position ( $H$ ), to account for the camera to image/model vectors change of a positional vector to a directional vector centred at origin/root. The resulting  $\hat{H}$  is a position along a vector of increased magnitude from the camera, which when the model  $H$  moves to this point, the model and image pivot point is aligned.

When the model and image have their positions aligned, calculation of the model’s rotation can be performed. This section of the thesis uses the condition  $\Delta < \delta$  to compare the results between iterations where:

- $\Delta$  is the previous iterations distance
- $\delta$  is the updated iteration distance

When  $\Delta < \delta$  we are moving away from alignment/convergence, and the incrementing value must change. The incrementing value is a small value represent by  $\Lambda$ . If the condition  $\Delta < \delta$  is true, then  $\Lambda$  is changed by:

$$\Lambda = -\Lambda \tag{4.4}$$

Equation 4.4 changes  $\Lambda$  between positive and negative values changing the rotation direction.

The rotation, unlike the position alignment, uses the full five points. By splitting the points into sets of three, that best represent the rotational axis. The sets used to calculate the face rotation are:

- *Yaw/Y*: Left mouth corner, nose tip and right mouth corner



- *Pitch/X*: Right outer eye corner, nose tip and right mouth corner
- *Roll/Z*: Left outer eye corner, nose tip and right mouth corner

The next stage is to project these points to the screen and retrieve the screen location of the points. Once the screen location of the points is retrieved, removal of unnecessary coordinates is made, for example in the Yaw rotation, the Y coordinated will not be useful in calculating the rotation. As illustrated in Fig 4.3 if we adjust the Y rotation of the model, the Y (up) value of the nose tip will remain constant, where the X (left) will change. For the Yaw, the X position represents the rotations effect well. Then the points are placed in order of ascending values:

$$B_1 < B_2 < B_3 \quad (4.5)$$

where:

- $B_1$  is the X coordinate of the left mouth corner
- $B_2$  is the X coordinate of the nose tip
- $B_3$  is the X coordinate of the right mouth corner

A method of representing this is that it creates a bar representing the current rotation (Fig. 4.3). By having a bar for both the image and model, when the model rotation is adjusted it affects the bar allowing testing whether it is closer to the images bar when compared to previous iterations.

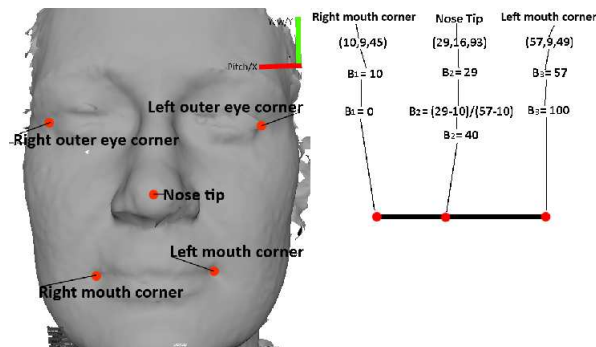


FIGURE 4.3: An example of the Yaw bar calculation.

---

$$B_1 = 0 \tag{4.6}$$

$$\hat{B}_2 = (B_2 - B_1)/(B_3 - B_1) \tag{4.7}$$

$$B_3 = 100 \tag{4.8}$$

where  $B_1$ ,  $B_2$  and  $B_3$  are equal to equation 4.5.

As the rotation of the model changes the  $\hat{B}_2$  value changes, the aim is to match the model's  $\hat{B}_2$  value with the images  $\hat{B}_2$  value. The model's rotation is adjusted by:

$$\hat{\theta} = \theta + \Lambda \tag{4.9}$$

where:

- $\theta$  is the original model's rotation on an axis
- $\hat{\theta}$  is the new model's rotation on an axis

Eq 4.9 adjusts the model's rotation  $\theta$  along an axis by the value  $\Lambda$ . However, many 3D engines and models use a different rotation axis, for example, Z up (3DSMax [90]) or Z forward (Maya). Because of this  $\Lambda$  must be reversed if the distance between the percentage of the model and the image is greater than the last distance calculation. The distance  $\delta$  is calculated using:

$$\delta = |\hat{B}_2(image) - \hat{B}_2(model)| \tag{4.10}$$

An issue during this stage is some of the available 3D tools, such as Unity [91] require the physics engine to update in the 3D scene, for rotations and translations to take effect; between the iterations, a break must be placed to allow for the scene to update or the results will remain the same which can cause errors. This process then repeats for the X and Z rotations of the model.

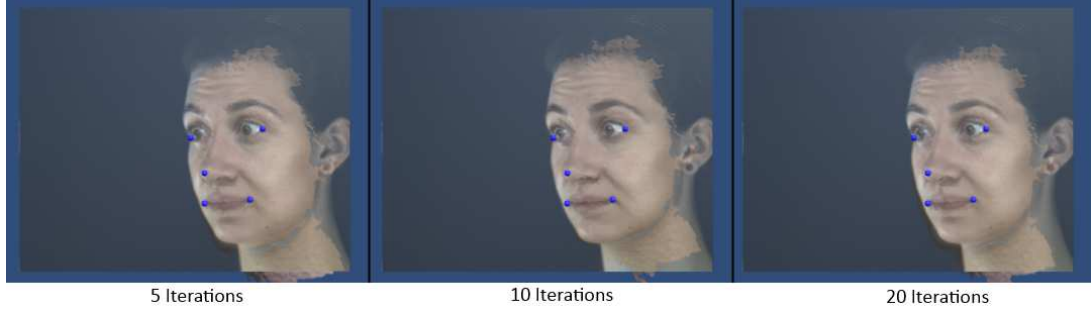


FIGURE 4.4: A comparison of the number of iterations.

The final part of alignment requires the model scaling to the image points. The positioning and rotation of the model methods are both scale-invariant, meaning the model can be any scale for alignment in those steps. However, for annotating the model it needs to be scaled to fit with the outline of the image features. Scaling the model uses the same principle as the rotation, but if the points, when projected back to the screen are higher for the model than the image, the model requires a lower scale. Similarly, if the model points are lower than the image points the model need to enlarge. The result of the higher or lower point changes  $\Lambda$ , so the model scales correctly, similar to equation 4.4. The adjustment of the scale is performed by:

$$\hat{S} = S * \Lambda \quad (4.11)$$

where  $S$  is the scale of the model.

To ensure the scaling is correct the distance is calculated twice, once for the highest correlating points and the two lowest correlating points, to attune for rotational error. The scaling is performed uniformly, meaning the size on each axis will remain the same but can be flipped either positive or negative.

Annotation of the points is performed using equation 4.2 and extending the magnitude of the vector by a large scalar, usually infinity. By using collision detection, an accurate measurement of the extended vector passes through both the image and model can be gained.

The system uses iterations to refine the alignment. Each iteration consists of a rotation of the X, Y, Z axis and a scale step.

---

In Fig. 4.4, the image and model converge/aligns at ten iterations, but the system de-aligns the image and model slightly at twenty iterations. The converging and de-alignment is a result of the system using unified scaling. However, with multiple iterations, the system will re-align/converge. By using unified scaling there is less chance of error scaling, but the slight differences between the alignment points cause the system to re-evaluate the rotation from a different angle to attempt a better fit.

### 4.2.3 Results

The annotator has been developed using the Unity3D game engine [91] in a WebGL build, allowing for the tool to work inside modern internet browsers. The proposed method is compared to Maya in three categories:

- Alignment time
- Annotation time
- The accuracy of annotations, by comparing the distance to ground truth

A visual comparison will also be shown demonstrating the difference between the results. The test was conducted by asking participants to annotate twenty-five landmarks on three models. The five points used to align are not considered annotation points, this is because the five alignment points do not have to be in fixed points, for example, mouth left corner, but in places which have clearly defined features the user can identify in both image and model. The models consisted of 3 expressions:

1. Neutral (Model 1)
2. Eyebrows raised (Model 2)
3. Mouth open (Model 3)

---

The experiment recruited sixteen volunteers with computing backgrounds with some having previous annotation experience. A certified [FACS](#) coder annotated the ground truth data. All participants were trained in both systems before the experiment began that included a test to check the participants understanding; they were also free to ask any questions. The average time for the experiment was 2 hours excluding any breaks.

The three models are used to identify the current capabilities of annotating [3D](#) under an array of different expressions, for example, are the lower face movements in model 3, tracked similarly to an upper face movement in model 2. Using the multiple expressions allows for a more comprehensive comparison, visualising annotation of the different facial landmarks. Firstly, the participants were asked to annotate the three models using the conventional method (Maya). The participants were then asked to repeat the annotations in the proposed annotator. To remove bias from encountering one set of models first, the experiment order was counterbalanced between experiments; this means half the participants used the proposed annotator first and Maya second. Participants completed both systems in one session but had breaks between each model.

For this experiment, the  $t$ -test was calculated using the average distance from the ground truth. The  $t$ -test was used due to sample size, the paired  $t$ -test compares two populations of means, with samples that are correlated. This comparison suits the purpose of the experiment as the tests provides the same data outputs. The first experiment is to determine two hypotheses; the first is to test if the distance from the ground truth between the two methods is significant. Then if the results are significant, which of the systems gave the highest accuracy.

TABLE 4.1: Table of  $t$ -test results comparing the facial landmark accuracy.

Model	$t$ -value	$p$ -value
1	-0.78	0.21
2	-2.12	<0.019
3	-0.11	0.45

---

Table 4.1 shows mixed results with each model. The mixed results could be linked to the expressions on the models, as model 1 is the baseline using a neutral face with the lips closed. Lips closed, when represented on a 3D model, creates a single surface, creating difficulty in locating the outer and inner lip corners. Whereas, in model 3 the mouth open is open creating clearly defined edges and has the least difference. The second model represents an upper facial movement (eyebrows raised), this shows a significant change implying that for this feature without texture is difficult for users to annotate accurately. This will lead to future work as, without accurate annotations of landmarks on a 3D model, the location of landmarks will not be suitable for facial animation. Currently, the manual Maya annotator provides better accuracy on the 3D face, but this could be related to user learning the alignment tool and occlusions. The overall average distance from ground truth is calculated to note the difference between the annotator and the Maya method (calculated by the in both model and annotator) shown in Table 4.2.

TABLE 4.2: Table of the overall distance in millimetres.

Software	Model 1	Model 2	Model 3
Maya	63.54	66.67	64.04
Annotator	67.76	78.73	64.70

In models 1 and 3, the discrepancy of distances is minor, but the model 2 has a large discrepancy. The second model is the only model that contains a movement hard to detect in 3D, the eyebrows raised. However, the SD in Maya remains constant, and by comparing the results of the participant annotations (Fig. 4.6) the landmarks on the eyebrows in Maya remain consistent, indicating that the participant is guessing the facial landmarks.

The timing is a crucial part of the annotation because if the system takes longer to align the image with the model before the annotations begin, then just annotating in Maya, the system will not be preferred as Maya already requires

---

significant time to annotate. Table 4.4 shows the mean and SD times, for both the alignment and annotation of models in the proposed method. For evaluation, the one-tailed  $t$ -test with a  $p$ -value of 0.5 was performed. The tests are to evaluate the hypothesis that the two systems are not equal in the time taken to annotate and the accuracy of annotations.

TABLE 4.3: Table of the  $t$ -test results for the time comparison.

Model	$t$ value	$p$ -value
1	-8.74	< 0.00001
2	-8.54	< 0.00001
3	-9.10	< 0.00001

Table 4.3 shows that the annotator and the conventional Maya method are not equal, by having a significant difference between the times required to annotate. As the value of  $p$  is low, it demonstrates a high significance, which is a result of the reduced time required by the proposed annotator.

Fig. 4.5 compares the overall times between the traditional method (Maya) and the proposed method. The time comparison took into consideration the time to align and annotate a 3D model. As the proposed method has two sections, alignment and annotation the times were recorded separately. Fig. 6 for the proposed method shows the full times (alignment and annotation).

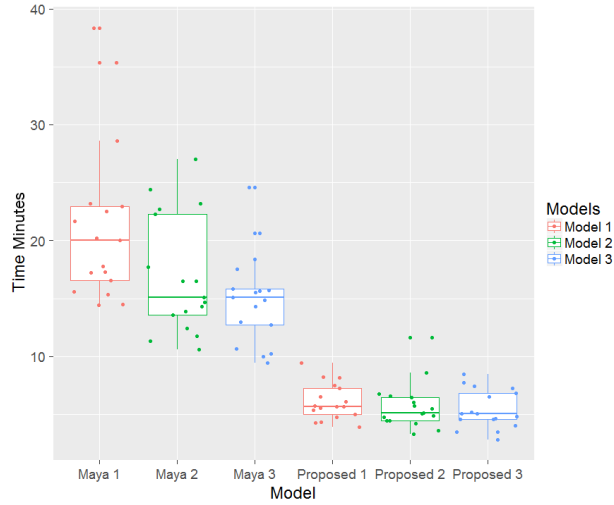


FIGURE 4.5: The total Maya times (left) compared to the proposed method (right)

TABLE 4.4: Table of the proposed methods average alignment and annotation times (minutes) with [SD](#).

Model	Alignment	Alignment <a href="#">SD</a>	Annotations	Annotation <a href="#">SD</a>
1	2.95	0.46	3.13	1.50
2	3.53	1.61	2.18	0.86
3	3.61	1.56	1.48	0.98

TABLE 4.5: Table of Mayas average alignment times with [SD](#).

Model	Annotation	Annotation <a href="#">SD</a>
1	21.26	6.98
2	16.98	5.05
3	15.06	4.00

Table 4.4 shows that a neutral face is, on average easier to annotate the alignment points, meaning the system can align the image and model when compared with models performing expressions. The time difference between expressions indicates that annotating the [3D](#) models is a much more difficult task without texture





FIGURE 4.7: The points placed back on the model similar to Fig. 4.6, but aligned with an image.

that uses expressions. The difference in time gives a path for the future work to provide an intelligent system, that can aid the user in aligning models that use expressions and to reduce the overall alignment time and alignment errors.

Next, a visual comparison of the results is presented, to demonstrate the accuracy of 3D points in relation to the face on a 2D image. For the comparison all the collected points were placed back onto the 3D model and then aligned with an image.

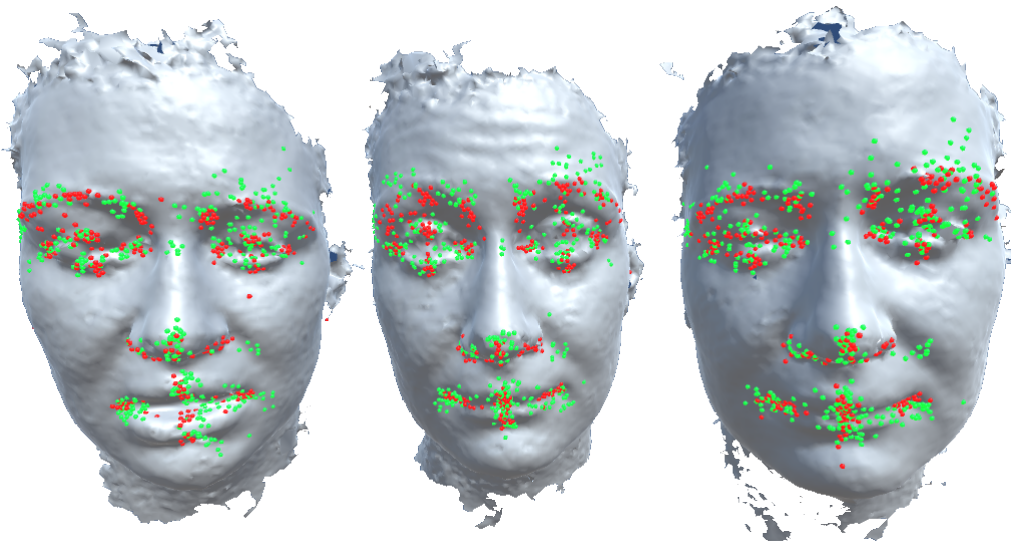


FIGURE 4.6: The results of the current tests the green points are the proposed annotator and red points, Maya.

On inspection of Fig. 4.6, areas where the face curvature changes significantly have the highest accuracy in Maya. Areas of high curvature change are face sections, such as the nose tip and the eye ridge.

---

#### 4.2.4 Discussion

Fig. 4.7 compares Maya and our proposed method annotation points when aligned to an image. The alignment gives a visual comparison between the two systems. The alignment demonstrates that Maya does not locate all the landmarks correctly. The eyebrow edge is being annotated to close to the inner face, and in the eyebrows raised, the points are placed lower down. The errors locating the landmarks in Maya would prevent a system trained on this data to identify the eyebrows successfully.

Overall, the proposed annotator performs well on the side of the face which is most visible, but the occluded side of the face has worse groupings. A solution to occlusions is to realign with an image from another angle, but the tests were performed on the D3DFACS dataset [1], where only one side-on image is available. The D3DFACS dataset was used as the models contain little noise and is a high-resolution model that gives a clear view of the facial landmarks when compared to Kinect models where features are hidden by the noise. The D3DFACS removes bias as it gives models which user have the best chance of identifying features in Maya. However, the D3DFACS dataset provides a challenge for the proposed method owing to the extreme rotation of the face, shown in the reference images, Fig. 4.7.

This section of the thesis demonstrated a proposed novel method of aligning a 3D model to a 2D reference image. By aligning the model to a reference image, the issue of identifying facial landmarks on a 3D model is resolved. The proposed method is designed to reduce the total workflow of the annotator and enable the use of tried and tested methods of annotating 2D images. This system highlights a commonly overlooked issue of annotation, which has a significant impact on facial tracking in 3D and proposes a solution to this issue.

---

## 4.3 Kinect One Expression Dataset (KOED)

This section of the thesis we analyse the in-house dataset made for the work. The dataset overcomes a series of issues, such as the synchronous capture of both depth and RGB. This Dataset is used in chapters 5, for the synchronous RGBD capture, and 7 for the expressions. The dataset resolves a number of limitations of other datasets by focusing on key aspects:

- Accurate ground truth landmarks
- Full recordings of the seven-universal expression with onset and offset
- Synchronous capture of RGB and depth data
- Controlled capture environment with ground truth colour and measurement info in the scene

### 4.3.1 Experimental Protocol

For each participant, we followed a strict recording routine. We ensure that all the ethics and consent forms were signed before beginning any recording. Each participant was recorded 8 times, 16 for females with makeup on and off. Each recording is of the participant starting in neutral, performing expression and back to the neutral pose. The seven-expression captured are:

- Happy
- Sad
- Surprise
- Anger
- Fear
- Contempt

- 
- Disgust
  - Neutral (full clip)

each clip is 5 seconds long at 30 FPS giving a total of 180 frames per expression clip.

#### **4.3.1.1 Emotional Replication Training**

During each recording, a trained individual was present to advise of the participant's expressions and give some prior training. However, during the recording, the trainer would not give any advice to prevent distracting the participant.

#### **4.3.1.2 Ethics**

Ethics was reviewed and approved by the Manchester Metropolitan University ethics committee (SE151621).

#### **4.3.1.3 Equipment and Experimental Set-Up**

The experiment required some equipment to record, such as:

- Green screen room access (with lighting)
- Kinect One with a tripod
- SSD spec machine for recording Kinect data
- Chair
- Colour Checker with stand

For the recording, the participant sat 1 meter away from the Kinect One, with the Kinect positioned at the participant's head height. The participant then has the

---

colour checker placed to their left, also at head height and in the frame. The six main lights were split into three groups directed at the participant from their left, front and right. All cameras used the same lighting condition to ensure consistent lighting across the face.

The Kinect requires no prior calibration, as we use the default factory settings. We record at the Kinect maximum capabilities, Red Green Blue Alpha (RGBA) 1920\*1080 @ 30FPS and depth 512\*424 @ 30FPS, for speed we save both files in binary format. We use six directional lights focusing on the individual participant. The lights are set to emit white light only to prevent any discolouring of the participant's faces.

### 4.3.2 Dataset Ground Truth Acquisition

To crop faces from the recording we implement Dlib [71], to ensure consistent bounding boxes. To ensure the accuracy of the system we evaluate the performance on a subset of our KOED dataset using the methods found in [92]. The system has been evaluated on multiple recordings. The recordings were performed by having a participant sit in front of the Kinect and during the recording moving slightly to ensure the tracker is following the face. To ensure the system can still function when a user is not present a recordings containing no faces, was presented to the system. The empty recording will determine if the system will give false positive values. The scoring was performed using these rules:

- **True positive:** A true positive is define by a face detection by the system, which outlines a face present in the image.
- **False positive:** A false positive is defined by a face detection by the system, but the outline does not cover a face in the image.
- **True negative:** A true negative is defined by the system not detecting a face, when there is no face present in the image.

TABLE 4.7: Table of the overall performance

Result	Score (percentage)	Meaning
Recall	0.99	How often the system has predicted true correctly over the available data
Precision	0.96	If the system predicts true, how often is this correct
Accuracy	0.97	The classifiers overall correct classifications for the sample data
Inverse recall	0.88	How often the system has predicted false correctly over the available data
Inverse precision	0.99	If the system predicts false, how often is this correct

- **False negative:** A false negative is defined by the system not detecting a face, when one is present in the image.

TABLE 4.6: Truth table of the Dlib face detection

		Predicted	Predicted
	Total frames = 1616	Positive	Negative
Actual	True	1244	44
Actual	False	1	328

Using the values produced in table 4.6, the performance measurements of Dlib can be calculated, such as the recall. The system shows a high rate of correct detections, while maintaining low false detections, but has failed to detect the face in some of the images. To further understand these values, they have been used to calculate the results in table 4.7. Table 4.7 shows that the Dlib face detector, from the tests conducted, can perform accurate facial detection with low false detections. By using the face detector in combination with Dlibs face tracking allows for accurate face shape detection. Table 4.7 shows that Dlib provides an excellent method of tracking the face in 2D with a low false detections. However, these results are preliminary and comparison against other existing methods, such

---

as EMGUs [93] face detection algorithm using HAAR [22] cascades should be performed.

### 4.3.3 Dataset Analysis

#### 4.3.3.1 Demographic Breakdown

The dataset has 35 participants, with a wide range of ages from 19 - 76. The dataset manages to have a good male female split, but has a majority of white British as shown in Table 4.8. However, it does include residents of Saudi Arabia, India, Malaysia, China and Philippine. Collection of the dataset is still ongoing.

TABLE 4.8: Table showing break down of overall participants

Participants	Age Mean	Age SD	Percentage Male	Percentage Female
35	37.8	14.8	51%	49%

#### 4.3.3.2 Comparison with Current Datasets

As the experiment required RGB and depth data from the same synchronous capture for the merging network and to prevent bias between the RGB only and depth only networks, a review of the available datasets was performed. As the results of the neural network are to produce the landmark locations in 2D and 3D the depth data should be captured from a location relative to the camera, meaning that most of the features will be in the same capture. Because of requiring the features to match datasets that use devices similar to the Kinect are required as it uses forward facing sensors that are only a few millimetres apart, resulting in similar data outputs. A table showing the different attributes of the datasets is shown in Table 4.9. The datasets analysed for this experiment are shown and described:

- 
- *Face Warehouse*: [94] is a large-scale dataset containing 150 participants with an age range of 7-80. The dataset contains RGB images  $640 \times 480$ , depth maps and 3D models. 74 2D landmarks are given, but only for the 2D colour images. The dataset only focuses on the posed expressions giving one model and image when the participant displays the expression. Furthermore, for capture, they use the Kinect version 1 [86]. The dataset also suffers as it is captured under different lighting and in different places. As only the expressions peak is captured, there is not a significant amount of data for training deep learning, and it is at a low resolution compared to modern cameras. Overall, the Face Warehouse is a good 3D face dataset providing a wide assortment of expressions with landmark annotations, but with no onset or offset of the expression.
  - *Biwi Kinect Head Pose*: [65] is a small scale Kinect version 1 dataset containing twenty participants, four of the participants were recorded twice. During the recording, keeping a neutral face, the participants would look around the room only moving their heads. The recordings are of different lengths. The depth data has been pre-processed to remove the background and all non-face sections. The recording contains no facial landmarks, but the centre of the head and rotation is noted per frame. Although the recording was done in the same environment, the participants can be positioned in different sections of the room changing the background, but the lighting remains consistent. Overall, the Biwi Kinect dataset was not suitable for the experiment as it contained no facial expressions and was recorded using the Kinect version 1.
  - *Eurocom Kinect*: [95] is a medium-sized dataset containing 52 participants; each participant was recorded twice with around two weeks in between. Participants were recorded by having single images of them performing nine different expressions. The images were taken using the Kinect version 1 and have had the heads cropped out of the images. The coordinates for the cropping are given as well as six facial landmarks. The Eurocom dataset contains



---

a few images for a deep learning network and is recorded with the Kinect version 1 making it unsuitable for the experiment.

- *VAP face database*: [96] is a small size dataset containing 31 participants. The dataset was recorded using the updated Kinect version 1 for Windows, this version gives a bigger RGB image ( $1280 \times 1024$ ) and larger depth map ( $640 \times 480$ ), but at the cost of reduced frame rates. The recording was also done using the Kinects ‘near-mode’ which allows for the increase resolution described. Each participant has 51 images of the faces taken at different head angles performing a neutral face and some frontal face with expressions. The recordings were done in the same place with consistent lighting. As the dataset contains single images and few participants performing facial expressions, it is unsuitable for the experiment, but for head pose estimation would be appropriate.
- *3D Mask Attack*: [97] is a small to medium-scale dataset containing 17 participants, but a large collection of recordings. The participant is recorded in three different sessions; in each session, the participant is recorded five times for 300 frames per recording holding a neutral expression. The recording uses the Kinect version 1 at  $640 \times 480$  for both RGB and depth images. The eyes are annotated every 60 frames with interpolation for the other frames. The recordings were done under consistent lighting and background. The 3D Mask Attack dataset contains a vast number of frames, but all use the neutral expression, face the camera and use the older Kinect making it unsuitable for the experiment.

As shown, the currently available datasets are not suitable for the experiment, for deep learning many images are required, that contains facial expressions. The version of the Kinect used is the older Kinect that has lower resolution and slow capture rate. Many researchers have used the Kinect version 1, and extensive analysis of its accuracy [98] has been performed. However, though it is used less, the Kinect version 2 offers improved image quality for RGB ( $1920 \times 1080$  compared to version 1 of  $640 \times 480$ ) and improved accuracy for depth information and higher

---

resolution ( $512 \times 424$  compared to version 1 of  $320 \times 240$ ). The Kinect 2 is less used in research, but its accuracy improvements are recognised, because of its accuracy and HD image quality, it is more suited to this experiment. For a full performance evaluation between the Kinect 1 and 2, the reader is recommended to see [99, 100]. For a review of other available 3D datasets, not discuss in this section, the reader is advised to read [101].

TABLE 4.9: Table Comparing available datasets

Dataset	Data	Capture	Participants	Age Range
KOED	RGB, Depth, Model	Time-series	35	19-76
D3DFACS	RGB, Model	Time-series	10	NA
Face Warehouse	RGB, Model	Peak	150	7-80
Biwi Kinect	RGB, Depth, Model	Peak	20	NA
Eurocom Kinect	RGB, Depth, Model	Peak	52	NA
VAP face dataset	RGB, Depth, Model	Peak	31	NA
3D Mask Attack	RGB, Depth, Model	NA	17	NA

## 4.4 Summary

The Chapter has shown the current methods of acquiring 3D facial landmarks for use as ground truth. We extended the work by producing an in-house annotation tool and testing against the traditional methods. For our in-house technique, we use a novel alignment algorithm to align 3D geometry in a virtual environment. Then we discussed available dataset for our project; we noted that the dataset in vastly different in each one and few are at a large enough scale for training effective neural networks. Thus we produce our dataset. The in-house dataset overcomes many of the limitations of the other available datasets, such as synchronous full expression recording on a broad demographic of individuals.

# Chapter 5

## Real-Time Facial Landmarking using Neural Networks

*This chapter shows the deep neural networks approaches for 2D and 3D facial landmarking. In 2D, the accuracy, as well as the implementation on mobile devices, are demonstrated. In 3D, the accuracy and effect of auxiliary information on performance are measured.*

### 5.1 Introduction

Motion capture in commercial use has two main categories. For consumer based applications the use of traditional machine learning for marker-less landmarking is implemented. However, in recent years deep learning with CNNs has shown to outperform traditional machine learning significantly [102, 103]. The accuracy improvements do come at a cost; the machine defined features require a large scale dataset for training to ensure the weights of the network are generalisable to a wide demographic. Additionally, the storage of a deep learning network can be significantly higher than a machine learning model. Finally, when training a model on high-end machines with large capacity Graphical Processing Unit (GPU)s is required, which creates an interpretation that a model can only run on similar

---

machines. However, in reality, a trained model can process single images on a low power device can perform adequately. In deep learning, the use of pooling has many advantages it focuses on key values and reduces the input size, to aid in over-fitting prevention. Furthermore, the ability of pooling to focus on high values, while reduces image size significantly reduces the processing requirements of the network, and both the Random Access Memory (RAM) and storage memory requirements. We perform an analyse on how effective mobile devices are for facial landmarking with CNNs and the effectiveness of max-pooling to allow more efficient processing.

With the availability of RGBD sensors, the potential to increase accuracy is possible by merging the data streams within a neural network. Merging data streams, allows a marker-less system to predict depth without the requirement of multiple cameras with high accuracy. The depth information assists significantly in identifying facial feature movement and synthesising to 3D models.

An important factor is that although depth provides additional information, in object recognition  $G_s$  outperforms RGB data significantly [104], showing more information may not necessarily produce a better system. Thus we also compare against  $G_s$  image and merged Greyscale Depth ( $G_sD$ ). This study improves upon current work, as the literature is split between networks that use full RGB [41, 43] and networks that run  $G_s$  [35, 42] without justification, and focuses solely on 2D landmarks prediction. 3D landmarks are essential for face recognition in the presence of expressions [105] and real-time facial animation [78]. To do this, we expand the existing work to predict 3D landmarks and investigate the impact of 2D and 3D data when used as auxiliary information.

As with many fields of research, the implementation of deep learning has shown significant improvements in the area of facial landmarking [106], when compared to traditional machine learning [45]. In this work, we focus on the use of CNN, similar to the literature in this area. To perform the experimentation, we develop near identical networks to reduce the deviation between results. In this chapter, we target the issue of facial landmarking in the two categories shown, the

---

2D techniques and demonstrate that consumer-based devices can effectively run a landmarking system. We then extend the work and refine the networks to produce 3D landmark prediction networks and determine the effective streams.

This section of the thesis demonstrates to key attributes of the work. Firstly, an analysis of state of the art landmarking neural network networks is performed. The networks are then an neural network is designed, with variation that differ in the amount of pooling layers. The networks are ported to mobile phones to evaluate their performance in FPS, the accuracy and loss of training is recorded. Secondly, using the information gathered from the first stage a neural network is designed. Again, slight variations of the network is designed, but to analyse the affect of how different or multiple input streams affect the networks performance, such as is RGB or  $G_s$  better for landmarking. The comparison of stream allows deeper understanding of networks interpretation of data, as RGB and  $G_s$  are easily accessible to most devices and depth data is available on some, devices could see significant improvement because of this. Additional networks are created with different outputs, such as 2D and 3D landmarks. The additional outputs is to test the ability of auxiliary information, to improve the landmarking performance, as both tasks are similar.

## 5.2 Related Works

The related work divides into three sections. Firstly, we give an overview of the current state-of-the-art deep learning to predict facial landmarks. We demonstrate the critical aspects of the networks functionality and the features used to localise landmark regions. The second section focus on the compression and optimisation of the neural network on mobile devices. Lastly, we evaluate merging  $G_s$ /RGB and depth information in a neural network and the current implementation methods.

---

### 5.2.1 Facial Landmarking with Neural Networks

Facial landmarking in deep learning is well established, with state of the art showing both real-time and high accuracy results. Neural networks have solved a wide range of problems, such as facial landmarking, age identification and gender classification. Due to the adaptability of neural networks, previous literature has evolved to use multi-output networks [40, 107]. Multi-output networks perform an array of predictions simultaneously, such as age and gender. For our review, we focus on both single and multi-output networks, such as landmark and gender [43] and landmarking only networks. We discuss multiple output networks as they can outperform landmarking only networks as research shows that auxiliary features have a positive effect on network performance [63]. Auxiliary features, boost the performance of the network by adding critical pieces of information. For example, in age prediction, if using gender as an auxiliary feature, it aids the network as it learns how the make-up and facial hair affect age prediction. The network predicts auxiliary information in addition to other outputs, the input to the networks is still a single or merged stream of data. Although our experiment is to see the effect of different streams of data, on a neural network, for the area of facial landmarking, auxiliary features, such as age and gender would be an aspect of future work.

We first evaluate networks that focus solely on the prediction of landmarks. In 2013, Sun et al. [34] proposed an end-to-end network that takes a facial image through a series of convolutions, max-pooling and fully connected layers, to predict five facial landmarks with reasonable accuracy. Zhou et al. [35] expanded on the work, by proposing a series of detectors to identify facial regions and process them by small neural networks. They also use a refinement approach that aligns the facial features before landmark prediction. Lia et al. [37] proposed a complex network for landmark detection where they implemented a two-stage network, the first stage is a series of convolution and deconvolution layers to process the image given into a high-value feature set. The features were then processed by a series of LSTM [38] layers to identify and refine the landmark position. Recently, Liu et al. [36] used a multitude of facial feature detectors for localising regions, such as

---

the eyes, nose and mouth. The authors processed these regions with small sized neural networks that identify the landmarks on each of the features. This method achieves high accuracy results, as the network and detectors specialise in different aspects of the face, instead of trying to generalise to all of the unique features. However, unlike Zhou et al. [35], they did not align the features.

We now review the work that uses multiple output networks. Zhang et al. [40] experimented in the use of auxiliary features to increase a network understanding of facial structure and features. They created multiple networks with the structure remaining the same except for the outputs changing by adding critical pieces of information such as facial direction, age and gender. By incorporating auxiliary features, networks learned facial features in more depth. The authors showed a significant increase in accuracy when asking the network to determine these extra features, even when training the network to perform normally difficult tasks, such as facial direction. More recently, Zhang et al. [63] extended their work on facial alignment. Jourabloo et al. [42] used a similar method to predict landmarks by having a series of networks to refine the positions. However, they focused on using the landmarks to refine the appearance of a 3D model. Even though Zhang et al. [63] and Jourabloo et al. [42] provide high accuracy networks, the networks require pre-processing to crop faces out of the image.

Finally, we review all-in-one networks, requiring no pre-processing before network prediction. The most recent research for facial landmarking focused on end-to-end networks based on Recurrent Neural Network (RNN) [108]. Zhang et al. [41] present an all-in-one neural network to identify and landmark faces in an image. They used three interlinked networks to refine the landmarking approach. The result of the network is five facial landmarks and bounding box for every face in an image. On the other hand, Ranjan et al. [43] produced their all-in-one network to retrieve the face bounding box, landmark, facial direction and gender with high accuracy. The network included a separate classifier to check if the first section of the network returned an actual face.

---

The networks when trained of the separate streams of data, give high-end accuracy results starting from the small-scale one output networks to complex multi-model methods. However, the work is limited as it only considers single **RGB** or **G<sub>s</sub>** images to predict **2D** landmarks. Whereas, state of the art uses multiple cameras or depth data to estimate the desired **3D** landmarks. Additionally, the literature does not give justification for the use of either **RGB** or **G<sub>s</sub>**. As neural networks are adaptable, we want to investigate how the different streams of data affect a neural networks ability to predict both **2D** and **3D** landmarks. Furthermore, we extend this by analysing the effect of merging multiple data streams for accurate facial landmark prediction, such as integrating both **RGB** or **G<sub>s</sub>** with depth. We also extend on Zhou et al. [35] work by analysing the effect of using UV and XYZ as auxiliary features, compared to UV or XYZ only to train a model that understands facial structure in detail.

Investigation on the use of depth information to predict facial landmarking has been performed [109]. However, much of the focus is on using surface curvature analysis. Curvature analysis does give reasonable results on low noise models, but it is a slow process and can only track a few points in areas of high curvature change. Another method of predicting **3D** facial landmarks is shown by Nair et al. [110], who impressively have predicted a total of 49 landmarks on the face, but avoids the mouth area. However, this method required a generated **3D** model as point distributed model is used to deform a template face with landmarks assigned to the new mesh. Deforming a **3D** mesh to a template is an intense and computationally expensive task. Both methods required pre-generated models that is difficult at real-time on a consumer base, our focus in the sole use of images to accurately infer the landmarks.

## 5.2.2 Neural Network Compression

There are many different options to make a neural network more efficient, such as using smaller kernels in the convolutions and output fewer convolution images, fewer filters, different output images are produced for each filter requiring more



---

memory, at each of the layers. However, another method to reduce the strain of the network is to implement max-pooling layers [76] into the network. The max-pooling layer split the images into a number of regions, from each region the highest value is taken and placed into a new matrix, illustrated in Fig 3.6. By reducing the feature maps, max-pooling also helps with over-fitting as the data becomes more abstract, as well as reducing the processing requirements. The issue with implementing max-pooling layers is data loss by shrinking the data, possibly reducing the accuracy, but reducing processing time. For facial landmarking, the data should be preserved as much as possible, as it is easy to lose vital facial features, such as the eye corners and nose tip when employing max-pooling layers. Implementing max-pooling allows networks to run much faster on lower power devices. The max-pooling acts as a cost/reward system which will be analysed in this section of the thesis on the trade-off between speed and accuracy

Neural networks can be of varying sizes and scales. The types of layers can drastically increase the amount of processing required to run the network, such as the Inception [111] and LSTM [38] when compared to a convolution layer. This section will describe the work done to compress and analyse neural network performance on mobile devices.

The most commonly used methods to compress a neural network is the hashing trick or HashedNets by Chen et al. [112]. Hashed networks function by grouping random connection weights into a single ‘bucket’; the connection weights are all tuned by one parameter reducing the networks total size and memory requirements. However, joining multiple weight values into one can cause the network to become less diverse, meaning the networks accuracy can be negatively affected. Another method called One-Shot Whole Compression by Kim et al. [113] focuses on shrinking the entire convolution based network. Kim et al. [113] split the compression into three main stages:

- Identification. Using Bayesian matrix factorisation, the sections of the neural network, which contributes most to the success of the network. By identifying the highest contribution sections, the method can better preserve its

---

accuracy.

- **Reduction.** The method can apply one of two versions of Tuckers decomposition, using the ranking from the Bayesian matrix factorisation for decision making, the first analyses the core components and then merge them to a single tensor. Whereas, the second performs single value decomposition.
- **Fine Tuning.** As the method modifies the network, the results produced can differ significantly. Kim et al. [113] retrain the network with pre-built models to aid in accuracy recovery.

Overall, Kim et al. [113] method allow for substantial decreases in file size with only small losses in accuracy. Research has also been done to see how different layers, such as recurrent layers can be used to shrink the network size [114].

### 5.2.3 Merging Visual and Depth

A multi-model network [115] for the merging of data, such as  $G_s$  and depth, usually implements three separate networks that work together. The first two networks take input from the separate streams of data; then they can be processed the same way as a traditional CNN. The network uses these convolutions to extract the unique features in each of the data streams. After the processing, the input for unique features the outputs of the network can be fed into the third neural network and the data merged using basic matrix operations. The third network, similarly to the first two networks, functions as a traditional convolution network.

Merging separate streams of data is, in some areas, a common practice, such as action recognition [116]. Berg et al. [116] show that by merging an RGB stream with its optical flow counterpart in a neural network, significantly improves the networks accuracy, by segmenting out the motion in action recognition.

Merging different data streams have also shown increased accuracy in object recognition [117, 118]. Socher et al. [117] use a single layer CNN to retrieve RGB and depth images to extract low-level features. The output of these networks is fed

---

into separate [RNNs](#). The results of both [RNNs](#) are fed into a softmax classifier. By combining the data Socher et al. [117] shows significant improvement in object recognition. The research in this field mainly follows Socher et al. and Berg et al. on merging to increase the accuracy of detection and recognition systems.

For our experiment, we are trying to solve a different type of problem where detection and recognition system use classification, landmarking is a regression-based problem. Applying classification to a landmarking problem would mean assigning a true or false value for every pixel in an image, which would be too processor intense for real-time performance. Whereas, regression allows a single output to be a wide range of values, significantly reducing the processing requirements. We implement the in-house dataset described in chapter 4 of the thesis, as for training synchronous captured [RGBD](#) data is required.

## 5.3 Proposed Methods

The methodology splits into two sections; The first section focuses on [2D](#) facial landmarking for mobile applications. The second section focuses on facial landmarking methods for [3D](#) depth data.

### 5.3.1 Facial Landmarking for Mobile Applications

This section will demonstrate and analyse the networks designed for this experiment and give details into the reasoning behind them. After this, a description of the dataset and the elaboration of the pre-processing steps.

Three basic neural networks have been defined, to test the processing capability of the devices when implementing neural networks, each with similar layouts designed to maximise the efficiency of the network. Each of the networks follows the same structure as the Base network, as the purpose of this experiment is to discriminate how the max-pooling layer affects the applications capabilities, a simple neural network (henceforth, basic network) is chosen. The basic network consists

---

of three layers: convolutions, max-pooling and activation, followed by two fully connected layers with intermediate activation layers and the final output layer. The initial convolution layer uses a  $3 \times 3$  kernel, whereas the following convolutions used a  $2 \times 2$  kernel. The first two convolutions used 32 filters on each image, and the last convolution uses 64. Filters are used to widen the neural network as illustrated in Fig 3.4; they work by creating different convolution kernels for using on the input image. Having multiple filters aid the network, as different filters focus on the type of variation the network could encounter, such as a frontal face filter or side face filter. By having filters account for variation more reliable network can be produced. The max-pooling layers were fixed to  $2 \times 2$ , effectively shrinking the convoluted images by half as shown in Fig 3.6. The initial input size of the image was  $96 \times 96$  meaning the final image size for the fully connected layer was  $12 \times 12$  which equals to the required output of 136 ( $68 \text{ points} \times 2$  for the X and Y). The two activation layers use 1000 and 500 connected neurons before connecting to the output layer. The goal of these networks is to give a clear view of the cost/reward for implementing max-pooling layer to reduce the workload of the network and how that impacts the networks accuracy. The max-pooling layers are removed starting with the first, as the longer, the network maintains the full data, the more it can learn. The seed for generating the neurons initial weighting was fixed to 7 to improve reproducibility. Table 5.1 shows the number of max-pooling layers in each model.

A network with no max-pooling was not considered due to the memory limitations. The memory requirements for training could be improved by reducing the batch size. However, this would have a detrimental effect on the accuracy and increase training times. Deciding the correct batch size, is a complicated process, regarding requirements, as each filter produces an additional image for each layer, so from a single input image, 65536 images of  $96 \times 96$  are produced ( $1 \times 32 \times 32 \times 64$ ), this would require 604MB, much higher than what current phones have. For the training stage, even the Titan Pascal (12GB) card with reasonable batch sizes was not possible. The networks train for 300 epochs, this is because of data retention through the removal of max-pooling layers, the longer the networks need

---

to converge, and this gave the fairest option for comparison, as this shows when convergence occurs.

TABLE 5.1: A comparison of the accuracy and loss for three types of network.

Network	Max-Pooling layers	Accuracy	Loss
Basic	3	0.63	0.17
Two max-poolings	2	0.64	0.12
One max-pooling	1	0.69	0.38

For each of the networks, [ReLU](#) was implemented as well as root mean squared error (eq. [5.1](#)) for loss calculation equation:

$$RMSE = \sqrt{\sum_{i=0}^n \frac{(y_i - y'_i)^2}{n}} \quad (5.1)$$

where:

- $n$  is the number of samples in the training batches.
- $y_i$  is the ground truth output for the training image.
- $y'_i$  is the predicted output for the training image.

These methods prevent any ‘squashing’ of the data between 0 - 1, which prevents a thoroughly regressive learning system. The networks implement the Adam optimiser [[119](#)] as it has been shown to perform well in this research area.

The accuracy for the network was calculated using binary accuracy (eq. [5.2](#)), as with landmarking the chances of early networks getting an exact hit of the landmarks is slim. As a result, the distance to the point is required, and this gives a more explicit indication of how well the trained model is performing.

$$Accuracy = \frac{\sum_{i=0}^n y_i}{n} \quad (5.2)$$

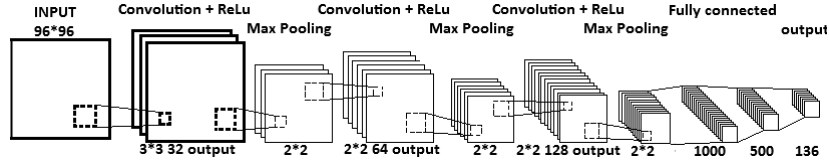


FIGURE 5.1: A visualisation of the basic network used for porting to mobile.

where:

- $n$  is the number of samples in the training batches.
- $y_i$  is the ground truth output for the training image, where if the out was correct is 1 else is set to zero.
- $[y'_i]$  is the rounded predicted output of the training image.

The networks were designed and trained using Tensorflow [120]. Thus the tensors were saved in graph form with both weights and structure in the same file. Optimization was then performed to remove the training, testing and none require variables from the graph. The networks trained using the GTX 1080Ti GPU (11GB) with a batch size of 120. The batch size was determined by modifying the Basic network multiple times and comparing the accuracy which has a high impact. All networks were trained with the same batch size and the random weights initialiser seed was set to 7 to increase reproducibility and consistency. We also split 10% from the training set to be used for validation. The Keras interface performed the data chosen for validation.

In this section, we review the methodology behind the training of the neural network and the preparation of data. The network was trained on the Morph dataset [121] using 49828 out of the 55134 available within the dataset, images were excluded if they had incorrect lighting, deformed or unrecognisable, had most of the face obscured or failed in both the OpenCV and Dlib [71] face detectors, as both are commonly used face detectors. Images that met the required standard were passed to the next stage of annotation. An automatic approach was taken, similar to [122, 94], was used to speed up the landmark annotation and maintain consistency, which shows to boost system accuracy [84]. The dataset was loaded

---

into a created program that would first transform all the images into grayscale; the faces were then cropped out using Dlibs face detector and resized to  $96 \times 96$  images. The images were passed into Dlibs face point detector to retrieve 68 facial landmarks. The points and  $96 \times 96$  images were saved into a Comma Separated Value (CSV) file as input data for training. The format for the CSV file was the face point  $x$  coordinate, face point  $y$  coordinate for all 68 points on the face and the last cell was the raw  $96 \times 96$  pixel information with a space between each pixel value. The morph dataset consists of the frontal face only and with limited expressions.

The neural networks were trained to take in a  $96 \times 96$  grayscale face image and process to retrieve the facial landmarks. To retrieve the facial image; the OpenCV library [123] was used as it has high accuracy [124], the library is compatible with mobile platforms making it suitable for this experiment. The OpenCV library uses a Viola and Jones [22] style cascade classifier to detect the faces. To prepare the images for the network, the images were first converted to  $G_s$  from  $RGBA$  to reduce the processing time and also resized to  $96 \times 96$ . The network then processed the resulting image after cropping and rescaling. The process of preparing data was integrated into the mobile application using the phones backwards facing camera (faces away from the user's face) as the live image feed allowing for the real-time cropping of the face and landmarking.

Neural Networks have seen an increase in use in recent years [125, 126], but focusing on how accurate and effective the networks were. Whereas, another primary focus should be the implementation of neural networks into consumer and industry-based systems. As neural networks are showing increase accuracy and reliability, it shows that many current consumer-based applications could be improved by using these techniques. It is possible once the model is trained, to run the model on devices of significantly less processing power as the removal of the backpropagation and training layers allow faster computation. However, the memory for processing the image is still required which is still a significant amount for a mobile device.

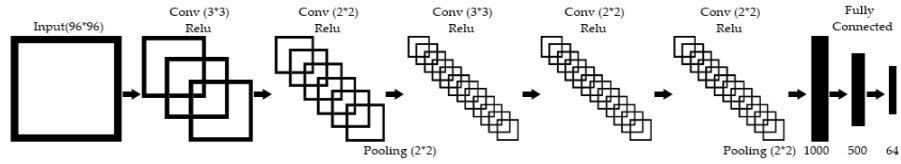


FIGURE 5.2: A visualisation of the basic network used for the auxiliary information experiment. The networks was adapted from 5.1, having fewer pooling layer to preserve the data.

Neural networks are usually stored as two separate files. The first file is the model, the layout and structure of the networks. The second file is the weights of the networks, which is larger size and controls how the network processes the data. For porting to a mobile device, merging the weights and network structure files is required saving loading times and memory. To run a trained neural network, the device has to be able to load the file into memory, which is an issue on mobile devices as even the most modern mobile devices contain little RAM, such as the iPhone 7 which has 2GB. Also, Tensorflow can have issues with file sizes over 68mb, due to android compression. To load the model onto the device, Tensorflow has a mobile-specific library [120] for Android; this allows the device to open a tensor graph containing the model and weights.

Android Studio [127] was used to build an Android Package (APK) with a build targeting the API level 25 on x86 and x64 CPU architecture; this was to ensure compatibility with the Tensorflow library and OpenCV. By using the Tensorflow library, the model can be loaded onto the device if sufficient memory is available. OpenCV is then used to access the mobile camera to detect faces and pre-process the images for the neural network.

### 5.3.2 3D landmarking with auxiliary information

We implement multiple near-identical networks that function by pre-processing the image with convolutions with ReLU activations and then a series of fully connected layers to determine the final output. We illustrate the base networks in Fig 5.2. The base networks take a single stream of data,  $G_s$ , RGB or depth and



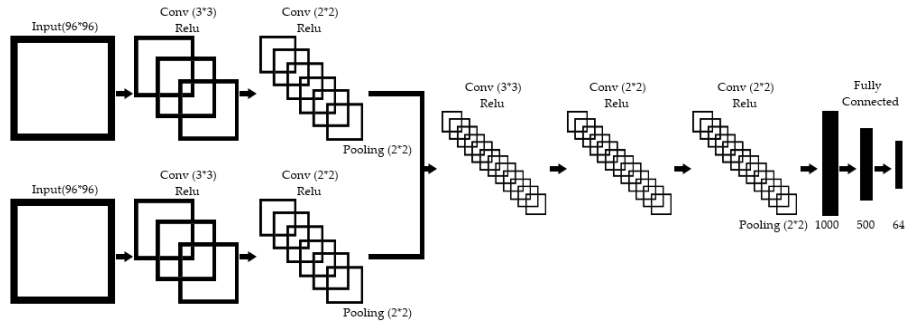


FIGURE 5.3: A visualisation of the merged network used for this experiment

process through a series of convolutions to extract facial features. We use max-pooling to focus on high-level features and decrease processing requirements, but take into consideration this can negatively impact accuracy [128]. The network utilises [ReLU](#) as an activation function after each convolutional layer as it does not normalise data. The resulting feature maps are then processed by fully connected layers to predict the facial landmarks. For the second stage, we examine the effectiveness of merging data streams, [RGBD](#) and [G<sub>s</sub>D](#); we have a multiple input model, shown in Fig 5.3. The merged network used two [CNN](#): one to take the [RGB](#)/[G<sub>s</sub>](#) image, and another to take the depth image. The two networks then use a series of convolutions to extract unique features from each of the inputs. The results of the two [CNN](#) are combined and used as input to a third network. The third network further convolutes over the images giving a high-value feature set for the fully connected layer.

As auxiliary features do affect how the network learns and XYZ points are desired, but not commonly predicted, we repeat the experiments not just with different data streams, but alternative outputs. The different outputs aid in showing how the networks can understand and learn both the features and facial structure, in different spaces. The three types of outputs and their metrics that we train the networks to predict are:

- The UV coordinates, in pixels
- The XYZ coordinates, in meters
- The UVXYZ coordinates

---

where the UV points are the 2D image landmark coordinates and the XYZ point are the 3D location of the landmarks in camera space. As the outputs are in non-compatible metrics, they cannot be predicted in the same fully connect layer. To overcome this, we propose a multi-model output, where the final convolutional outputs are fed into different output models. Having different outputs from the convolutions means, for UV and XYZ there will be one model of fully connected layers for the convolution to be passed into. However, the UVXYZ network will have the convolutions output into two different models, one for UV calculation and one for XYZ. Traditionally with the Kinect, we require the 2D landmarks and use them to reconstruct the 3D points with a depth map. Furthermore, by asking a network to infer UV and XYZ points, it could adopt the similar methodology, thus improving performance.

The networks are trained with a batch size of 240 using a stride of one over 100 epochs, using tensor-flow [120] with the Keras [129] API. We used the KOED dataset with 10-fold cross-validation, this ensures the network is trained, validated and tested on multiple participants, illustrating reliability. The cross-validation split was performed semi-randomly, with 70% training, 20% validation and 10% testing, ensuring no participant existed in multiple sets. We use MSE as our loss function, as shown in Eq. 3.7, using Adam [119] as the optimiser. MSE has more emphasises on large numbers allowing for large outliers to be resolved during training. However, we also calculate the MAE, as shown in Eq. 3.8. MAE gives equal weight to all the errors illustrating the overall error. By using these error functions, we can determine the number of errors the networks produce and the size of errors. We use MSE for training as it is traditional in regression-based deep learning.

## 5.4 Results

Using multiple networks, we showed that to track a large number of points on the face, max-pooling layers can have a detrimental effect on the results of the

---

tracking as facial feature and expression are lost to the degradation of the data. We compare the accuracy and the loss of different max-pooling layers in Table 5.1.

As the experiments are to determine the effectiveness of how deep learning can be implemented on a mobile device, the file size of the trained net is compared and illustrated in Table 5.2. The FPS is also given on the mobile devices as well as the image size before the fully connected layers.

TABLE 5.2: Table of the result file sizes

Network	File size	Image size	FPS
Basic	32.5MB	$12 \times 12$	15
Two max-poolings	120MB	$24 \times 24$	N/A
One max-pooling	518MB	$48 \times 48$	N/A
Face Detector Only	50KB	$864 \times 480$	17

As shown in Table 5.2, the number of max-pooling layers has an incremental impact on the resulting weights file. For reference of how this would affect an application, the current Google Play store [130] rules for publishing an application are provided. An application on the Play store cannot exceed the files size of 100MB. Compression on the optimised file also cannot be performed as standard methods negatively affect the reading and use of the files. As a result, the file size of the one max-pooling and two max-poolings could not be loaded onto the mobile device.

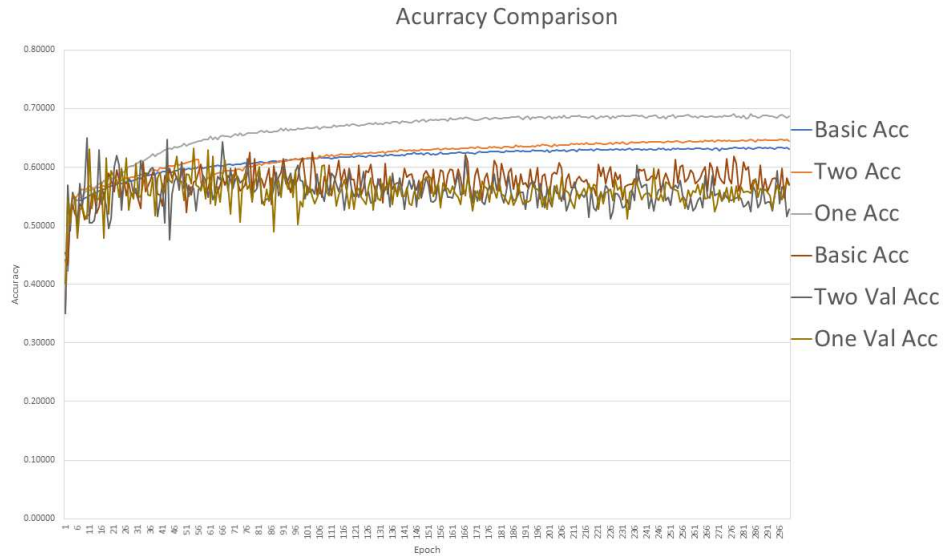


FIGURE 5.4: The networks accuracy, both training and validation.

Fig 5.4 illustrates that as the network begins to retain more data by removing a max-pooling layer, the network gets significant boosts in accuracy. The increase is much more prominent from one max-pooling to two max-pooling, this could be as it loses more data through max-pooling compared to two max-poolings to Basic max-poolings. Fig 5.5 shows that the more max-pooling layer lowers the loss, but is not as significant as the impact on accuracy. Fig 5.6 illustrates that when removing max-pooling layers, the networks accuracy increases this is much more prominent between Basic max-poolings and two max-poolings. The reason behind the accuracy drops is that by using max-pooling on the neural face and large movements can be learned, such as mouth open, because subtle movements and deviation a lost though pooling. However, the method fails to track expressions, such as mouth widen and struggle with accurate placement of the eyes. However, with each improvement, we also receive a corresponding drop in performance.

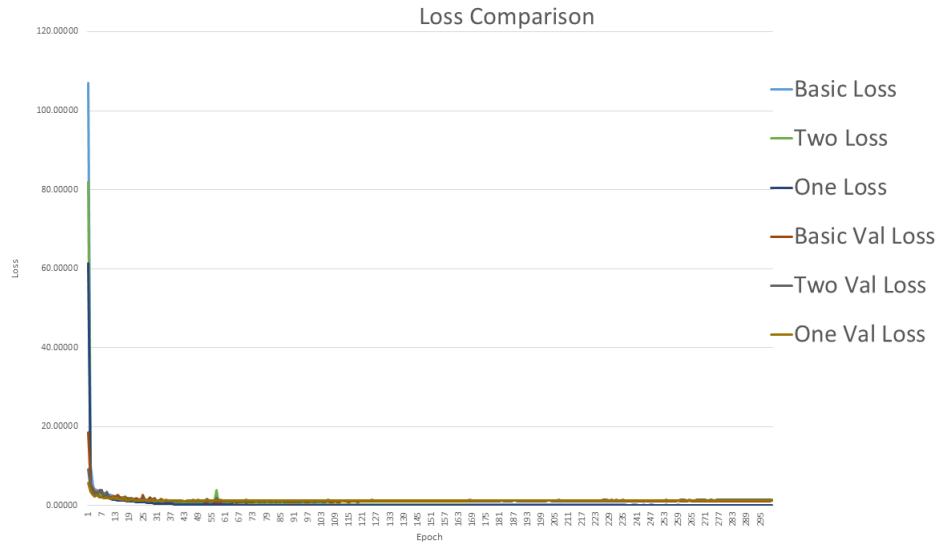


FIGURE 5.5: The networks loss, both training and validation.

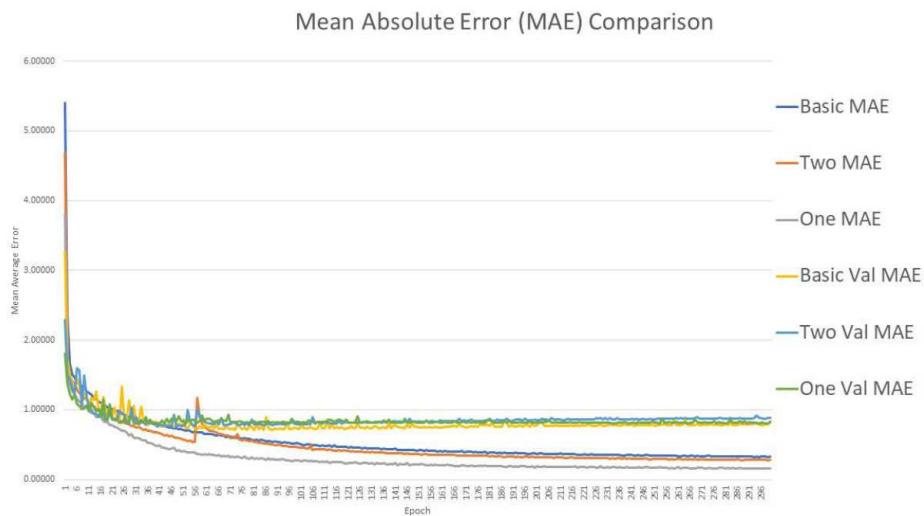


FIGURE 5.6: The networks MAE, both training and validation.

We ported each of the models to an HTC 10 mobile phone (4GB RAM, Quad-core ( $2 \times 2.15\text{GHz}$  and  $2 \times 2.16\text{GHz}$ ) Central Processing Unit (CPU), Adreno 530 GPU) to test the effectiveness on live data. The live tests show that even though the basic network runs the fastest it cannot distinguish between many of the facial movement include mouth widen, the system favours the neutral face. As shown in Table 5.2 all trained models struggle to reach a real-time landmarking time, but the system also had to perform features, such as the face detection. When

---

comparing the performance of the systems we recommend to implement one max-pooling as it aids performance significantly with little impact on accuracy, giving the best cost/reward trade-off, but further research needs to be done to optimise the size for mobile devices. However, if the hardware is restricted and accuracy is not too important, then the two max-poolings will suffice. Image from the live capture shown in Fig. 5.7, where we tested in both controlled light conditions and uncontrolled lighting Fig. 5.14. We also performed a comparison of glasses on/off as in Fig. 5.15. We provide all codes use to build the Android application (Android studio project), codes to train/export (Python 3.5) the model and the models (.pb) used for this experiment publicly available.

The results from the changing the pooling layers and porting to mobile devices, showed that state of the art layers were over pooling the input data, which when coupled with the low image size input meant not enough data was left in the images for the neural networks to track a wide range of facial movements. However, the work did show that the network where capable of performing accurate facial landmark on low powered devices.



FIGURE 5.7: Example output of the android application

When we extend the work to 3D to compare the networks, we first show the validation during training and examine the performance of each stream. For each of the results we start with the UV (2D), then XYZ (3D) and finally the UV XYZ (All) results. After this, we show an evaluation of the networks on testing data

---

and the feature maps produced by the networks. Finally, we examine the results of the testing set with both [MSE](#) and [MAE](#) scores.

We use the information learned from porting to mobile networks to adjust the neural networks structure to reduce the amount of pooling. We also use multiple convolutions, in-between pooling as this is shown to allow for more in-depth features to be collected, similar to [43]. This allowed the produced networks to have increased performance when identifying facial landmarks.

Fig 5.8 illustrates for the prediction of UV landmarks, both [RGB](#) and  $G_s$  converge at similar epochs, 40. Also, they both share many similar traits, such as they both start with a significantly lower loss and have more stable learning than input streams that incorporate depth. Overall, [RGB](#) performs the best in both [MSE](#) and [MAE](#). The networks that merge visual and depth data converge much later than [RGB](#) and  $G_s$ , but their results of [MSE](#) are close to the [RGB](#) and  $G_s$  scores. [RGBD](#) and  $G_sD$  have unstable learning curves and encounter hidden gradients that cause loss to increase rapidly. The single channel  $G_sD$  converges earlier than [RGBD](#), indicating that a single clean frame learns faster on how to smooth a noisy depth map than a three channel [RGB](#) image. The single channel depth encounters the most unstable learning and converges at a much later stage, showing without a visual stream to assist the depth data cannot easily locate UV landmarks. Further illustrated by depth performing the worst when evaluated on [MSE](#) and [MAE](#).

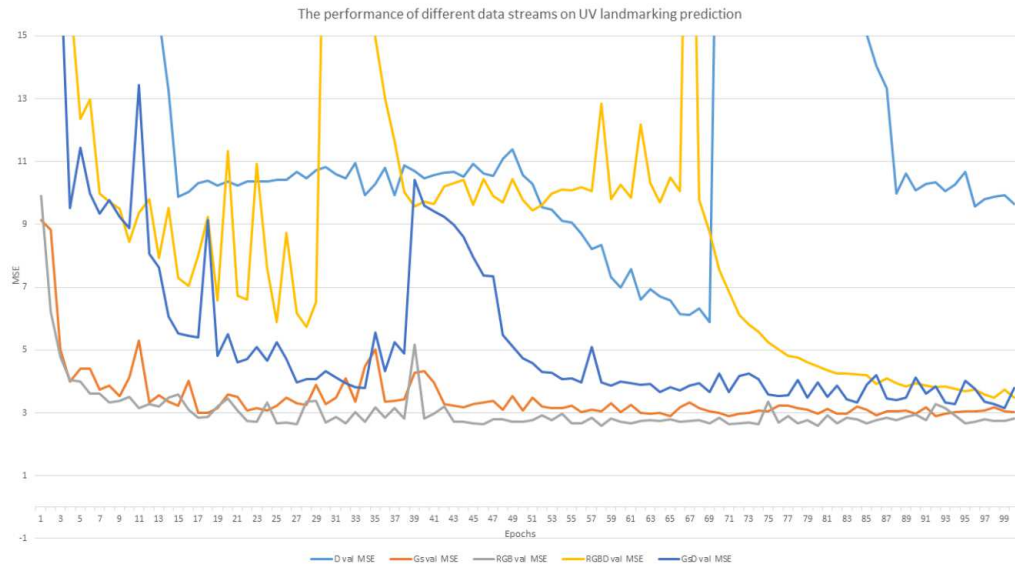


FIGURE 5.8: The MSE of the UV Only networks validation over 100 epochs.

Fig 5.9 illustrates, the MSE of the XYZ only network, similar to UV, RGB and  $G_s$  start with a low loss and converge the quickest at around epoch 30. However, the learning is unstable, indicating retrieving accurate 3D landmarks from visual images is a difficult task, although in the final epoch RGB has the lowest MSE. The input streams that incorporate depth converge sooner than in the UV prediction networks. Furthermore, their learning rate is more stable than the RGB and  $G_s$  stream, but hidden gradients are still an issue. Also, they converge at a similar location slightly higher than RGB and  $G_s$ , although at some point they score lower loss than the RGB and  $G_s$  networks. This convergence also occurs after a hidden gradient, indicating there is a shared local minimum caused by the inclusion of depth data, the most prominent of these is  $G_sD$ , in which consistently has the lowest loss over epochs until it reaches a hidden gradient, to which it then becomes the worst performing stream.



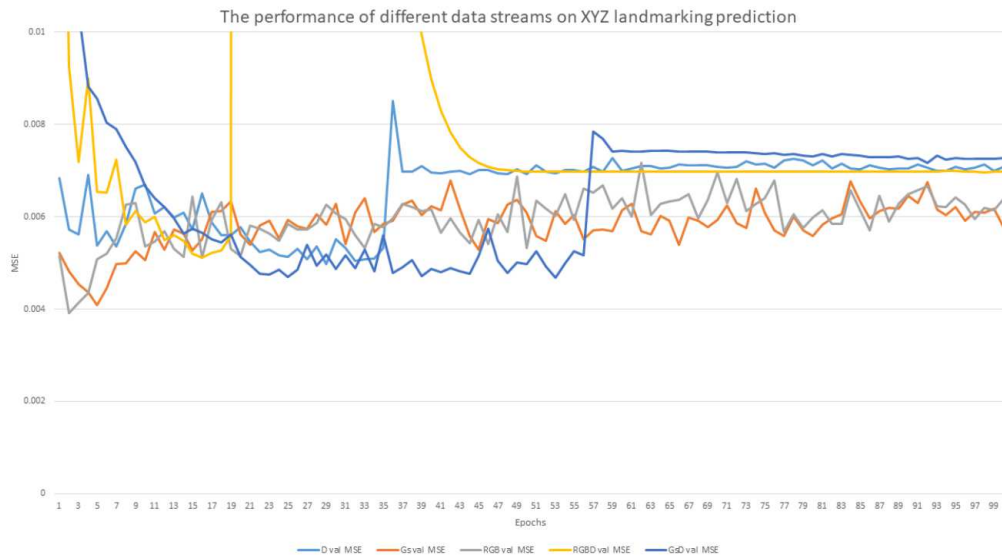


FIGURE 5.9: The MSE of the XYZ Only networks validation over 100 epochs.

Fig 5.10 illustrates the MSE of the UVXYZ networks, where RGB and  $G_s$  begin with the lowest loss, but RGB has a significantly lower loss than  $G_s$ . The learning rates of RGB and  $G_s$  are stable and converge quickly around epoch 43, with  $G_s$  performing the best. The input streams that incorporate depth data also converge quickly, with depth and  $G_sD$  having stable learning rates, unlike RGBD. Furthermore, hidden gradients are still an issue. However, unlike in UV and XYZ only networks the UVXYZ quickly recovers. The recovery demonstrates, how auxiliary information is benefiting the networks ability to learn from the different data streams by overcoming issues, such as the local minimum seen in Fig 5.9.

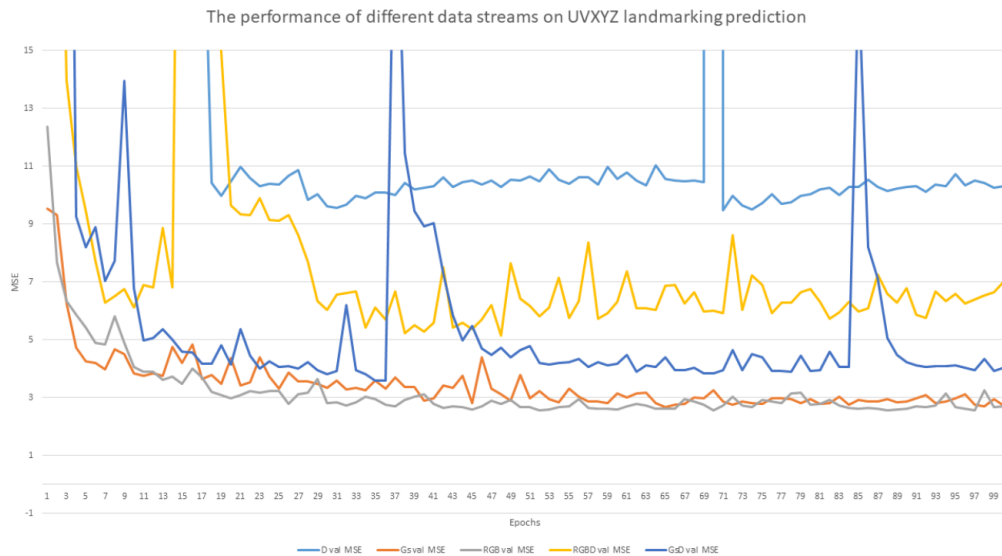


FIGURE 5.10: The MSE of the UVXYZ networks validation over 100 epochs.

Fig 5.11 visually compares the results in both 2D and 3D. We summarise the observations:

- In the UV only prediction, the results are visually similar, but there is some deviation between each of the networks. When using depth as the input stream, the predictions of both the right eye and lip corners are predicted less precise than the other input streams; this could be directly affected by the noise in the depth maps, as when merged with a visual stream, performance is improved.
- For UVXYZ, there is no noticeable difference between the UV results.
- For the XYZ only predictions, we see much larger discriminations, in the predicted facial landmarks. Some of the significant changes are:
  - From the frontal view, there is a variation in the mouth width, with  $G_s$  being the smallest and depth being the widest.
  - Nose landmarks shifts in  $G_sD$  were the nose tip, and right nostril is predicted close to each other.
  - Eye shape changes between networks,  $G_s$  and RGBD, produce smooth round eyes. Whereas, others are more jagged and uneven.

- From the side view, we see the profile of the face change with the forehead and nose shape varying greatly between networks.
- In contrast to the UV results in the UVXYZ network, with the addition of auxiliary information the resulting geometric landmarks on the mouth, nose, eye and eyebrows, become more precise and consistent. In most of the cases, the eyes are smoother, the eyebrows are more evenly spaced, the nose irregularity in  $G_sD$  no longer occurs and the mouth width consistency has significantly improved. These results show that, as UV is easier for the networks to learn as all streams manage similar results when used as auxiliary information, they standardise the 3D locations as well. However, there are still some variations in the profile of the nose, and in RGB the right eye is predicted shut.

Input Data	UV	XYZ (Frontal)	XYZ (Side)	UVXYZ (UV)	UVXYZ (XYZ Frontal)	UVXYZ (XYZ Side)
Grey						
Depth						
Grey Depth						
RGB Depth						
RGB						

FIGURE 5.11: A visual comparison of the results from the 2D and 3D networks.

As shown in Table 5.3, for UV landmarks RGB has the lowest MSE, with  $G_s$  not far behind. It also shows that for predicting landmarks in 3D only, that having both a visual and Depth data allows for the highest precision results, with RGBD and  $G_sD$  scoring the lowest with marginal differences in score. For the MAE and MSE of the UVXYZ networks we show the separate stages of the loss calculation:

- Combined loss, which is the sum of UV and XYZ layers loss.

- UV loss, the loss of the UV layers alone.
- XYZ loss, the loss of the XYZ layers alone.

The combined loss shows the networks overall performance, but the UV and XYZ alone show the performance of the networks on the individual outputs. By comparing the loss of the UV and XYZ alone, we illustrate how the auxiliary information is affecting network performance, compared to networks predicting UV only or XYZ only landmarks. When trying to predict UVXYZ data,  $G_s$  performs the best overall. We show that by introducing the 3D landmarks, we reduce the overall loss significantly to UV alone in both RGB,  $G_s$  and  $G_sD$  networks. Furthermore, the prediction of XYZ is improving in the same networks. We see similar results in the MAE, shown in Table 5.4, where networks reduce the loss below the UV alone networks. However, RGB sees the least MAE for UV. For overall combined loss and XYZ loss,  $G_s$  scores the lowest in MSE and MAE.

TABLE 5.3: Table of the testing set evaluations on MSE.

Input Data	UV	XYZ	UVXYZ	UVXYZ(UV)	UVXYZ(XYZ)
$G_s$	1.8192	0.0023	<b>1.3695</b>	<b>1.3676</b>	<b>0.0019</b>
Depth	6.4672	0.0023	6.6509	6.6482	0.0027
$G_s$ Depth	2.1845	0.0022	1.8933	1.8911	0.0022
RGB Depth	2.1561	<b>0.0022</b>	2.8744	2.8752	0.0022
RGB	<b>1.7488</b>	0.0023	1.5612	1.5592	0.0019

---

TABLE 5.4: Table of the testing set evaluations on MAE.

Input Data	UV	XYZ	UVXYZ	UVXYZ(UV)	UVXYZ(XYZ)
$G_s$	1.0052	<b>0.0341</b>	<b>0.9127</b>	<b>0.8797</b>	0.0330
Depth	1.9150	0.0361	1.9705	1.9322	0.0382
$G_s$ Depth	1.1210	0.0379	1.0617	1.0246	0.0371
RGB Depth	1.0848	0.0367	1.3056	1.2685	0.0371
RGB	<b>0.9553</b>	0.0346	0.9685	0.9388	<b>0.0297</b>

The key differences in single task networks and multi-task networks in predicting facial landmarks were observed in the feature maps of the networks. The network kernels learnt the spatial information from UV prediction. Therefore, the feature maps shown in the UV prediction demonstrates the activation of appearance-based facial features. On the other hand, when predicting the geometry coordinate of XYZ, we observed that the feature maps of the convolutional layers with point-based (facial landmarks) activation. The point-based landmarks are due to the Z component which makes the facial landmarks more separable. The UVXYZ column depicts the features maps in UVXYZ prediction. We observed it has better pattern representation with both appearance-based and point/landmarks information. The feature maps show the networks are capable of processing the input stream to focus on the specific landmark regions of the face. Furthermore, by adding auxiliary information, the kernels become refined able to detect features with high intensity, as the network is forced to learn how the structure appears both 2D and 3D.

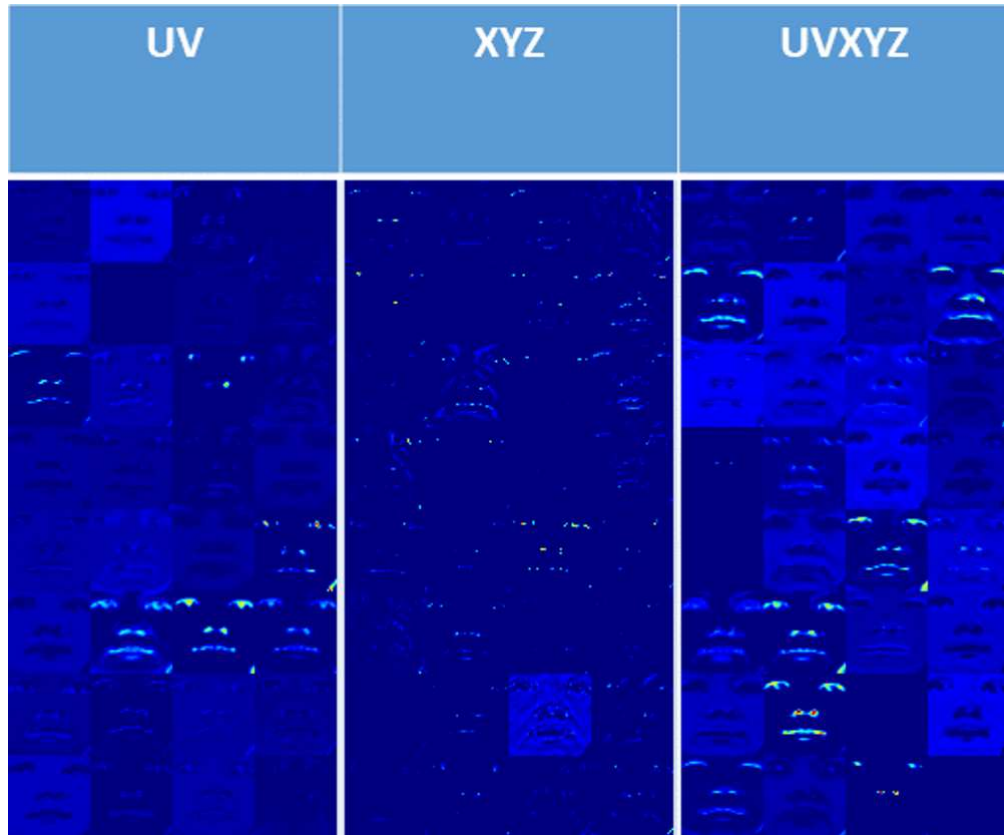


FIGURE 5.12: A visual comparison of the output of the final convolutional filter for each type of network prediction on the RGB Images. The third column illustrates the feature maps for UVXYZ prediction, the best performance with auxiliary information.

To demonstrate the effectiveness of the network, we visualise the predicted landmarks of the  $G_s$  network on a 3D model, shown in Fig 5.13. With  $G_s$  as input data stream, our proposed method predicts accurate 3D facial landmarks on raw depth data using auxiliary information. Furthermore, this illustrates the accuracy of the network, even with raw depth data, our proposed method manages to estimate accurate 3D facial landmarks after pre-processing to crop and resize depth images for the network, where a human would be incapable of without full-size depth images [131]. However, due to the noise from the raw data, the limitation of our proposed method is not able to locate the Z position precisely in some cases.

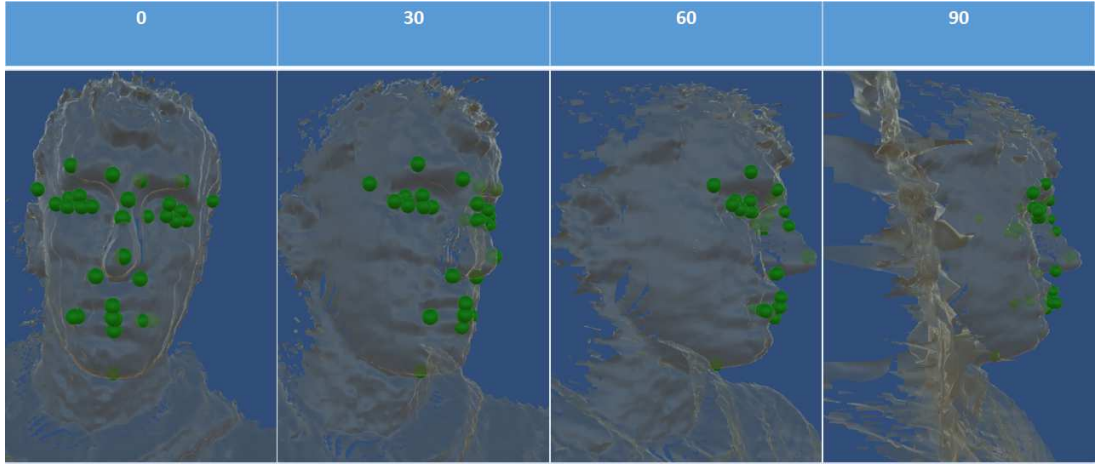


FIGURE 5.13: The result of the  $G_s$  UVXYZ trained network and the appropriate model from the same input depth map. The model is transparent to show the geometry coordinates of the facial landmarks.

## 5.5 Discussion

We have demonstrated the effectiveness of basic neural networks on mobile devices and compared the subtle changes in the network design and its effect on the performance, both on the device and the accuracy of the network. More processor intensive networks can achieve better accuracy for the system, but its performance on mobile is unregistered and an open area of research. By testing higher resource required networks on mobile devices and showing their effectiveness in comparison to the one shown in this paper will aid in benchmarking the progression of systems in the future. A major requirement for future work is to perform optimisation on the application to increase the frame rate to real-time, methods, such as Fagg et al. [132] fast fourier transform face detector, that can run over 60fps freeing up memory and time for the neural network. Other methods could also be tested, such as Ranjan et al. [133] to allow a neural network to handle everything from face detection to landmarking.



FIGURE 5.14: Example output of the android application in a controlled environment(bottom) and wild environment (top), with both male and female faces.



FIGURE 5.15: Same Scenario as Fig 5.14, but with glasses.

In the extended work, we have shown and illustrated the effect of different data streams within neural networks, to identify which streams are ideal for current research topics, as current literature uses a mixture. We also extended the work by the prediction of points in the camera (XYZ) space as this is a valuable resource in the area of facial expression recognition and animation synthesis, but current literature focuses on image (UV) space coordinate systems. Unique insights into each stream of data were obtained, demonstrating, the pros and cons of each stream. To prevent bias, an in-house dataset was used, showing that each network was able to some extent reliably track facial features and expressions in both [2D](#)



---

and 3D. The networks showed that the existing data-streams could accurately predict 2D and 3D landmarks.

Comparing the results and feature maps of the networks demonstrates the ability of the networks to process and understand the different forms of data and if they are beneficial to the network. Full RGB performed the most effective on UV with the least amount of errors and lower scale of the errors that occurred. While depth shows its effective at predicting landmarks, the noise it presents requires additional streams, such as RGB to smooth out and retrieve reliable results. In the final experiment, for predicting UVXYZ, we show that although for UV alone RGB is the most efficient,  $G_s$  outperformed it, illustrating that more generalizable single frames are more effective when predicting a wide range of values. While, depth has shown to be difficult for the networks to learn from, with limitations, such as exploding gradients, even after merging with cleaner streams it has been showed to be effective even when cropped and resized for the prediction of landmarks, where traditional methods require full-size depth images.

This work focused exclusively on the use of neural networks to predict facial landmarks without the aid of physical markers, sensors, or references points placed on the individuals. There have been many incremental studies into the used of neural networks to predict the image (UV) space landmarks successfully. However, the results all use different streams of data with little consensus on why the stream is used, except due to dataset or memory limitations. Also, XYZ coordinates are not being predicted by neural networks in current systems. In-which, many industries desire the use of 3D landmarks in real-time.

There are several limitations to this study, mostly related to the data used to train the network and the difficulty of 3D landmarks. Firstly, due to the context issue of cropping a depth map recording was done in a controlled environment, so the network only has to learn a manageable part of the 3D viewing frustum. This, regarding animation, has an advantage as it normalises the facial position, while still tracking 3D facial movement. However, for full 3D prediction full depth maps would still be required. Future work should seek out new technologies, such

---

as the Intel real-sense D435 [56], which could resolve the noise issue of the Kinect as it uses stereo IR emitters with an IR projector to create clean depth maps, which could see significant improvements. Other aspects, would be to further the work with a larger dataset to test the reliability, of no depth streams with a wider demographic of faces.

## 5.6 Summary

We have demonstrated that the max-pooling layers have high potential to aid in compressing the size of a network to operate on mobile devices. Employing max-pooling layers allows networks to become ‘lighter’ and be trained faster, but with reduced accuracy. With the basic network, the standard learning libraries in networks that don’t employ the max-pooling layers have improved results. We have implemented a deep learning network onto a mobile platform, tested the performance on real-world data. We have also show that the outputs on an array of varying neural networks to demonstrate how the networks were affected by the max-pooling layer. By comparing the cost/reward, we recommend the Basic max-pooling network as it has high accuracy, with little impact on the phones memory or processing capabilities for mobile platforms.

We have shown and analysed how the input data stream can affect a deep neural network framework, for the analysis of facial features, which can have an impact in facial recognition, reconstruction, animation and security, by providing how the networks interact with the different data streams. The stream shows different levels of accuracy and reliability which can positively affect future work. Future work will include increasing the number of the participants, increase the amount of reliably track without markers 3D reference point on the face as current literature is limited in this area.

# Chapter 6

## Depth Data Denoising

*This chapter discusses the issue of the noise in depth data produced by the commercial sensors. Additionally, we evaluate the purpose of neural networks for RGB image denoising, and the possibility to transfer it to depth images. We show neural network denoising on depth images by using a generated 3D depth image dataset.*

### 6.1 Introduction

Facial animation is challenging because of the highly deformable surface of the face, and the structural difference between individuals [134, 135]. Furthermore, many of the techniques available use RGB data that gives a 2D view, where some motions, such as pouting and cheeks puffed are easier to be view in 3D depth data. Many motion capture productions use “middleman” models and have the final model resemble the actor’s face, which synthesise the motions onto the model. The requirement for a model resembling the acting has been a research challenge for many years, with satisfactory results from single camera reconstruction using 3DMM [26]. Although this method produces high-quality results that simulate a real face, errors in the facial structure can occur and the method is profoundly affected by lighting [136, 137], which is why many commercial companies employ the use of optical motion capture for capturing the motion and use a light stage

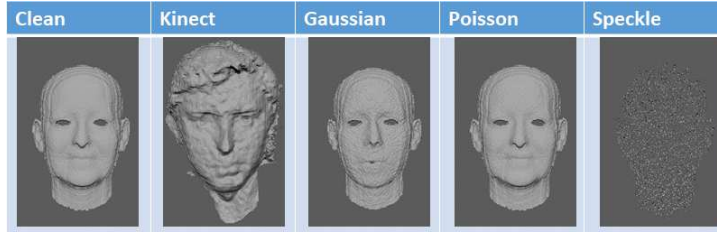


FIGURE 6.1: A comparison of different noise generation methods applied to the depth data and then render into models.

[138, 51] to generate the 3D expression models used in the animation. 3D depth sensor has been used in the past [4] mainly for model generation, which is a much cheaper alternative, but the noise produced by the sensor has prevented its widespread adoption by the research community.

Noise reduction in standard image processing has been significantly improved with the introduction of deep learning techniques capable of denoising a wide range of images, that preserves fine details in images, such as facial features [66]. However, for deep learning approaches, this requires both noisy images and the clean images (as ground truth) to train denoising networks. In depth sensors, from a Kinect One [87] perspective, providing clean depth data as ground truth remains a significant challenge. As supervised learning algorithms requires a substantial amount of data [126], it is common for researchers to incorporate synthetic data into their training data [139]. There are techniques [140, 141] using neural network to synthesise the training data and generate training data on the go.

Several attempts at noise reduction have been made in the past. A popular method of denoising natural images is BM3D [67]. Difilippo et al. [142] suggested to “warm up” Kinect before recording, which improved the result of full body images but did not preserve the face details. Temporal denoising [143] require sequences of depth data, where the subject needs to stay still or perform certain poses, which are computationally expensive and do not work on a single frame. Zhang and Wu [63] proposed a light convolutional network to denoise depth image. However, the method did not preserve the image size. Consequently, it is not able to generate the 3D model due to the changes in perspective and size.

---

This chapter presents a novel method in denoising face depth data in real-time while preserving the details of facial features. We introduce a customised loss function for the training of neural networks and demonstrate its capabilities on 2D data, as well as 3D data. We show that it is possible to use Neural Networks to achieve comparable denoising results as state of the art, BM3D, with lower computational time. To overcome the issue of data deficiency in depth data, we proposed a new method for generating synthetic depth data (clean model) and add Gaussian noise (noisy model) for our experiment. The neural networks train on this large synthetic dataset and able to denoise the synthetic depth data and real depth data on single frame depth data in real-time.

## 6.2 Related work

The depth sensing devices, such as Kinect, has been used to build 3D face models in the past. Frame integration and non-rigid ICP [25] were used to remove noise from the constructed model, which allows for accurate facial landmark annotation [131]. Frame integration is vulnerable to motion distortion, and it allows a full rotational model of the face to be generated. An example of the Kinect raw frame vs an integrated frame is shown in Fig 6.2.



FIGURE 6.2: An example of an integrated model (left) and a single frame model (right).

Noise reduction is an important field for signal and image processing. Images can be affected by a wide range of noise types [144]. Image processing approaches have been used widely in denoising images, such as median filter and average filter.

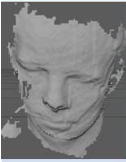
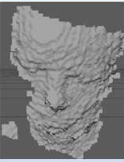
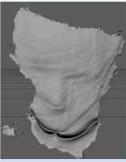
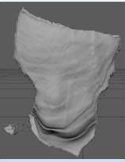
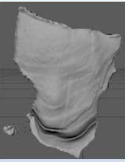


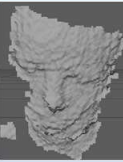
Clean	Noisy	Sigma 25	Sigma 15	Sigma 5	Sigma 1	Sigma 0.1	Sigma 0.01
							
PSNR:	46.78	47.98	47.87	47.58	46.85	46.70	46.78

FIGURE 6.3: A comparison of the different values of sigma for [BM3D](#). Traditionally sigma is increased with noise, but for depth data, we found reducing sigma increases performance.

However, such a filter may not be suitable for face depth data as they remove the finer details in the image, such as facial features.

[BM3D](#) [67] is the state-of-the-art algorithm for natural [2D](#) image denoising. It identifies similar regions in an image and places them into a [3D](#) group. The groups go through a series of [3D](#) transformations, linear, shrinkage and finally an inverse [3D](#) transformation. The final image is then produced by a collaborative filter that preserves the unique features of each image and the features that are common. However, the method is an intensive computational task, and the image must be split into overlapping blocks and compared, which gives both its broad application and long processing time. [BM3D](#) is also limited due to its block matching nature, without any matching blocks it cannot perform. Furthermore, as sections of noise can match, the system can in extreme cases spread the noise.

The related works are split into two sections. Firstly, we evaluated the method of merging [RGBD](#) for cleaner depth maps. Secondly, we investigate depth only refinement methods. However, many of the techniques shown focused on denoising [RGB](#) images, where they showed good results, which could be transferred onto depth.

Traditional [ToF](#) depth data denoising follows a strict path of using the easily acquired high-resolution colour images to give detail into the depth surface and refine depth positions. Diebel and Thrun [145] proposed the use of a multiple-resolution based Markov Random Field to combine a low-resolution depth image and a high-resolution colour image to refine the depth data and improve its resolution. This method was proposed as light is absorbed differently by the different

---

colours, such as black absorbing more light causing depth to be predicted further away and vice-versa for white. They provided a good clean result on the area with a distinct deviation between colours but is not able to smooth areas, such as wood due to its colour diversity. Qingxiong et al. [146] proposed an up-sampling approach for multiple functioning sensors with a refinement approach. They refined the depth map with a series of cost calculations, bilateral filtering and then sub-pixel refinement to the depth map. Their method provided clean up-sampling rates at higher factors than other methods at the time. Chan et al. [147] extended on existing work on Joint Bilateral Upsampling by Korf et al. [148]. They wanted to improve the system as although the methods allowed some improvements, texture information from the colour image could be synthesised into the depth map, such as text on a bag that could not, naturally, be picked up by a depth camera. The effect of textures being applied causes problems as the true depth has changed, Chan et al. proposed a noise aware filter that considers colour before applying changes to the depth map to reduce the texture effect by applying Gaussian filters independent to the depth. By using this approach, they produce high-resolution depth maps with less large noise peaks. Park et al. [149] broke away from using bi-lateral filtering to use a least-squared algorithm that utilises multiple weight factors with nonlocal means filtering to preserve depth edges better while maintaining smooth surfaces, even on complex shapes. He et al. [150] created a guided filter that uses information from a reference RGB image or the image its self. The guided filter improves upon existing methods by describing more than just when to smooth the depth map, by having filters available for edge preservation, and when encountering light scattering surfaces, such as fur. He et al. focused their attention on the denoising of colour images instead of depth but produced a noteworthy high-end result. Wu et al. [151] provided a much-improved method of improving depth images, that again uses a reference RGB, but normalises the luminance in the scene and produces an albedo image. The albedo image is then used to produce an edge map, and coarse information refined using the shade information. They then used an iterative process to refine the depth map that allows for even subtle details to be passed to the depth map while avoiding colour

---

texture, such as text being passed. They finalised the work by employing it to a GPU allowing for real-time processing at 30 FPS. The methods shown provide effective means to clean a depth image and improve some detail found in the images. However, the issue with the methods shown is that they rely on receiving a clear RGB image and then good alignment of RGB and depth images, this means areas in low (dark)/high (glare/white-wash) of fast /blurry movement cannot be used as an effective method to clean the depth image. A method of depth only cleaning would be more effective.

Although many denoising techniques with deep learning use single images, some use multiple images as reference for the neural network, such as Zhang et al. [152]. Like the previous section, they used an aligned RGB reference image to aid clean the depth map. However, instead of using a statistical model they use deep learning to refine the depth map. The work gave good results but implement max-pooling which reduced image size drastically; they also used the Middlebury dataset which used different scanners for ground truth but adds ‘holes’ in the depth map to simulated noise, which is not an entirely accurate depiction of depth noise. For our experiments, we want to only use single depth maps as input for denoising, bridging the gap towards more traditional methods of image denoising. Due to the nature of depth maps and the techniques of rendering 3D data, we review networks used in denoising, but only test networks where the input size is equal to the output. Xu et al. [153] improved upon denoising by using Deconvolution Neural Network (DCNN) targeted for denoising through the uses of deconvolution filters. Deconvolution filters have the opposite effect on image size to normal convolutions allowing for the image to be up-sampled, this is commonly used in VAE, and they compress and uncompress data which has the potential for denoising. They also introduce a multiple stage network that performs outlier-rejection to improve network performance. Using the outlier-rejection network, the authors significantly improve denoising accuracy, but their network down-samples images from their input size, even with deconvolution layers owing to the use of Max-Pooling. Dong et al. [69] experimented with the use of deep learning for denoising with a small scale neural network. The authors create a three-layered



---

neural network that utilises [ReLU](#) activation after the first two convolutions. The network although small uses a large number of kernels, 64 for the first convolution, 32 for the second, and just 3 ([RGB](#) channels) for the third. The authors achieved a high accuracy well performing neural network, that can restore and highlight many different types of features. Zhang et al. [[154](#)], created a much deeper neural network. The authors created this network to be able to clean noisy images with a variety of noise levels, against the traditional method of denoising images with a consistent Gaussian noise level applied. As a result, the network was much more resistant to different noise levels and became more diverse. The network can outperform many state of the art methods, such as [BM3D](#) [[67](#)], which is commonly used as a benchmark. Mao et al. [[155](#)] expanded denoising in two ways firstly; they use VAE methodology, where the image is convolved into its core features and then restored, usually used in compression. However, by reducing an image to its core features, noise can be removed before the restoration process. Additionally, Mao et al. implemented a technique where the result of previous convolutions is saved and added to future convolutions to preserve information. The preservation of data and the length of the network makes the authors technique effective, but memory and processor intensive. Nah et al. [[156](#)] proposed a series of refining neural networks that cleans a noisy image. The authors' technique starts with a small image that is processed by a neural network to perform denoising. After the processing, the image is then up-sampled and passed to another neural network that performs more denoising. This process is repeated until the image size is met. Their technique improved performance significantly in dynamic scenes, such as pedestrians moving or camera-jerk. Bae et al. [[157](#)] proposed a large scale neural network that repeats a module of convolutions with batch normalisation. The network is designed to denoise Gaussian data. In most recent work, Mildenhall et al. [[158](#)] produced a new network, similar to Mao the use a VAE style network with data retention between the layers. The network is also trained to be diverse in the levels of noise the network can remove. The network uses a large number of filters in each layer and instead of max-pooling implements average pooling. A large number of filters allows full in-depth de-noising that can target both large

---

and subtle features in an image.

The networks describe, except Zhang et al. [152], focused on the denoising of 2D images. When using neural networks for denoising, MSE is used as a standard as a loss function, and then PSNR is used on the result to determine the effectiveness of the model. However, with minor adjustments, PSNR can be used as a loss function for neural networks allowing the network to optimise for the standard evaluation metric. We show the customised loss function in equation 6.1, which is the negative of 3.11. By using the negative form of PSNR the metric can be used as a loss function in a neural network, which means the network will be optimising for the metric it is commonly evaluated on.

$$\text{PSNR} = -(20 \times \log_{10}(\text{MAX}) - 10 \times \log_{10}(\text{MSE})) \quad (6.1)$$

where:

- MAX is the maximum possible value in the ground truth, for the depth data, this is 8000.
- MSE is equation 3.7.

## 6.3 Proposed Methods

During this experiment, we went through significant revisions of our methodology after being peer-reviewed by experts in the field. In this section, we discuss our first attempt of denoising depth data. Then we show and give reason for the the drastic changes in the methodology and demonstrate are revised technique.

### 6.3.1 Proposed Data Synthesizing method

Two existing methods are currently used for synthesising depth data. The first is BlenSor [159], which employs a physically accurate ray-cast system to calculate

---

the light bouncing off surfaces. The ray-cast give BlenSor high accuracy but is very computationally expensive. BlenSor produces an accurate depth map and then post processes to add noise. The second method by Rahmani et al. [139] employed a series of computationally intense processes, such as backface culling and hidden point removal. They then use Gridfit [160] to put the vertices in the form  $Z(X, Y)$ , image form. Gridfit smooths the surfaces, so nearest neighbour is used to remove any unwanted points and lastly normalisation between 0-255 is performed. The authors method creates a depth map but loses information along the way with Gridfit and normalisation.

As the two previous methods have shown that they can produce depth data, they are processor intensive tasks to produce their results. They also use complicated steps that take time to recreate and follow. However, most of these steps have been done naturally for 3D computer graphic systems. To explain how virtual environments are rendered, the viewing frustum is shown in Fig 6.4. The viewing frustum is common in all 3D applications; many referred it to the camera or eye. The viewing frustum is used for rendering the scene to the 3D view. During the rendering process, the origin of the camera becomes the origin of the scene; every object position is relative to the camera. The frustum then begins a sequence of culling:

- *Out of view culling:* The pyramid style structure of the camera is used to detect if an object or part of an object is within the view of the camera. If any section of an object is not within the frustum, it is culled from the scene as it will not appear in the rendered image; thus, wasting memory by performing rendering and lighting calculations on it.
- *Occlusion culling:* Another pass is then made to detect if an object is occluded by another in front of it, any occluded sections of an object are culled from the scene. During this stage, a special case can happen owing to the transparency of objects; this should be taken into consideration if working with a model implementing alpha channel textures.

- 
- *Distance culling (Z pass)*: As shown in Fig 6.4 the frustum contains a near plane and a far plane. The frustum then calculates the distance between the camera and each of the remaining objects in the scene. The system uses the distances to cull any points outside of the viewing range.

To generate the depth data, we use the Distance culling or Z pass. The frustum uses the points within its view to produce a depth image of the scene, using the camera as an origin, just like the Kinect. The depth image is produced in real-time (or the speed at which you render) and is accessible in most 3D software packages, either through code or a shader [161]. This allows full scene depth capture with preserved accuracy while avoiding computationally expensive steps in previous research [139, 159] such as hidden point removal, Gridfit and nearest neighbourhood matching. The Z pass, also skips the processor intensive task of ray-casting the scene method of BlenSor. Similarly, compared to both BlenSor and Rahmani et al., this method gives a clean depth image and by setting the frustum to replicate the setting of any available depth sensor (FOV and resolution), an ideal image that we would want from the Kinect, which we can post process to add Kinect like noise to the image. We provide the code (python based) to extract this from blender, set to capture when the user renders an animation. We demonstrate this method using blender, but it can be easily recreated in a wide range of 3D applications, such as Maya [83], Unity3D [91] and the Unreal Engine 4 [162].

Even though this was made to simulate Kinect data to be a baseline for showing synthetic data can be used for improving real data, this applies to other depth sensing devices as they implement the same data capturing mechanism. For our method, we calibrate the camera to Kinect specifications. However, it is possible to recalibrate with the other camera. Furthermore, as most of the real-time sensors use IR or structured light, the noise is nearly identical. As noise is near identical, it implies that simulation of noise is achievable by fine-tuning the Gaussian intensity and distribution.

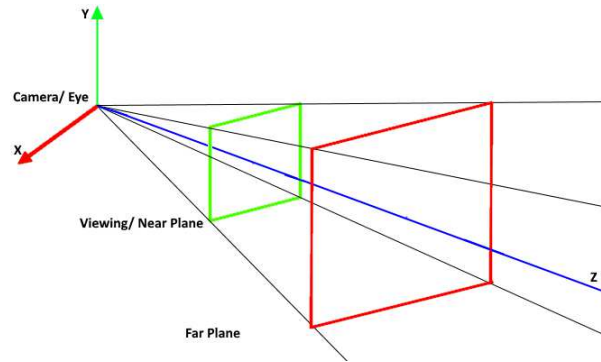


FIGURE 6.4: The 3D viewing frustum (perspective mode) used in many 3D applications.

### 6.3.2 Generation of noise upon depth images

The Kinect One's noise is generated by its ToF system, which implements an IR blaster [163]. The Kinect sends a pulse of IR light and uses the time it takes to return to triangulate the distance from the sensor. Although depth sensors, such as the Kinect can provide accurate depth images of the environment, Wasennmüller et al. [164] and Choo et al. [165] showed the sensors usually produces a wavelet style noise over the whole scene, with additional noise at edges of objects. It should also be noted that the noise generated by the Kinect is also reduced once it reaches operating temperature [98, 166].

The wave style of noise on the Kinect image is similar to ripples in a puddle, where we still get a smooth surface, but the noise comes from the waves causing some sections of the puddle to be higher and some lower than the still height, in our case further away (lower) and closer (higher). To identify the best noise generation method, we compare three state-of-the-art methods: Gaussian, Poisson and Speckle noise. We applied different intensities and value to closely represent the noise created by the Kinect sensor as shown in Fig 6.1. As shown in Gaussian, noise resembles the noise produced by the Kinect sensor. This is similar to the BlenSor system that uses Gaussian to add noise to the system. The Gaussian blur was set with the mean of the random distribution to 0 and a variance of 1. However, they did not fully resemble the Kinect ToF noise, making the method

---

ideal for real-time clean depth collection, but not simulating Kinect style noise. However, the method of using Gaussian to simulate ToF data has been used in modern research [167].

In our initial work we used the propose data generation method. However, after peer-reviews we adjust the noise generation method to a C++ implementation of BlenSor with a sensor set up to the Kinect standards. Due to the method of BlenSor, where a ray-cast must be used and then noise applied to change the ray-cast, this means an original clean depth map can be captured before noise simulation. The ability to capture both clean and noisy depth maps allows the system to take a 3D model and produce the dataset required for de-noising.

The split is performed semi-randomly to ensure no participants exists in multiple sets to prevent bias. The datasets used are:

- D3DFACS [1]
- Face warehouse [94]
- Biwi Kinect [65]

Each model was placed one meter in front of the sensor facing directly into it, all models were uniformly scaled to ensure natural proportions were maintained and the datasets use different size metrics. This method generated a total of 46,140 noisy facial images. A sample of a model generated by the noisy and clean images can be seen in Fig 6.5 and demonstrates the clear difference between what sensor could predict vs what current sensors receive. If it is possible using deep learning to receive models similar to clean it could improve the accuracy of 3D systems drastically.

### 6.3.3 Experimental Setup

This section describes the experimental setup for all methods. To ensure a fair assessment, the experiments follow a similar protocol. For input data, we test the

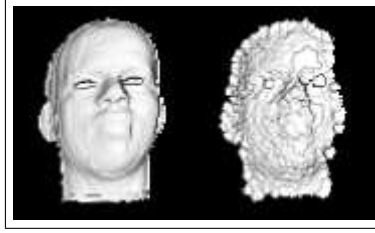


FIGURE 6.5: An example of a clean (ideal) model versus a noisy model. The image illustrates the type of noise we must simulate, such as the wave type effect over the skin and the self-occlusion noise of the positional difference between the IR transmitter and receiver.

networks with two types of input: raw data and normalised data. Raw data is the depth sensor's reliable range, i.e., 400-8000 for Kinect. For our first attempt at denoising, the systems outperformed BM3D.

We perform 10-fold cross-validation on the dataset. As our training dataset was constructed from multiple existing datasets, we ensured that an equally representative amount from each dataset was in the testing set. The split provided use with 34122 training images and 3844 testing images. Due to the size of the training set and memory limitations on RAM, we split the training set into mini-batches of 4000, meaning it took 9 of these to train a single epoch.

In our revised work from our initial submission, we test against a series of state-of-the-art networks that focus on colour image denoising, to demonstrate the increased difficulty in denoising ToF data compared to standard 2D images. Also, we propose a customised loss function derived from PSNR. As illustrated in equation 6.1, we flip the value of PSNR to negative allowing it to be implemented as a loss function. We implement the PSNR loss function as MSE is used in standard denoising neural networks, but to test the effectiveness of the network PSNR is used. With deep learning, it makes more sense to use a loss function that optimises for the evaluation metric performed. To test the performance of PSNR loss, we train networks with 2D  $G_s$  images, using both MSE and PSNR as loss function and compare the results.

---

TABLE 6.1: The results based on raw depth data. The mean and SD are obtained for PSNR and Time. The model indicates if the method can generate a 3D model.

Methods	PSNR	Time (seconds)	Model
BM3D [67]	31.98 $\pm$ 0.48	1.8559 $\pm$ 0.036	No
BaseNet1	64.36 $\pm$ 0.07	<b>0.0027 <math>\pm</math> 0.0004</b>	Yes
BaseNet32	<b>87.56 <math>\pm</math> 2.50</b>	0.0077 $\pm$ 0.0005	Yes
VAE1	52.92 $\pm$ 3.11	0.0039 $\pm$ 0.0004	Yes
VAE32	58.80 $\pm$ 3.58	0.0065 $\pm$ 0.0005	Yes
Burger et al. [66]	34.57 $\pm$ 0.49	4.6503 $\pm$ 0.0909	No

TABLE 6.2: The results based on normalised data. The mean and SD are obtained for PSNR and Time. The Model indicates if the method can generate a 3D model.

Methods	PSNR	Time (seconds)	Model
BM3D [67]	<b>92.60 <math>\pm</math> 0.52</b>	1.9660 $\pm$ 0.0501	Yes
BaseNet1	31.70 $\pm$ 22.69	<b>0.0028 <math>\pm</math> 0.0004</b>	Yes
BaseNet32	31.59 $\pm$ 37.14	0.0077 $\pm$ 0.0005	Yes
VAE1	31.93 $\pm$ 8.96	0.0039 $\pm$ 0.0004	Yes
VAE32	31.78 $\pm$ 11.91	0.0065 $\pm$ 0.0005	Yes
Burger et al. [66]	16.19 $\pm$ 0.01	4.6903 $\pm$ 0.0599	No

## 6.4 Results

In our initial work, we experimented on a series of small neural networks for depth data only. When we extended the work, we implemented the state-of-the-art networks and test on both colour image denoising and depth data.

As mentioned, VAE were used in the hope of acting like an attenuation network, learning just the key features in the images and then from them to reconstruct the facial features. Although the networks scored high in training, surprisingly they performed poorly on the normalised depth values (0-1), and 32 kernels could produce a model, and it did not produce a usable depth map.

In Table 6.2, we showed a comparison of the available methods performances. For the BaseNets and VAE network, they were trained on normalised data. Table



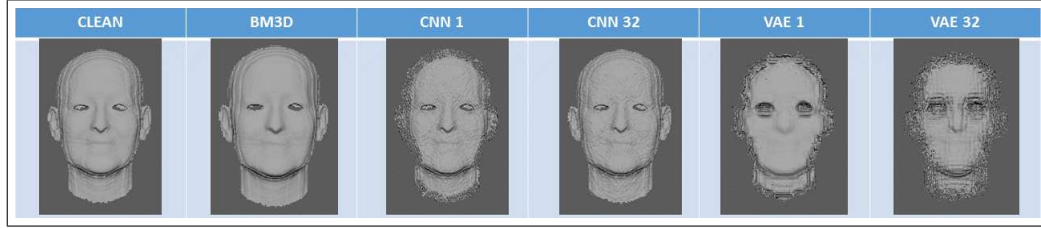


FIGURE 6.6: This figure represents a visual comparison of the denoised 3D models using normalised data. The clean shows the ground truth, BM3D is state of the art. BaseNet1 and BaseNet32 are our proposed methods; VAE1 and VAE32 are the networks based on VAE.

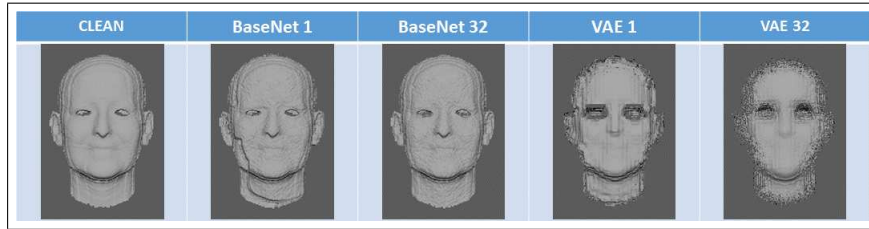


FIGURE 6.7: A visual comparison of the denoised 3D models using raw depth data. The clean show the ground truth model; BaseNet1 and BaseNet32 are our proposed methods; VAE1 and VAE32 are the networks based on VAE.

6.2 shows that BM3D has achieved a mean PSNR of 92.60 and an SD of 0.52. In contrast, the BaseNets and other methods have poor accuracy and a high SD, but BaseNet1 is significantly faster than BM3D. A visual comparison of the denoised models using normalised data is illustrated in Fig 6.6, where it is evident that BM3D produces a clean model. The BaseNets struggle to clean the complex features of the face and the edges of the model recede inwards. Whereas the VAE network reconstructs some of the facial features, such as the eyes, much of the model has been smoothed or is missing.

Table 6.1 shows the experimental results on raw depth data. BaseNet32 achieves the best result with a mean PSNR of 87.56 and an SD of 2.50. State of the art, BM3D, and Burger et al.’s method performed poorly on raw depth data. The BaseNet1 achieves the fastest processing times, but BaseNet32 is not far behind. BaseNet1 has improved if compared to the results on the normalised data, and BaseNet32 manages to denoise while preserving the facial features, such as the mouth as illustrated in Fig 6.7.

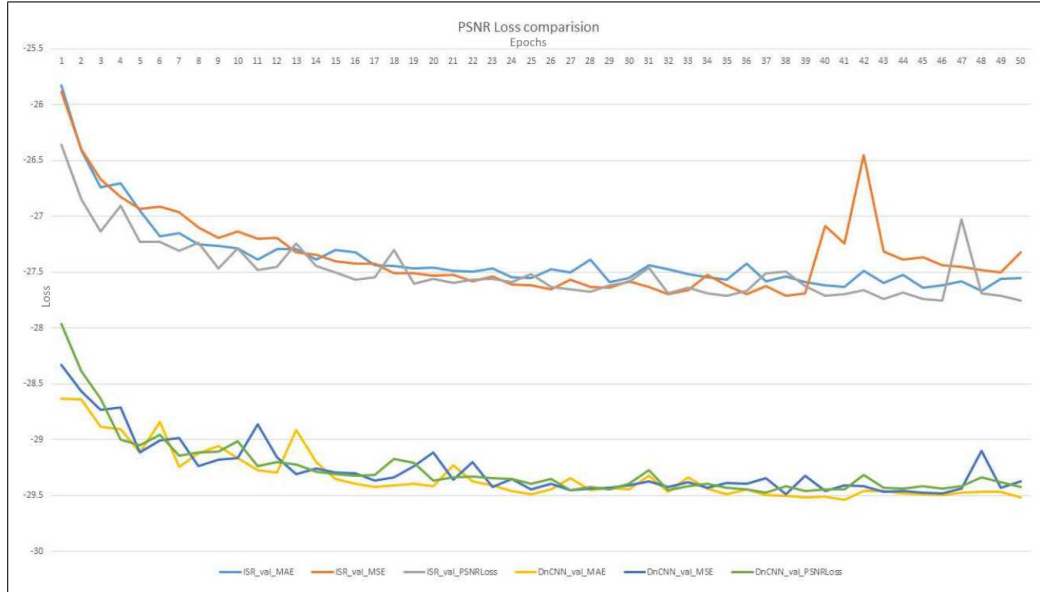


FIGURE 6.8: The PSNR Loss of networks the trained networks.

In the extended work, we began by training state of the art denoising networks, image super-resolution [68] and Deep Denoising Super Resolution Convolutional Neural Network (DDnSRCNN) [155] on the morph dataset. The networks used to allow for state of the art results and comparison. The networks were trained three times using alternative loss functions:

- MSE
- MAE
- PSNR

where MSE is the traditional loss function, but MAE is implemented in some cases. The use of PSNR is to compare the results of the differently trained network to demonstrate a change in the networks performances.

For RGB denoising, we train on the Morph dataset. To simulate noise, we perform the traditional method of applying a Gaussian filter, reducing the image size by half and restoring the image size. We evaluate the custom loss function against network trained to optimise MSE and MAE loss.

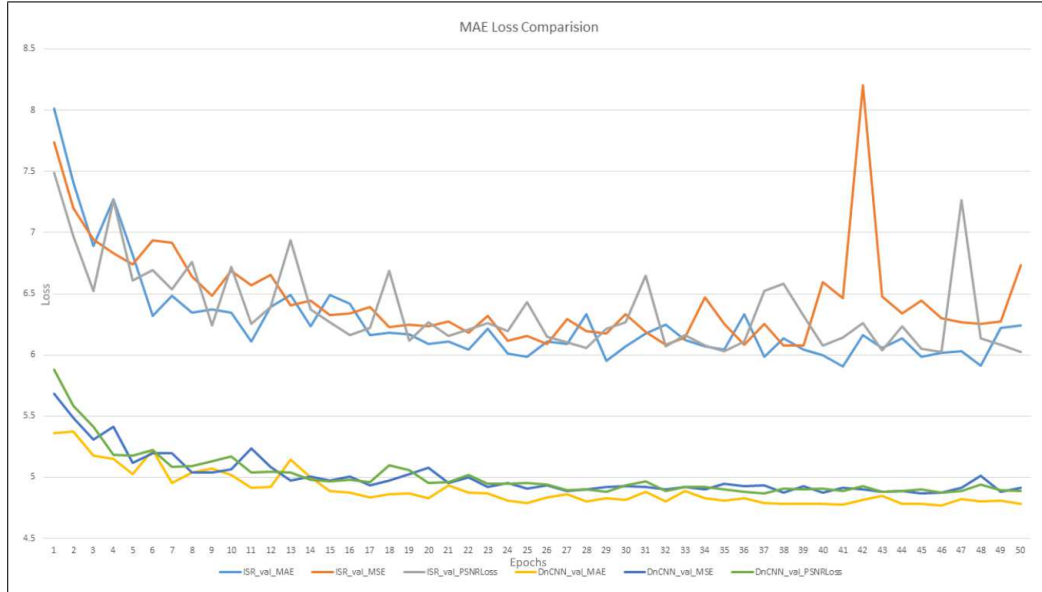


FIGURE 6.9: The MAE Loss of networks the trained networks.

Fig 6.8, illustrate that when optimising the network, it naturally has increased performance for PSNR evaluation in the image super-resolution network, but for DDnSRCNN the optimal loss function is the MAE. The difference between the two networks highlights the variability between neural networks, where network structure can be more suit to alternative functions.

Furthermore, the difference was highlighted in the when the networks are compared on MAE, illustrated in Fig 6.9, where MAE performs well, but in image super-resolution, PSNR performs well. In contrast, DDnSRCNN again has improved performance with MAE as its loss metric. Also, we observed similar result when evaluating the MSE components. A visual comparison on the super-resolution network is illustrated in Fig 3.8, showing how changing the loss gives vastly different results.

For 2D image denoising, we have shown that for some cases using the PSNR loss function can improve network performance, but plenary tests should be performed. Using the knowledge gain from this work we expend the networks to denoise depth data. For the state of the art methods, we used data generated using the C++ version of BlenSor.

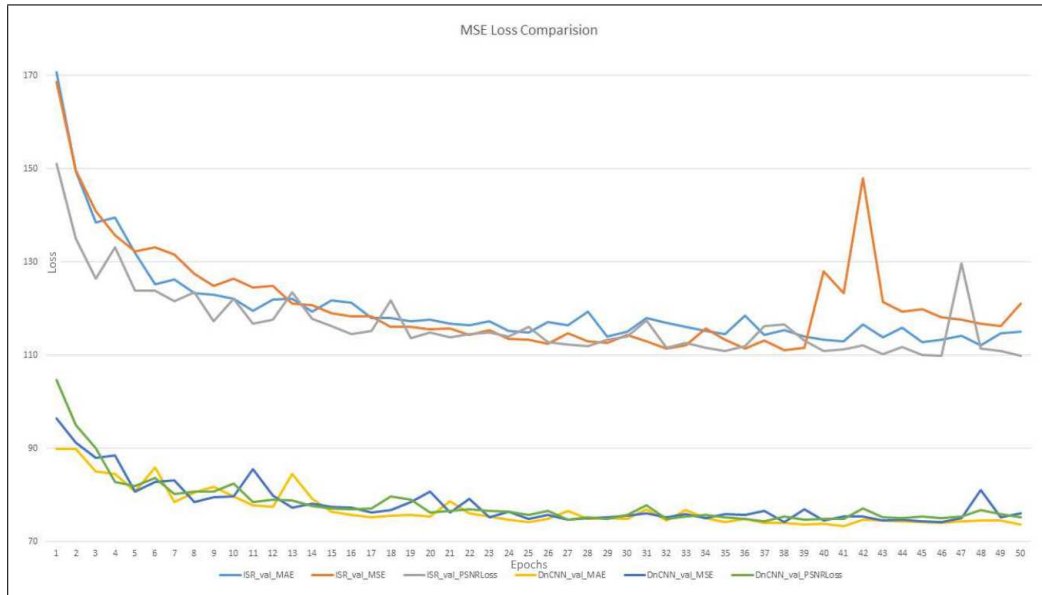


FIGURE 6.10: The MSE Loss of networks the trained networks.

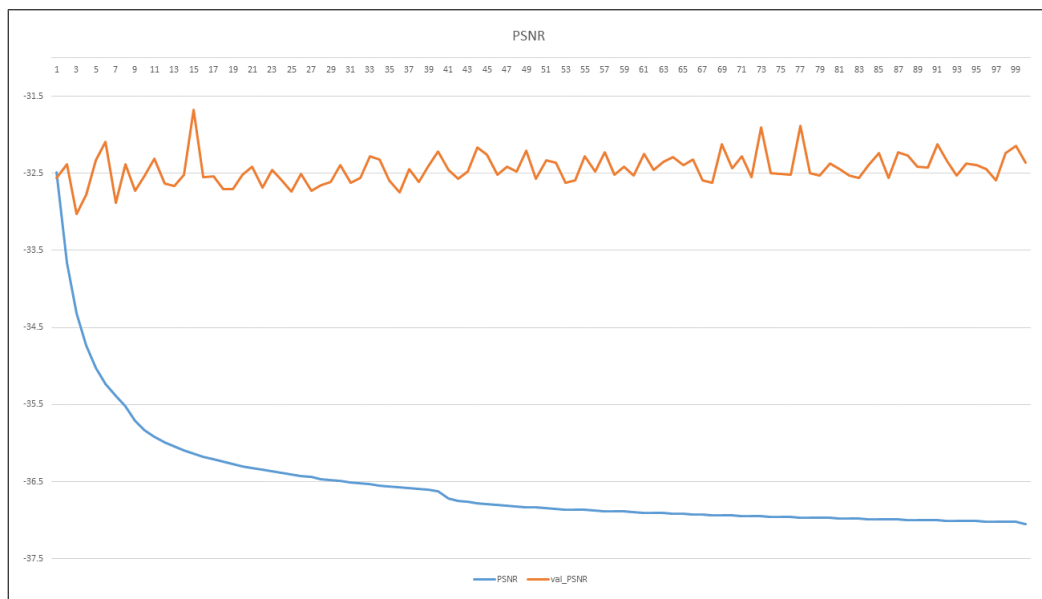


FIGURE 6.11: The Loss of the depth networks.

As illustrated in Fig 6.11, we see similar results to the RGB tests. Our results show that DDnSRCNN had significantly improved performance in comparison to the other evaluated networks. In some cases, the depth data even scores higher PSNR when compared to RGB data. However, the validation illustrates a very different aspect, whereas the training steadily decreases validation remains consistent, which is also shown in MSE Fig 6.13. In contrast, this differs when visualising

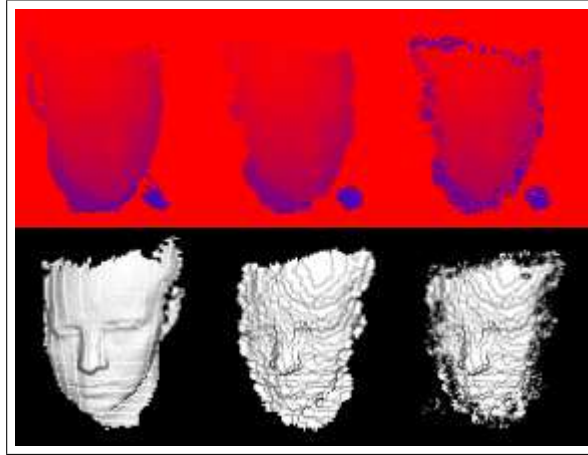


FIGURE 6.12: The ground truth clean (left), the noisy input (middle) and the network result (right).

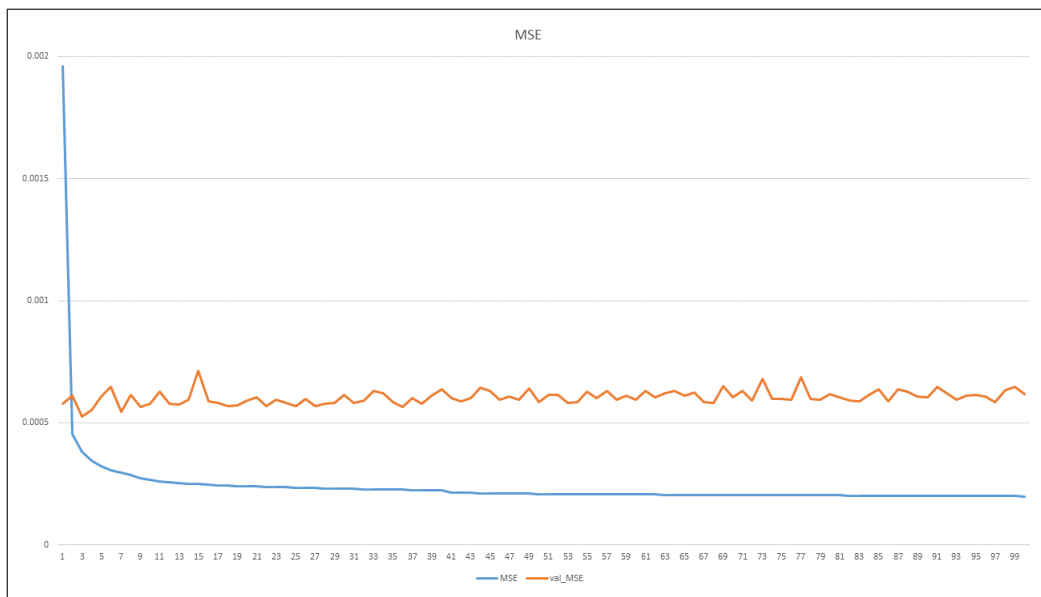


FIGURE 6.13: The Loss of the depth networks.

the data, as shown in Fig 6.12 the edges of the face have become faded. Furthermore, this extended to 3D when using the depth map to render a 3D model.

## 6.5 Discussion

From our experiments, only BM3D (on normalised depth data) and BaseNet32 (on raw depth data) are suitable for face depth data denoising. However, when compare the computational time, BM3D takes 1.9660s to denoise while BaseNet32

---

only requires 0.0077s for the similar task. Also, [BM3D](#) requires a pre-processing stage to normalise the depth data. In this aspect, BaseNet32 is more suitable for an end-to-end solution as it works well with raw depth data. Even though [BM3D](#) has the best [PSNR](#), for real-time depth data denoising, we recommend our proposed BaseNets (BaseNet32) as the solution. To illustrate the practicality of our proposed BaseNet32 in a real-world application, we have demonstrated its performance on a real-time Kinect depth sensor system.

In the extension, we showed a custom loss function could help with the denoising of [RGB](#) data. However, for depth, the system could not easily denoise, even with a large-scale dataset. A solution to this would be to generate an even larger dataset, but even the current dataset has memory restrictions. For training the depth denoising networks a machine with 128GB [RAM](#) is required. Additionally, as depth data is saved in its custom format, loading the data can be a slow process. Another, solution is the use of inception network architecture [[111](#)], the fading of the edge, could be attributed the kernels not seeing the full curvature, only the noise from the small filter size, our initial work shows large kernels overcome this, but state of the art shows small kernels are more effective denoisers. Inception networks could overcome this as they can gather both the larger surface and the smaller details, at the cost of computation time.

## 6.6 Summary

In this section of the thesis we have shown a novel method of capturing synthetic depth data and then using that data to train a Deep neural network to denoise real-world Kinect One depth data, with the easy application to work on similar depth sensing devices. We evaluated current methods and neural networks capable of performing denoising and propose a new network that can take in noisy depth data and output a clean depth map that preserves the facial features. We also show the running times on each of the methods to show their applications to real-time usage. We compared this to the state-of-the-art denoising methods and showed

---

a minor cost in accuracy to increase the processing time significantly. We have shown that using the synthetic data we are also not restricting the method by testing on real-world data and receiving a clean model. This work can be applied to many fields of facial tracking and animation, where due to the noise in the depth sensor has been avoided. The method of creating synthetic data also has many applications in future work, which is why we have provided all code for capturing, training, using the models as wells as the pre-trained models available for use.

This work has been peer-reviewed at an international conference (CVPR2017), where although they liked the idea, it was rejected. The focus of the rejection was the initial methods of data synthesis using the viewing frustum and not enough comparison against state of the art. From the conferences feedback, we implemented the C++ version of BlenSor to replicate the noise more realistically.

The main reason for maintaining the size of the depth images is to enable future expansion of our work to broader applications. Now that we know that neural networks can remove the noise from depth sensor data in real-time, we want to generate random office and rooms within a virtual environment. We then can render out depth maps and use this to test how neural network can denoise images with a wide range of structure, such as a smooth wall, humans and various small items.

In the extension of the work, we analysed a custom loss on [RGB](#) images and showed improved performance with some networks. We then applied the networks to depth data. Unlike tradition de-noising depth poses additional challenges for denoising. As the datasets used for generating the training set focuses are head only models, we crop the images to 96\*96 containing the head only. We crop the faces as preliminary work showed that as the full image contained no information and the loss function of [MSE](#) and [MAE](#), cause then network to create a plain average image. Whereas, by focusing the network on the face, the networks could learn. However, when the networks learn the can only denoise the centre of the face, and the edge fades significantly.

# Chapter 7

## Multi-Tasking Neural Network for facial landmarking

*This chapter introduces an all in one network for facial landmarking and expression intensity prediction. In the area of deep learning, some networks target key sections of the facial animation production pipeline but do not focus on the use of it.*

### 7.1 Introduction

As shown in chapter 5, by adding additional auxiliary information to a deep learning network, performance can be increased significantly. Expression recognition using deep learning is widely performed, but the majority of the existing works did not include expression intensity [168], which is an important property for animation. However, some FACS prediction work [169], focuses on the intensity aspect as it is a natural part of FACS coding. Since high-end systems for facial animation employ landmarking based inference and consumer-based use generated model interpolation and comparison for the expression prediction, we want to integrate these systems into a single neural network.

The purpose of combining landmark prediction and expression intensity is twofold:



- 
- Firstly, previous research [24] showed that using the networks to predict additional auxiliary information boosts the performance of the network significantly. As the state-of-the-art methods use landmarks for emotional inference, these can provide critical descriptors to the network when trying to learn the facial structure and what features best represent the facial expressions.
  - Secondly, a restriction on many trained models is that they can only predict what they have been trained on, with access to the landmarks with a system that understand the deformities cause by expressions, it can be used in a multitude of ways through post processing, such as alignment.

Landmarking is a widely researched with implemented techniques, in both 2D and 3D. In research, landmarking is used for facial alignment and to guide processing methods to target areas of the face, such as wrinkle analysis [170] and movement detection [171]. Whereas in commercial, landmarking is used in various mobile applications that augment facial appearance, identify faces for security and animation models. As shown, much of the work deals with dynamic faces, such as alignment and facial feature augmentation, whereby training a system to understand expressions the systems landmarking could improve due to the knowledge of facial movements. By providing additional 3D landmarks, further, understanding of facial structure and movement can be learnt, such as when a landmark could be occluded.

Gender prediction is widely researched and can be done with high precision [43, 172]. As networks can distinguish common attributes that indicate whether a person is male or female, such as the width of the chin being generally wider in males. By using the network to discover features that separate male and female, it can use this knowledge in other areas, such as how beards affect the landmarks through obscuring the view.

The aim of this chapter is to demonstrate the effectiveness of extending 2D and 3D landmarks to accurately predict intensity expression. To do this we perform the following steps:

- 
- Adapt landmarking networks to predict both expression intensity and gender, as landmarks can provide valuable insight into both of these fields.
  - Visualise the convolutions to demonstrate networks understanding of facial regions. Additionally, describe convolutional outputs to show their significance.
  - Visualise the [MLP](#) to demonstrate how the neural networks interpret the convolutions and differentiate between the expressions and genders.
  - Demonstrate the training process of the network.

## 7.2 Related Works

In commercial productions, such as the widely used Visage face tracking [173], used in FaceRig [3], can use single [RGB](#) or [G<sub>s</sub>](#) cameras to track a face in a video input. They can produce a wide range of outputs, such as a [3D](#) face model, allowing for both [2D](#) and [3D](#) landmark prediction. Furthermore, like many commercial productions for animation, they follow MPEG-4, which allows extensive use of the [3D](#) model for animation, such as those from Autodesk character generator [174] and Mixamo [175] of instant transfer for the facial pose to the models. In commercial production, the method of following an MPEG-4 or [FACS](#) method is conventional as it requires little prior information from the user, mainly just a neutral face image. The landmarks are then normalised to allow for the tracking of facial movements. The advantages of these methods are:

- Widespread use of the MPEG-4, means many [3D](#) models are pre-setup to accept this information for animation
- The MPEG-4 and [FACS](#) cover natural motions of the face
- Built-in expression intensity
- The knowledge that different features will move in tandem, but must be categories separately

- 
- Little human interaction required

However, the system is difficult to train as the motion cover a wide range of movements. In contrast, some high-end productions will use customised actor models for tracking, as it tailors to the actors own unique facial features.

As shown in the Chapter 2 literature review, the pre-existing research in animation focuses on the use of tailored 3D models that are pre-recorded. Then, with a repository of existing 3D models when a new frame is present, comparisons can be performed to determine the expression/s being performed. Building a dataset for individual users has some advantages:

- Custom Blendshape library
- Tailored to each user

However, this method requires a great deal of human interaction, such as choosing motions and holding expressions for recording. Also, where MPEG-4 would categories the expression happiness into separate movements of the mouth and eyes if a recording of a smile is used the combined movements can later confuse a pre-built movement model. The process of using pre-recorded models is demonstrated in a commercial product, such as Faceshift [2].

## 7.3 Proposed Method

We implement a CNN style network for prediction using an entirely regressive approach. We train a single network to predict four key facial features:

- UV Landmarks: these are the coordinates of facial key points in the input image.
- XYZ Landmarks: These are the coordinates of facial key points in 3D real-world space.

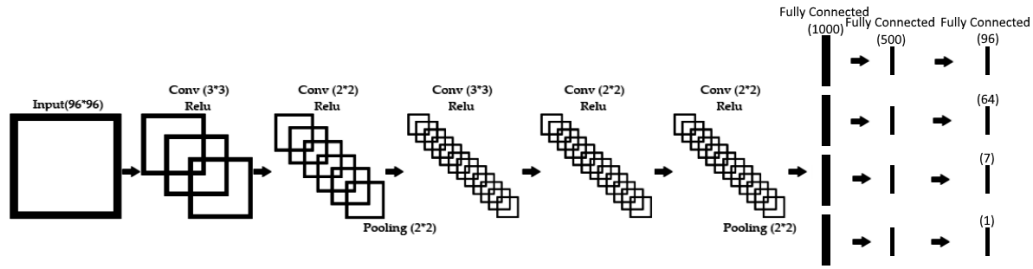


FIGURE 7.1: A visualisation of the network used for this experiment

- Gender: This is the gender of the person present in the image.
- Expression intensity: This is a series of values that determine the current expression and its intensity.

The neural network, adapted from previously done research [24], showed the most effective stream for a neural network when working with multiple outputs. Furthermore, the network showed improved results against other state-of-the-art networks. As illustrated in Fig 7.1, the network takes single  $G_s$  images and processes through a series of convolutions with ReLU activations. Unlike other current methods of landmarking [63, 34], we perform multiple convolutions between pooling layers as:

- Convolutions act as the feature selection method, by performing multiple convolutions with only activation in-between we can get highly descriptive features. Whereas, few convolutions can only achieve general facial features.
- By only using the pooling layer twice and starting with large images we better maintain the image ratio which directly affects the accuracy [128]. The cost of accuracy with max-pooling is due to the loss of features during pooling. Features can be lost during pooling, due to the reduction of the image size, pooling half the images size with a  $2*2$  kernel, further pooling reduces it again. The reduction from multiple pooling layers means not all features can remain in the image.

---

As illustrated in Fig 7.1, the fully connected layers for each output use the same amount of neurons, except for the last output which is the number of required outputs, for example, 64 for UV landmarks and 1 for gender.

The network takes in a large scale facial image ( $96\times 96$ ) that provides a full view of the facial and allows room for pooling without the removal a key facial details. The network integrates two max-pooling layers ( $2\times 2$  windows), with the first after two convolutions when the general facial features have been collected and secondly before the MLP, by using the pooling layer before the MLP we remove a great deal of processing requirements, while holding onto the final convolutions key features. The network uses five convolutions layers the first layer implement a  $3\times 3$  layer followed by a  $2\times 2$ , after the max pooling the convolutions follow a similar pattern, but with the last convolutions layer at  $2\times 2$ . The output of the convolutions is feed into multiple MLP layers that is used to predict the alternative outputs, 2D landmarks, 3D landmarks, gender and expression intensity.

We performed a 10-fold cross validation on the dataset. The split was performed semi-randomly to ensure no participant existed in multiple splits of the data. The networks are trained using a batch size of 50 over 50 epochs.

## 7.4 Results

We report the results by each of the separate outputs of the network and then the overall combined results of the network.

The first output is the standard 2D landmarks, which are the image coordinates of the facial features. As shown in Fig 7.2, The network quickly learned the relevant features for accurate facial landmarking. The networks additionally perform notably better than previous work [24]. The increased accuracy allows improved reliability in the tracking of facial features, thus better inference of emotion. The second is an extension of the 2D landmarks, which moves to 3D. It

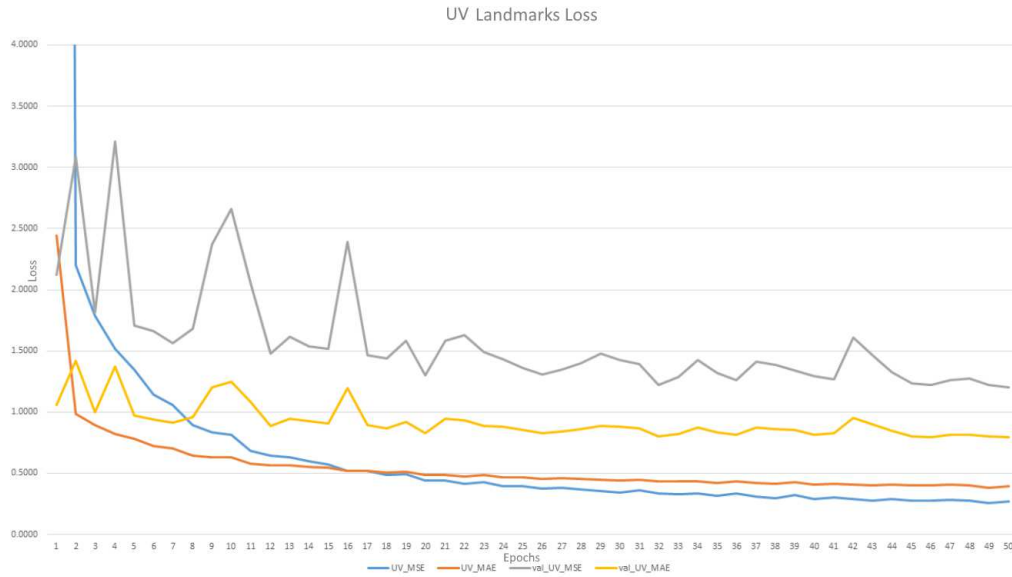


FIGURE 7.2: The MSE and MAE loss for the 2D landmark output

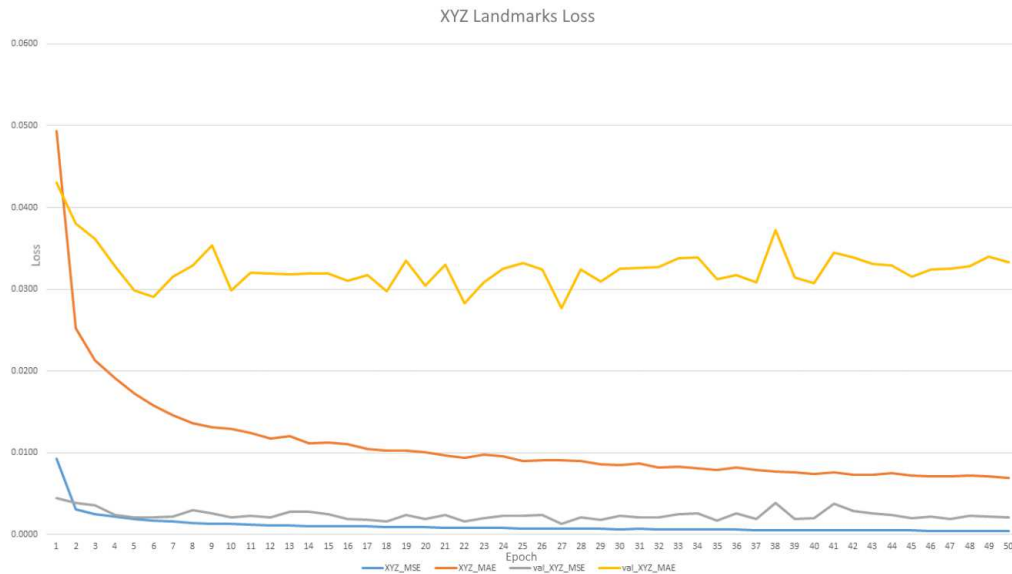


FIGURE 7.3: The MSE and MAE loss for the 3D landmark output

also learns similarly to the 2D landmarks as shown in Fig 7.3 and again performs better than previous work [24].

For the gender prediction, during the training stage, the method converged rapidly on the training set. However, this section has the highest deviation between the training and validation. The validation during training is shown in Fig 7.4 and shows the difficulty in classification is over a coin flip chance. This is due to lack of large participants in the data, but a solution could be provided through extended

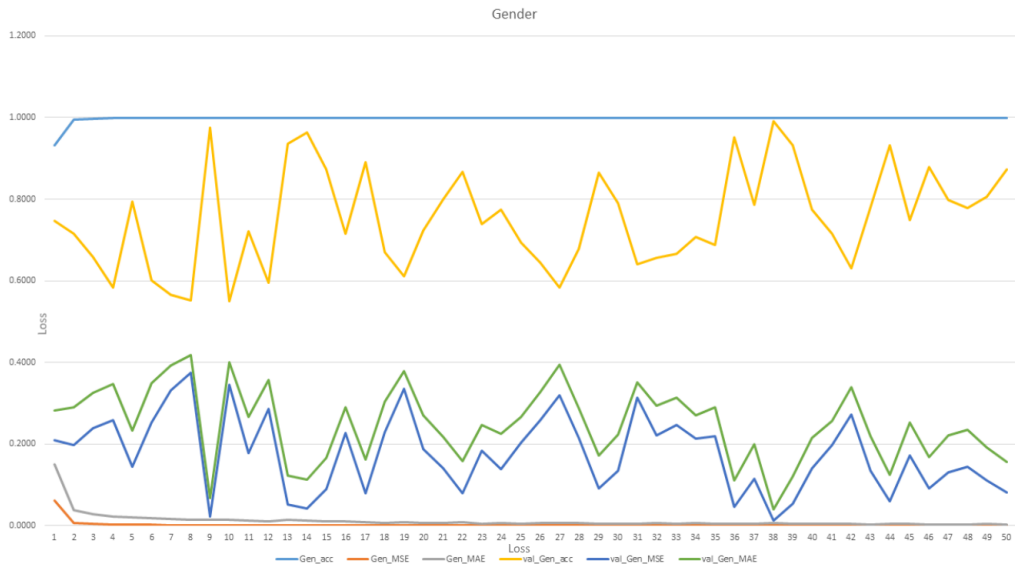


FIGURE 7.4: The MSE and MAE loss for the gender output

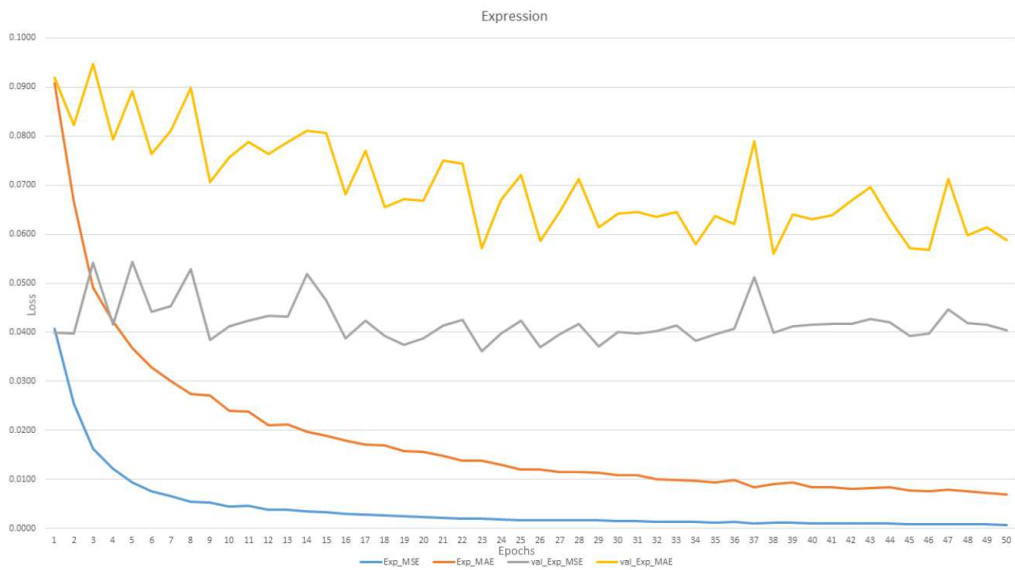


FIGURE 7.5: The MSE and MAE loss for the expression output

training on additional datasets. However, as shown by Hassner et al. [172], even with a large network trained with a diverse dataset, the state-of-the-art method still struggles to retrieve high accuracy results. An unexpected result is the effect of expressions on gender prediction, where males were more likely to be predicted wrong on data with expressions.

For the expression intensity prediction, the network performs well, with a low loss in both training and validation, as illustrated in Fig 7.5. The trained network

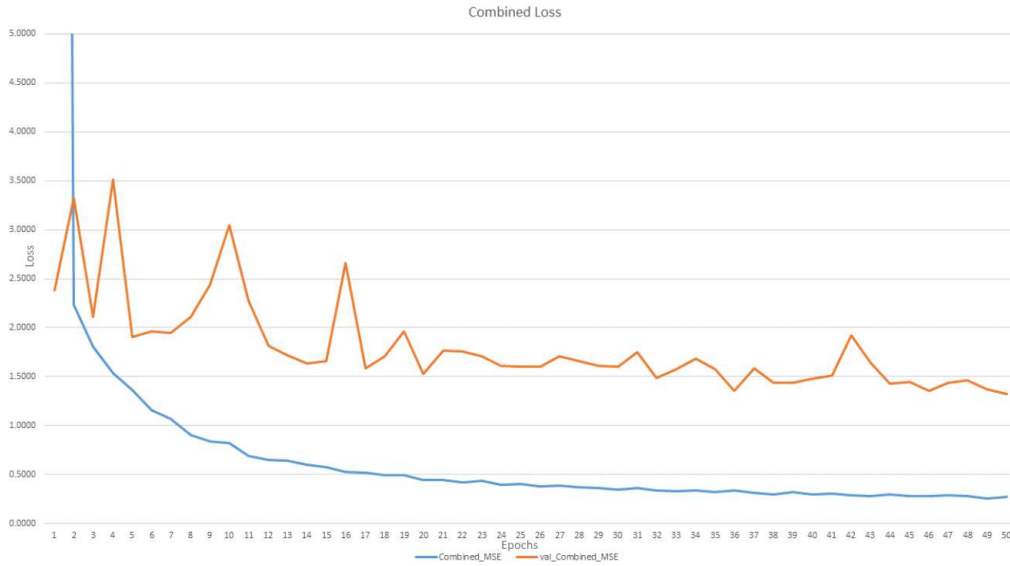


FIGURE 7.6: The MSE and MAE loss for the combined outputs

allows universal expressions to be tracked with ease, and with the system trained on the onset and offset of expression allows precise tracking of expressions.

Overall, the network achieves a low combined loss showing a deep understanding of the facial features required for tracking, as illustrated in Fig 7.6. The network gains a thorough understanding of the facial features using the convolutions, as shown in Fig 7.7, where:

- The method learns feature-based kernels shown by the full-face images.
- Key focus sections, such as the eyes and nostrils in the fourth row.
- Points based kernels, such as the bottom row kernels.
- Region-specific kernels, such as the first row kernel activates more on the lower face, but the sixth row focuses on the upper face regions.

These are the features learnt by the network. The testing scores, shown in table 7.1, showed the network manages to score low loss on the testing set. The low loss scores are highlighted with both the XYZ landmarks and the expression. These results directly compare with the 2D landmarks in table A.1, which allows





FIGURE 7.7: Outputs of the final convolution layer

TABLE 7.1: The results of the network on the testing set.

Metric	Combined	2D landmarks	3D landmarks	Expression	Gender
MAE	1.1919	0.9695	0.0422	0.0631	0.1170
MSE	2.4609	2.3323	0.0053	0.0433	0.0800

comparison of how adding gender and expression predictions directly affected the networks understanding of landmarks.

By extending how the results are predicted in the MLP of the network, as illustrated in Fig 7.8, there is a difference in the activations when discriminating between male and female. The female neurons activation is usually significantly higher than the male counterparts, indicating that the features for female are more visible than for males. However, where the males have higher activations, the female neurons do not activate, indicating that males have features identifiable that female don't, such as beards or the use of make-up could be covering the features

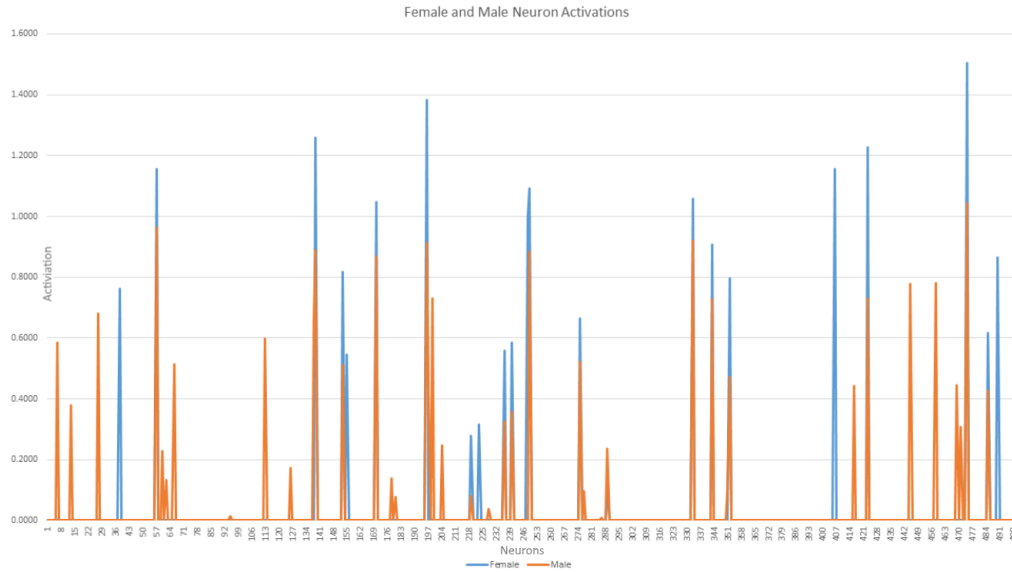


FIGURE 7.8: A comparison of a definite male and female prediction.

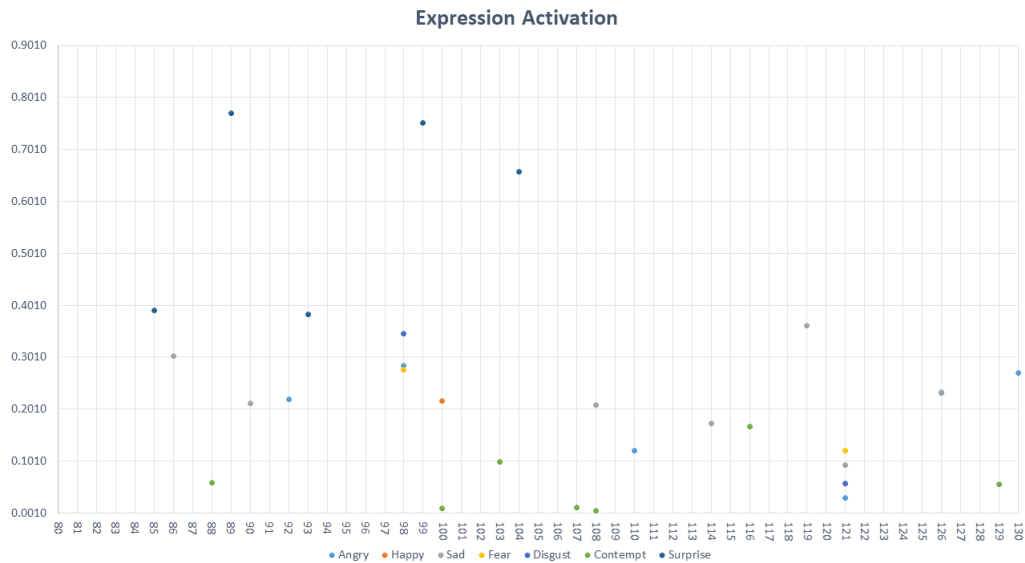


FIGURE 7.9: A comparison of definite expression predictions. This figure illustrates a subsection of the graph, a full version is available in the Appendix B Fig B.1

the system uses for males. Furthermore, key differences are identifiable when predicting various expressions as illustrated in Fig 7.9, where spikes in neurons can clearly define the predicted expression. The similarities between activations for multiple expression can be linked to the network registration similar movements, such as the mouth in fear and happiness.

---

## 7.5 Discussion

We have shown the results that a deep learning network is capable of regressing accurate facial expressions. The universal expressions use a wide array of muscle on the face and demonstrate these movements are track-able, along with the 2D and 3D landmark coordinates. By using landmarks and gender as auxiliary features, the network is trained to focus on crucial movement points in the face when determining the types of expression.

In future work, we would expand the system to a categorise movement for methods, such as FACS or MPEG-4, as the current system is restricted to the universal emotion. However, by expanding into more specific muscle and facial movements, could prove difficult for a network or in-contrast could help the network more deeply understand the face. Also, by using MPEG-4 and FACS, the system would allow integration with existing facial animation systems intuitively. However, for MPEG-4 and FACS based systems, specialist annotators are required.

A solution to improving the work comes from the inherent ability to train separate sections of a neural network. The inputs and outputs of our network were directly affected by the availability of data. However, we can “freeze” sections of a neural network; this allows users to pre-train a network on our dataset and then refine sections of our network. For example, since gender classification in our network performed the worst and there are many available datasets, that only include gender we could freeze all convolutions and only train the fully connected layers that deal with gender improving the system. By freezing and retraining sections, the network it allows the use of multiple datasets that does not cover all parts of the network.

## 7.6 Summary

This chapter has introduced a neural network for processing single images and providing essential information for animation purposes. The network is refined

---

from the previous piece of work and inspired by the current state of the art that allows a wide range of facial details to be acquired.

The network manages to achieve high accuracy results for the desired outputs, such as landmarking in [2D](#) and [3D](#). In-which, aids the system to correctly predict and expressions intensity. By using the expression intensity, a natural interpolation of and actors motion can be realistic synthesised to a [3D](#) model. Furthermore, the network successfully regresses a series of universal expressions, allowing for a smooth transition of expressions. We have demonstrated the feature sets the network has learnt and will use to predict the outputs, showing that the system uses a wide range of features in its predictions. We also illustrate and compared the [MLP](#) sections of the network and how they are affected by the different genders and predictions.

# Chapter 8

## Conclusion

*The final chapter of the thesis contains a summary of the contributions discussed throughout the previous chapters. Furthermore, it performs critical analysis of the limitations of the work and the improvements it has made. The final section illustrates the future paths of research and improvements in this field.*

### 8.1 Introduction

Facial landmarking and animation is a vast area of research with much research being performed. The use of computer vision methods, such as [3DMM](#) to generate [3D](#) facial models is a common practice. However, there are alternative methods to identify the facial landmarks and build expression models. There are many challenges and limitations to overcome to ensure the accurate [3D](#) structure of the face is captured and correctly landmarked. The use [3D](#) depth sensing devices allow for multiple frames to be integrated and create a full expression model and a blend-shape library of individual users, which allow for accurate facial representation. Other methods employ the use of [3DMM](#) models to build a statistical [3D](#) representation of an individual's features from a single image.

This thesis proposed a series of methods to improve the use of [3D](#) data to advance the fields of [3D](#) landmarking with [ToF](#) data and expression recognition.

---

The first being an investigation into the accuracy of retrieving accurate 3D landmarks for ground truth comparison. Secondly, an investigation of the ability of neural network performance for landmarking on low power devices available and the effectiveness of the networks. We then extend the work to 3D landmarking and the capabilities to predict landmarks with auxiliary information and prediction of 3D landmarks, with standard images. As the work focused on the unique aspect of facial landmarking with depth data, a new KOED dataset was made and used, that allowed synchronous information from RGBD data for training. Finally, we provide an aspect of future work in the form of improving the depth data quality and showing a working system capable of predicting facial expression with the inference of 3D landmarks.

## 8.2 Research Findings

A summary of the research findings is given in this section of the thesis and how they relate to the objectives shown in chapter 1 of the thesis. The findings will give why the objective was set and how it was achieved, a summary can be seen in Table 8.1.

The first objective was to analyse the ability to retrieve ground truth 3D landmarks via annotation. The ground truth is necessary, as without this a system cannot be trained to predict the landmarks. Retrieving the ground truth was achieved by a creating an in-house tool that used a novel alignment algorithm to align 3D geometry with a 2D reference image. By merging RGB reference images, it allowed the identification of facial features throughout a series of movements. For this work, we used the D3DFACS dataset, as it avoids bias towards the annotator rather than the traditional method. Also, it provides a clean model and requires alignment at extreme poses. Additionally the objective was to collect a new synchronous dataset that collects both RGB and depth data. As the merging of RGB and depth data in a neural network is a crucial aspect of our work. KOED dataset is used in the majority of the work.

TABLE 8.1: The research objectives against the actual outcomes.

No.	Objective	Outcome
1	Identify the research gap and create resources for ground truth for validation.	A gap was discovered in the acquisition of 3D ground truth landmarks, through the use of the 3D annotator. A new dataset, namely KOED, was created that addressed the lack of synchronous RGBD facial data that contains a full expression performance.
2	Propose new methods to detect 2D and 3D facial landmarks in real-time, with a demonstration of use on low powered devices.	An investigation showed the effectiveness of available streams in neural networks for the prediction of 2D and 3D data. We also demonstrate neural networks running on mobile devices.
3	Investigate the ability of current RGB image depth noising techniques and review the effectiveness for depth data.	The state-of-the-art methods of denoising images are applied onto a realistically generated depth image dataset. We demonstrate the effectiveness of using uncommon loss functions to improve denoiser output.
4	Investigate the potential of low-cost depth sensors for facial landmarking and animation in real-time.	We refine the knowledge gained from the previous objects to create a neuron network that can predict 3D landmarks, and the intensity of an expression.

The Second objective was to combine the available data streams to predict 3D landmarks. Our method deviates from the more traditional method that employs full generated 3D models for their prediction, or the use of 3D statistical model prediction for retrieving the landmarks. However, instead of just predicting the landmarks by combining the data streams in a neural network, we perform a critical analysis on the effect of merging compared to tradition landmarking networks. In which, we showed results against depth data, owing to the instability from the noise, whereas, the traditional single-stream network had increased performance. Furthermore, the objective was also to illustrate the effectiveness of lightweight

---

neural networks for mobile devices. The result showed that networks could work on mobile device effectively. However, network size is a crucial factor both for processing time and the limitation of available memory. To achieve the performance, we had to employ methods similarly used in landmarking, such as Zhang et al. [63], where pooling is key to allowing the network to run efficiently. Also, we note the accuracy and how this impacts the network. A key factor with this work is the real-time implementation of a neural network on devices with severely limited or no GPU and demonstrated an effective system that is in contrast to the vast belief networks require large GPU machine for both training and inference.

The third objective relates to the issue with noisy depth data and the lack of ground truth. As chapter 6 showed, we have managed to generate a large dataset of realistic clean and noisy depth data. We extended the work with an additional loss function that helps improve the performance of some RGB denoisers. However, although statistically we “cleaned” the depth images, in reality, the networks could not learn and understand the depth data enough to remove the noise and generate a 3D model. The issue of denoising depth data has shown to be difficult compared to 2D images, but we have provided solutions to this in future work.

The final section is built upon the work performed and knowledge gained from the prior sections. A network that targets explicitly the purpose of retrieving expression intensity that could be used for expression synthesis. The network uses auxiliary information as landmark and gender. The use of landmarks gives the option to add further personalised expressions over just the given universal. In this work, we also sought out how the network interprets expressions. Thus, we visualise both the convolutions, showing the features being highlighted, and the activation of the neurons, to show how they affect the results.

### 8.3 Future Work

This section of the thesis will highlight and examine prospects of future work.



---

### 8.3.1 Neural Network with Object Detection Integration

Neural networks appeal to research as they can provide high-accuracy end-to-end results. The end-to-end for our work cannot be fully implemented; As for facial landmarking it is naturally required for faces to be cropped and reshaped and sometimes aligned as a pre-processing step. The pre-processing steps mean the system is not fully end-to-end, and the results can be directly affected by the pre-processing methods.

In image segmentation, the issue of pre-processing has been resolved, through the implementation of mask-RCNN. Mask-RCNN uses ResNet [111] architecture to pre-process input images, and generate “heat maps” of possible object locations in an image. The network then implements a unique layer, that identifies the high-light areas and proceeds to segment and resizes the highlighted areas. However, a limitation of this method is that during the resizing, each possible prediction is placed into a new layer and all possible predictions are processed at the same time, not individually. The stacking of the possible predictions increases the memory requirements and means a hard limit is set on the total identifiable objects in a single image is done by the network.

For our work this would allow a system to fully process a facial image, having complete control over its ability to detect and what features to highlight. However, designing and training a system to do this is not a simple task. An additional extension would be to additionally show the effect on mobile devices as newer phone begin to integrate specialised neural network processors.

### 8.3.2 General Adversarial Networks based networks

General Adversarial Networks (GAN) [176] use competing neural networks, one to generate data that can be used as input. The second network acts as a discriminator try to identify if the image received is fake, from the generator, or a real input image. The networks during training compete against each other, allowing

---

the generator to produce realistic training data that is not identical to previous training data.

By using [GAN](#), datasets can be effectively grown substantially, adding more diversity into the training of the network. This method has been shown to drastically improve the performance of neural networks in many areas [[177](#), [178](#), [179](#)]. By including this methodology with ours, a drastic improvement in accuracy could be made.

### 8.3.3 Dataset Expansion and Improvement

The [KOED](#) dataset has provided a significant amount of data that has been used throughout the thesis. The dataset contains fewer participants than some of the more popular datasets as they only use key pose images, but we follow a similar path to the D3DFACS dataset, where we focus on the full expression from neutral to peak and back to neutral. By focusing on the full expression, a small number of participants can match the size of many full-size datasets. However, the dataset requires expansion to add more participants, we have a large number of training, testing and validation images, but requires more participants to ensure the network is diverse.

The dataset is also limited in that it is only annotated for the universal emotion, but for the target of our work, this is not ideal. To improve upon the annotation, external collaboration for MPEG-4 coding would improve the usability of the dataset significantly. Another improvement would be to add in real dynamic expressions rather than just the posed expression we currently employ.

### 8.3.4 Denoising Expansion

Denoising [ToF](#) data even with a newer sensor with high-speed capabilities and greater resolution capture details are still noisy and miss key details. When simulating the Kinect depth data which is at a lower resolution, could capture and

---

been used to generate 3D models with the subtle details of the face. With the production of neural network chips [180] training a neural network to perform denoise is a useful task. The improvements can be performed in a wide array on methods:

- Use inception based architecture
- Train on a larger dataset, with more diverse objects

The work would help in areas that implement consumer based depth sensors. Also, it would help extend the area of de-noising with a neural network as well as synthetic data generation.

### 8.3.5 Integration of Improved Depth Sensing Technology

The Kinect was, and still largely, a drastic improvement in consumer use of depth sensing devices. The Kinect allows cheap access to high-accuracy, real-time and at a useful resolution, which is still largely comparable to more recent sensors, resulting in its diverse application from just a device for gaming. However, modern sensors employ newer technology and theories that can improve upon the resolution, capture rate and reduce the noise from the IR sensor, such as the realsense.

The intel realsense D435 [56], implements dual IR blaster and projector that allow improved accuracy over the Kinect. The sensor can also work at up to 90 FPS vs the Kinects maximum 30 FPS. An improved sensor would drastically change the work performed by:

- Improved 3D landmarks: As the depth data will be cleaner the landmarks acquired will fit the real structure of the face. Clean 3D landmark detection, would improve the reliability of the system and help to ensure fewer movements are missed or falsely predicted due to noise.

- 
- Higher-resolution models: As the sensor allows for higher resolution captures of the face, the output model would be able to capture significantly more detail of the face and its structure.
  - Easier denoising: As shown in chapter 6, the noise produced by the Kinect in capture is complex and on a frame-by-frame basis intense. The intensity of the noise makes denoising a difficult task, but although the theory behind the capture is the same, by having a sensor with less noise could allow an easier learning curve for neural networks.

## 8.4 Concluding Remarks

The work presented in this thesis forms contributions in facial landmarking detection and animation expression prediction using regression. We expand the work of landmarking using [RGBD](#) depth sensors; we developed a series of the network over the work performed in the thesis in which we show in the appendix outperforms state of the art. We also move away from the black-box nature of deep learning, by demonstrating the features the network uses for its predictions. This field is extensive in work done and still has much more possibilities for expansion into [3D](#) using into depth data. As technology continues to improve, we see an increase in consumer-based [RGBD](#) sensor-enabled device. In-which means depth based or assisted landmarking could soon become standard.

# Appendix A

## Comparison of 3D landmarks

We perform a competitive analysis of the network described in chapter 5. Although the works focus on a competitive comparison of the different streams and the effect of auxiliary information, it is vital to allow a comparison to other state of the art methods. We compare against to landmark dedicated networks.

As illustrated in Fig A.1 the most recent work of Zhang et al. [41], performs significantly better than previously done work, when evaluating on MSE. The result is due to the reduction in the amount of pooling and larger images, that

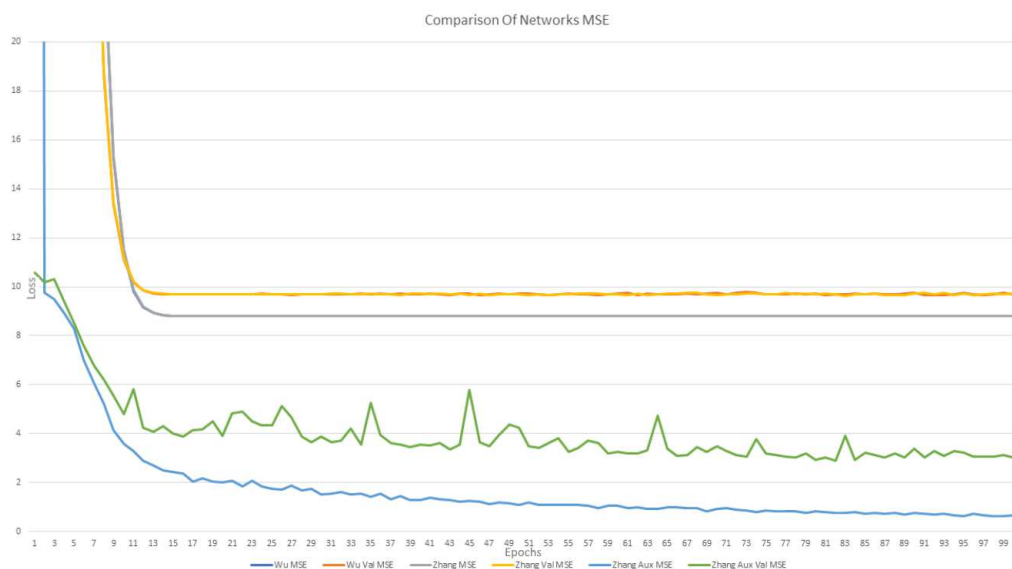


FIGURE A.1: The MSE scores of the network during training

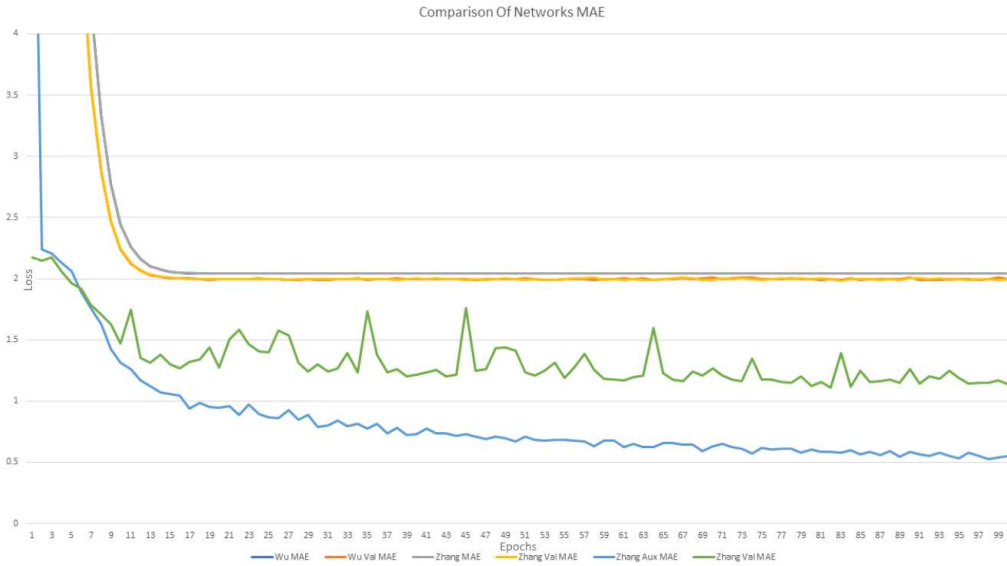


FIGURE A.2: The MAE scores of the network during training

allow the network to retain more information on the face. Furthermore, these improvements are demonstrated in the evaluation of MAE as illustrated in Fig A.2.

Table A.1, Show the results of the testing set on the trained models. The results show similar to Figs A.2, and A.1 as Zhang et al. [41] auxiliary network performs the best. However, on MAE the networks score very similarly. Also, in both MSE and MAE our network outperforms the other networks significantly.

TABLE A.1: A comparison of the test score of the networks and ours.

	Wu Vanilla	Zhang First	Zhang Aux (no transfer)	Our RGB UV
MSE	6.15	6.13	0.88	1.75
MAE	1.84	1.84	1.84	0.96

# Appendix B

## Expression Network Activation

As shown in Fig B.1, each definite expression triggers multiple neurons. However, some of the expression share neuron activations indicating that some on the movement detected by the system could indicate multiple expressions, such as the mouth widening could mean fear, anger or contempt. Also, some of the expressions, are clearly more defined, such as surprise which contains the highest of all the activation indexes. Some of the activations do only focus on single expression, which could be more weight to the decision then neurons that have multiple activations.

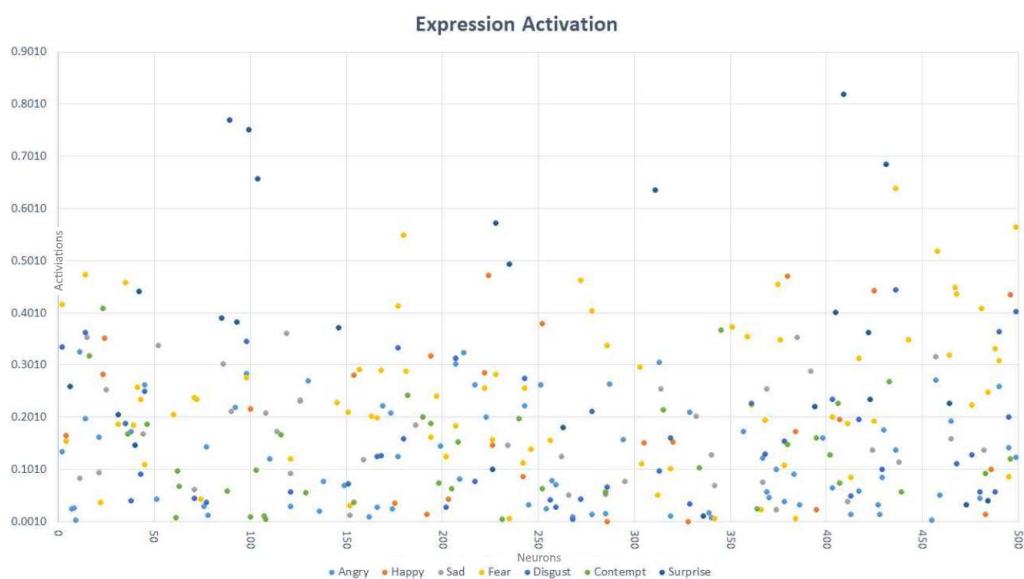


FIGURE B.1: The full graph of the expression networks activation

# Bibliography

- [1] D. Cosker, E. Krumerhuber, and A. Hilton, “A FACS valid 3D dynamic action unit database with applications to 3D dynamic morphable facial modeling,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2296–2303, 2011.
- [2] Faceshift, “faceshift,” 2012. [Online]. Available: <http://www.faceshift.com/>
- [3] Facerig, “Facerig,” Romainia, 2016. [Online]. Available: <https://facerig.com/>
- [4] T. Weise, S. Bouaziz, H. Li, and M. Pauly, “Realtime performance-based facial animation,” *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11*, vol. 1, no. 212, p. 1, 2011. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1964921.1964972>
- [5] T. Weise, H. Li, L. Van Gool, and M. Pauly, “Face/Off: Live Facial Pupperty,” *Symposium on Computer Animation SCA*, p. 7, 2009. [Online]. Available: [http://lgg.epfl.ch/publications/2009/weise\\_{\\_}2009\\_{\\_}LFP.pdf](http://lgg.epfl.ch/publications/2009/weise_{_}2009_{_}LFP.pdf)
- [6] C. Cao and D. Bradley, “Real-Time High-Fidelity Facial Performance Capture,” pp. 1–9, 2015.
- [7] M. Fleischer, “Method of producing moving-picture cartoons.” 1917. [Online]. Available: <https://www.google.com/patents/US1242674>
- [8] B. Robertson, “Mike, the Talking Head,” *Computer graphics world*, vol. 11, no. 7, p. 57, 1988.



- [9] T. B. Moeslund, A. Hilton, and V. Krüger, “A survey of advances in vision-based human motion capture and analysis,” *Computer Vision and Image Understanding*, vol. 104, no. 2-3 SPEC. ISS., pp. 90–126, 2006.
- [10] D. J. Sturman, “A Brief History of Motion Capture for Computer Character Animation,” pp. 3–7, 1999.
- [11] G. O. Paradiso, D. I. Cunic, C. A. Gunraj, and R. Chen, “Representation of facial muscles in human motor cortex,” *Journal of Physiology*, vol. 567, no. 1, pp. 323–336, 2005.
- [12] C. Darwin, *The Expression of the Emotions in Man and Animals*, 1st ed., J. Murray, Ed. London: John Murray, 1872. [Online]. Available: <http://darwin-online.org.uk/content/frameset?pageseq=1&itemID=F1142&viewtype=text>
- [13] S. Tomkins, “Affect Theory,” K. R. Scherer and P. Ekman, Eds. Psychology Press, 1984, ch. 7, pp. 163–195.
- [14] P. Ekman and W. V. Friesen, “Measuring facial movement,” *Environmental Psychology and Nonverbal Behavior*, vol. 1, no. 1, pp. 56–75, 1976.
- [15] P. Ekman, *The face of man: Expressions of universal emotions in a New Guinea village*. Scholarly Title, 1980.
- [16] P. Ekman, W. V. Friesen, and J. C. Hager, “Facs manual,” *A Human Face*, 2002.
- [17] G. N. Matre and S. K. Shah, “Facial expression detection,” in *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–3.
- [18] C. A. Corneanu, M. Oliu, J. F. Cohn, and S. Escalera, “Survey on RGB, 3D, Thermal, and Multimodal Approaches for Facial Expression Recognition: History, Trends, and Affect-related Applications,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8828, no. c, pp. 1–1, 2016.

- [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7374704>
- [19] G. Sandbach, S. Zafeiriou, M. Pantic, and L. Yin, “Static and dynamic 3D facial expression recognition: A comprehensive survey,” *Image and Vision Computing*, vol. 30, no. 10, pp. 683–697, 2012. [Online]. Available: <http://dx.doi.org/10.1016/j.imavis.2012.06.005>
- [20] A. F. Abate, M. Nappi, D. Riccio, and G. Sabatino, “2D and 3D face recognition: A survey,” *Pattern Recognition Letters*, vol. 28, no. 14, pp. 1885–1906, 2007.
- [21] Y. Wu and Q. Ji, “Facial Landmark Detection: A Literature Survey,” *International Journal of Computer Vision*, no. November 2016, pp. 1–28, 2018. [Online]. Available: <https://doi.org/10.1007/s11263-018-1097-z>
- [22] P. Viola and M. J. Jones, “Robust real-time face detection,” *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.
- [23] R. Lienhart, A. Kuranov, and V. Pisarevsky, “Empirical analysis of detection cascades of boosted classifiers for rapid object detection,” in *Pattern Recognition*. Springer, 2003, pp. 297–304.
- [24] C. Kendrick, “Towards Real-Time Facial Landmark Detection in Depth Data Using Auxiliary Information,” pp. 1–18, 2018.
- [25] S. Rusinkiewicz, “Efficient Variants of the ICP Algorithm a r c Levoy,” 2001.
- [26] V. Blanz and T. Vetter, “A morphable model for the synthesis of 3D faces,” *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH '99*, pp. 187–194, 1999. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=311535.311556>
- [27] D. Vlastic, M. Brand, H. Pfister, and J. Popović, “Face transfer with multilinear models,” *ACM SIGGRAPH 2005 Papers on - SIGGRAPH '05*, vol. 24, no. 3, p. 426, 2005. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1186822.1073209>

- [28] C. Cao, Y. Weng, S. Lin, and K. Zhou, “3D Shape Regression for Real-time Facial Animation,” *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 41:1—41:10, 2013. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2461912.2462012>
- [29] C. Cao, Q. Hou, and K. Zhou, “Displaced dynamic expression regression for real-time facial tracking and animation,” *ACM Transactions on Graphics*, vol. 33, no. 4, pp. 1–10, 2014. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2601097.2601204>
- [30] Y. Liu, Y. Cao, Y. Li, M. Liu, R. Song, Y. Wang, Z. Xu, and X. Ma, “Facial expression recognition with PCA and LBP features extracting from active facial patches,” *2016 IEEE International Conference on Real-Time Computing and Robotics, RCAR 2016*, pp. 368–373, 2016.
- [31] T. Cootes, “Active Appearance Models,” *Pattern Analysis and . . .*, vol. 23, no. 6, pp. 681–685, 2001.
- [32] T. Cootes, C. Taylor, D. Cooper, and J. Graham, “Active Shape Models—Their Training and Application,” pp. 38–59, 1995. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S1077314285710041>
- [33] V. Kazemi and J. Sullivan, “One millisecond face alignment with an ensemble of regression trees,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1867–1874, 2014.
- [34] Y. Sun, X. Wang, and X. Tang, “Deep convolutional network cascade for facial point detection,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3476–3483, 2013.
- [35] E. Zhou, H. Fan, Z. Cao, Y. Jiang, and Q. Yin, “Extensive facial landmark localization with coarse-to-fine convolutional network cascade,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 386–391, 2013.

- [36] H. Liu, J. Lu, J. Feng, and J. Zhou, “Learning Deep Sharable and Structural Detectors for Face Alignment,” *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1–1, 2017. [Online]. Available: <http://ieeexplore.ieee.org/document/7829264/>
- [37] H. Lai, S. Xiao, Y. Pan, Z. Cui, J. Feng, C. Xu, J. Yin, and S. Yan, “Deep Recurrent Regression for Facial Landmark Detection,” pp. 1–13, 2015. [Online]. Available: <http://arxiv.org/abs/1510.09083>
- [38] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [39] C.-h. D. Wu and G. Wetzstein, “Automated Restyling of Human Portrait Based on Facial Expression Recognition and 3D Reconstruction.”
- [40] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, “Facial landmark detection by deep multi-task learning,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8694 LNCS, no. PART 6, pp. 94–108, 2014.
- [41] —, “Learning Deep Representation for Face Alignment with Auxiliary Attributes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 5, pp. 918–930, 2016.
- [42] A. Jourabloo and X. Liu, “Large-Pose Face Alignment via CNN-Based Dense 3D Model Fitting,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4188–4196, 2016. [Online]. Available: <http://ieeexplore.ieee.org/document/7780823/>
- [43] R. Ranjan, V. M. Patel, and R. Chellappa, “HyperFace: A Deep Multi-task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition,” vol. XX, no. Xx, pp. 1–16, 2016. [Online]. Available: <http://arxiv.org/abs/1603.01249>
- [44] T. B. Laboratories, M. Avenue, and M. Hill, “Random Decision Forests Tin Kam Ho Perceptron training,” 1995.

- [45] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min, and W. Worek, "Overview of the Face Recognition Grand Challenge," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 947–954. [Online]. Available: <http://dx.doi.org/10.1109/CVPR.2005.268>
- [46] P. Hall, P. W. Editors, M. Sánchez, L. James, S. A. King, and S. Maddock, "Use and Re-use of Facial Motion Capture Data," pp. 1–8, 2003.
- [47] E. Sifakis, I. Neverov, and R. Fedkiw, "Automatic determination of facial muscle activations from sparse motion capture marker data," *ACM Transactions on Graphics*, vol. 24, no. 3, p. 417, 2005.
- [48] M. Gleicher, "Animation from observation," *ACM SIGGRAPH Computer Graphics*, vol. 33, no. 4, pp. 51–54, 1999.
- [49] Z. Deng and U. Pei-Ying Chiang USC Pamela Fox USC Ulrich Neumann USC, "Animating Blendshape Faces by Cross-Mapping Motion Capture Data," *I3D*, vol. 1, no. March, p. 7, 2006.
- [50] H. Huang, J. Chai, X. Tong, and H.-T. Wu, "Leveraging motion capture and 3D scanning for high-fidelity facial performance acquisition," *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11*, vol. 1, no. 212, p. 1, 2011. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1964921.1964969>
- [51] P. Debevec, "The Light Stages and Their Applications to Photoreal Digital Actors," *SIGGRAPH Asia*, pp. 1998–2001, 2012. [Online]. Available: <http://gl.ict.usc.edu/LightStages/SIGGRAPHAsia-2012-Debevec-LightStages.pdf>
- [52] S. Li, K. N. Ngan, R. Paramesran, and L. Sheng, "Real-Time Head Pose Tracking with Online Face Template Reconstruction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1922–1928, 2016.

- [53] V. Technologies, “MPEG-4 Face and Body Animation ( MPEG-4 FBA ) An overview,” *The Character nimation Company*, pp. 1–66, 2015. [Online]. Available: [www.visagetechologies.com](http://www.visagetechologies.com)
- [54] M. Proesmans and L. Van Gool, “Reading between the linesa method for extracting dynamic 3D with texture,” *Proceedings of the ACM symposium on Virtual reality software and technology*, pp. 95–102, 1997. [Online]. Available: <http://dl.acm.org/citation.cfm?id=261154>  
[http://ieeexplore.ieee.org/ielx4/5756/15374/00710851.pdf?tp={&}arnumber=710851{&}isnumber=15374](http://ieeexplore.ieee.org/ielx4/5756/15374/00710851.pdf?tp=&arnumber=710851&isnumber=15374)
- [55] V. M. capture Systems, “Vicon,” 2016. [Online]. Available: <http://www.vicon.com/>
- [56] Intel, “Intel Realsense D435,” 2018.
- [57] P. Dai, X. Wang, and W. Zhang, “Coarse-to-fine multiview 3d face reconstruction using multiple geometrical features,” *Multimedia Tools and Applications*, 2017. [Online]. Available: <http://dx.doi.org/10.1007/s11042-016-4325-y>
- [58] J. Roth, Y. Tong, and X. Liu, “Adaptive 3D Face Reconstruction from Unconstrained Photo Collections,” pp. 4197–4206, 2016.
- [59] X. Zhu, J. Yan, D. Yi, Z. Lei, and S. Z. Li, “Discriminative 3D morphable model fitting,” *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition, FG 2015*, 2015.
- [60] P. Szeptycki, M. Ardabilian, and L. Chen, “A coarse-to-fine curvature analysis-based rotation invariant 3D face landmarking,” *IEEE 3rd International Conference on Biometrics: Theory, Applications and Systems, BTAS 2009*, pp. 1–6, sep 2009.
- [61] S. Zhang, H. Yu, J. Dong, T. Wang, Z. Ju, and H. Liu, “Automatic Reconstruction of Dense 3D Face Point Cloud with a Single Depth Image,”

*Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, pp. 1439–1444, 2016.

- [62] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” *Machine Learning: Proc. of the 13th Int. Conf.*, pp. 148–156, 1996.
- [63] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks,” *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.
- [64] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 2980–2988, 2017.
- [65] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool, “Random Forests for Real Time 3D Face Analysis,” *International Journal of Computer Vision*, vol. 101, no. 3, pp. 437–458, 2013.
- [66] H. C. Burger, C. J. Schuler, and S. Harmeling, “Image denoising: Can plain neural networks compete with BM3D?” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2392–2399, 2012.
- [67] K. Dabov and A. Foi, “Image Denoising with Block-matching and {3D} Filtering,” *Electronic Imaging*, vol. 6064, pp. 1–12, 2006. [Online]. Available: <http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=728740>
- [68] C. Dong, C. C. Loy, K. He, and X. Tang, “Image Super-Resolution Using Deep Convolutional Networks,” pp. 1–14, 2014. [Online]. Available: <http://arxiv.org/abs/1501.00092>
- [69] J. Dong, X.-J. Mao, C. Shen, and Y.-B. Yang, “Learning Deep Representations Using Convolutional Auto-encoders with Symmetric Skip Connections,” pp. 1–17, 2016. [Online]. Available: <http://arxiv.org/abs/1611.09119>

- [70] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network,” 2016. [Online]. Available: <http://arxiv.org/abs/1609.04802>
- [71] D. King, “Dlib,” 2013. [Online]. Available: <http://dlib.net/>
- [72] D. Cristinacce and T. F. Cootes, “Feature Detection and Tracking with Constrained Local Models,” *Proceedings of the British Machine Vision Conference 2006*, pp. 95.1–95.10, 2006. [Online]. Available: <http://www.bmva.org/bmvc/2006/papers/024.html>
- [73] T. Baltrusaitis, P. Robinson, and L. P. Morency, “OpenFace: An open source facial behavior analysis toolkit,” *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016*, 2016.
- [74] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L. P. Morency, “OpenFace 2.0: Facial behavior analysis toolkit,” *Proceedings - 13th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2018*, pp. 59–66, 2018.
- [75] I. Megvii, “Face Plus Plus,” 2015. [Online]. Available: <http://www.faceplusplus.com/about/>
- [76] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.
- [77] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” pp. 1–15, 2014. [Online]. Available: <http://arxiv.org/abs/1412.6980>
- [78] F. Technologies, “faceware,” 2015. [Online]. Available: <http://facewaretech.com/>
- [79] G. Breton, “Dynamixyz,” France, 2003. [Online]. Available: <http://www.dynamixyz.com/>



- [80] D. Cristinacce and T. F. Cootes, “Boosted Regression Active Shape Models,” *Proceedings of the British Machine Vision Conference 2007*, pp. 79.1–79.10, 2007. [Online]. Available: <http://www.bmva.org/bmvc/2007/papers/paper-131.html>
- [81] A. K. Davison, C. Lansley, N. Costen, K. Tan, and M. H. Yap, “SAMM: A Spontaneous Micro-Facial Movement Dataset,” *IEEE Transactions on Affective Computing*, vol. XX, no. X, pp. 1–1, 2016. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7492264>
- [82] C. C. Ng, M. H. Yap, N. Costen, and B. Li, “Will Wrinkle Estimate the Face Age?” *Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, pp. 2418–2423, 2016.
- [83] Autodesk, “Maya,” 2016. [Online]. Available: <http://www.autodesk.co.uk/store/products/maya>
- [84] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, “Face detection without bells and whistles,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8692 LNCS, no. PART 4, pp. 720–735, 2014.
- [85] C. Wang, Y. Zeng, L. Simon, I. Kakadiaris, D. Samaras, and N. Paragios, “Viewpoint invariant 3D landmark model inference from monocular 2D images using higher-order priors,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 319–326, 2011.
- [86] Microsoft, “Microsoft Kinect 360,” 2010. [Online]. Available: <http://www.xbox.com/en-US/xbox-360/accessories/kinect>
- [87] “Microsoft”, “Microsoft Kinect,” 2013. [Online]. Available: <http://www.xbox.com/en-GB/xbox-one/accessories/kinect-for-xbox-one{#}fbid=3xbNWWc6k4c>
- [88] S. Zulqarnain, G. Faisal, and S. Ajmal, “Shape-based Automatic Detection of a Large Number of 3D Facial Landmarks,” *Cvpr*, 2015.

- [89] T. Franken, M. Dellepiane, F. Ganovelli, P. Cignoni, C. Montani, and R. Scopigno, “Minimizing user intervention in registering 2D images to 3D models,” *Visual Computer*, vol. 21, no. 8-10, pp. 619–628, 2005.
- [90] ”Autodesk”, “3DS Max,” 2016. [Online]. Available: <http://www.autodesk.co.uk/store/products/3ds-max?term=1year{&}support=advanced>
- [91] U. Technologies, “Unity3D,” United States, 2016. [Online]. Available: <https://unity3d.com/>
- [92] D. M. W. Powers, “Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation,” no. December, p. 24, 2007.
- [93] Canming, “EMGU,” *Canming. (2008). EMGU. Retrieved from* [http://www.emgu.com/wiki/index.php/Main\\_Page](http://www.emgu.com/wiki/index.php/Main_Page), 2008. [Online]. Available: [http://www.emgu.com/wiki/index.php/Main{\\_{}}Page](http://www.emgu.com/wiki/index.php/Main_{_}Page)
- [94] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou, “FaceWarehouse: A 3D facial expression database for visual computing,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 3, pp. 413–425, 2014.
- [95] S. Issue-regular and M. Communications, “KinectFaceDB : a Kinect Face Database for Face Recognition,” *Review*, pp. 1–15, 2013.
- [96] R. I. Hg, P. Jasek, C. Rofidal, K. Nasrollahi, T. B. Moeslund, and G. Tranchet, “An RGB-D database using microsoft’s kinect for windows for face detection,” *8th International Conference on Signal Image Technology and Internet Based Systems, SITIS 2012r*, no. 7, pp. 42–46, 2012.
- [97] N. Erdogmus and S. Marcel, “Spoofing in 2D face recognition with 3D masks and anti-spoofing with Kinect,” *IEEE 6th International Conference on Biometrics: Theory, Applications and Systems, BTAS 2013*, 2013.
- [98] K. Khoshelham, “Accuracy Analysis of Kinect Depth Data,” *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XXXVIII-5/, no. August, pp. 133–138, 2012.

- [99] D. I. Di and I. N. Dell, "Evaluation of Microsoft Kinect 360 and Microsoft Kinect One for robotics and computer vision applications," 2015.
- [100] C. Amon and F. Fuhrmann, "Evaluation of the Spatial Resolution Accuracy of the Face Tracking System for Kinect for Windows V1 and V2," *6th Congress of Alps-Adria Acoustics Assosiation*, no. October, pp. 9–12, 2014.
- [101] D. Chauhan, K. Umaria, and K. Dave, "3D Face databases : A review," vol. 5, no. 10, pp. 379–384, 2015.
- [102] M. Goyal, N. D. Reeves, A. K. Davison, S. Rajbhandari, J. Spragg, and M. H. Yap, "DFUNet: Convolutional Neural Networks for Diabetic Foot Ulcer Classification," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. PP, pp. 1–12, 2017. [Online]. Available: <http://arxiv.org/abs/1711.10448>
- [103] M. Goyal, N. Reeves, S. Rajbhandari, and M. H. Yap, "Robust Methods for Real-time Diabetic Foot Ulcer Detection and Localization on Mobile Devices," *IEEE Journal of Biomedical and Health Informatics*, pp. 1–12, 2018.
- [104] H. M. Bui, M. Lech, E. Cheng, K. Neville, and I. S. Burnett, "Using grayscale images for object recognition with convolutional-recursive neural network," *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, pp. 321–325, 2016.
- [105] X. Han, M. H. Yap, and I. Palmer, "Face recognition in the presence of expressions," *Journal of Software Engineering and Applications*, vol. 5, no. 05, p. 321, 2012.
- [106] Z. H. Feng and J. Kittler, "Advances in facial landmark detection," *Biometric Technology Today*, vol. 2018, no. 3, pp. 8–11, 2018. [Online]. Available: [http://dx.doi.org/10.1016/S0969-4765\(18\)30038-9](http://dx.doi.org/10.1016/S0969-4765(18)30038-9)

- [107] E. M. Hand and R. Chellappa, “Attributes for Improved Attributes: A Multi-Task Network for Attribute Classification,” pp. 4068–4074, 2016. [Online]. Available: <http://arxiv.org/abs/1604.07360>
- [108] P. J. Angeline, P. J. Angeline, G. M. Saunders, G. M. Saunders, J. B. Pollack, and J. B. Pollack, “An evolutionary algorithm that constructs recurrent neural networks.” *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, vol. 5, no. 1, pp. 54–65, 1994. [Online]. Available: <http://www.ncbi.nlm.nih.gov/pubmed/18267779>
- [109] H. Dibeklioglu, A. A. Salah, and L. Akarun, “2008\_3D Facial Landmarking under Expression, Pose, and Occlusion Variations,” vol. 00, pp. 3–8.
- [110] P. Nair and A. Cavallaro, “3-D Face Detection, Landmark Localization, and Registration Using a Point Distribution Model,” *Multimedia, IEEE Transactions on*, vol. 11, no. 4, pp. 611–623, 2009. [Online]. Available: <http://ieeexplore.ieee.org/ielx5/6046/4939467/04907232.pdf?tp={&}arnumber=4907232{&}isnumber=4939467{&}5Cnhttp://ieeexplore.ieee.org/stamp/stamp.jsp?tp={&}arnumber=4907232>
- [111] C. Szegedy, S. Ioffe, and V. Vanhoucke, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” *Arxiv*, p. 12, 2016.
- [112] W. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, and Y. Chen, “Compressing Neural Networks with the Hashing Trick,” *Proceedings of The 32nd International Conference on Machine Learning*, vol. 37, pp. 2285–2294, 2015. [Online]. Available: <http://arxiv.org/abs/1504.04788>
- [113] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin, “Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications,” *Iclr*, pp. 1–16, 2016. [Online]. Available: <http://arxiv.org/abs/1511.06530>
- [114] A. J. Robinson, “An Application of Recurrent Nets to Phone Probability Estimation,” *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 298–305, 1994.

- [115] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, “Multimodal Deep Learning,” *Proceedings of The 28th International Conference on Machine Learning (ICML)*, pp. 689–696, 2011.
- [116] A. C. Eunbyung Park, Xufeng Han, Tamara L. Berg, “Combining Multiple Sources of Knowledge in Deep CNNs for Action Reco,” *Journal of Chemical Information and Modeling*, vol. 53, no. 9, pp. 1689–1699, 2013.
- [117] R. Socher and B. Huval, “Convolutional-recursive deep learning for 3D object classification,” *Advances in Neural ...*, no. i, pp. 1–9, 2012. [Online]. Available: <http://papers.nips.cc/paper/4773-convolutional-recursive-deep-learning-for-3d-object-classification.pdf>  
[http://machinelearning.wustl.edu/mlpapers/paper\\_{\\_}files/NIPS2012\\_{\\_}0304.pdf](http://machinelearning.wustl.edu/mlpapers/paper_{_}files/NIPS2012_{_}0304.pdf)
- [118] L. Liu and L. Shao, “Learning discriminative representations from RGB-D video data,” *IJCAI International Joint Conference on Artificial Intelligence*, pp. 1493–1500, 2013.
- [119] D. P. Kingma, “A : a m s o,” pp. 1–15, 2015.
- [120] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, G. Brain, I. Osdi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow : A System for Large-Scale Machine Learning,” *Osdi*, 2016.
- [121] A. M. Albert and K. Ricanek Jr, “The MORPH database: investigating the effects of adult craniofacial aging on automated face-recognition technology,” *Forensic Science Communications*, vol. 10, no. 2, 2008.

- [122] A. Bulat and G. Tzimiropoulos, “How far are we from solving the 2D & 3D Face Alignment problem? (and a dataset of 230,000 3D facial landmarks),” 2017. [Online]. Available: <http://arxiv.org/abs/1703.07332>
- [123] I. Corporation, W. Garage, and Itseez, “OpenCV,” 2000. [Online]. Available: <http://opencv.org/>
- [124] X. Zhu and D. Ramanan, “Face detection, pose estimation, and landmark localization in the wild,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2879–2886, 2012.
- [125] D. Learning, “Deep Learning for Consumer Devices and Services,” no. APRIL, 2017.
- [126] J. Schmidhuber, “Deep Learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015. [Online]. Available: <http://dx.doi.org/10.1016/j.neunet.2014.09.003>
- [127] G. Inc, “Android Studio,” 2017. [Online]. Available: <https://developer.android.com/studio/index.html>
- [128] C. Kendrick, K. Tan, K. Walker, and M. H. Yap, “The Application of Neural Networks for Facial Landmarking on Mobile Devices,” in *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4: VISAPP,, INSTICC. SciTePress*, 2018, pp. 189–197.
- [129] F. Chollet, “Keras,” Mountain View, California, 2016. [Online]. Available: <https://github.com/fchollet/keras>
- [130] Google, “Google Play Store,” 2017. [Online]. Available: <https://play.google.com/store>
- [131] C. Kendrick, K. Tan, T. Williams, and M. H. Yap, “An Online Tool for the Annotation of 3D Models,” pp. 362–369, 2017.
- [132] A. Fagg, S. Lucey, and S. Sridharan, “Fast , Dense Feature SDM on an iPhone,” pp. 95–102, 2017.

- [133] R. Ranjan, S. Sankaranarayanan, C. D. Castillo, and R. Chellappa, “An All-In-One Convolutional Neural Network for Face Analysis,” pp. 17–24, 2017.
- [134] A. Savran, B. Sankur, and M. Taha Bilge, “Comparative evaluation of 3D vs. 2D modality for automatic detection of facial action units,” *Pattern Recognition*, vol. 45, no. 2, pp. 767–782, 2012.
- [135] J. Wang, L. Yin, X. Wei, and Y. Sun, “3D facial expression recognition based on primitive surface feature distribution,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 1399–1406, 2006.
- [136] M. Ma, X. Hu, Y. Xu, and S. Peng, “A lighting robust fitting approach of 3d morphable model using spherical harmonic illumination,” *Proceedings - International Conference on Pattern Recognition*, pp. 2101–2106, 2014.
- [137] K. W. Bowyer, K. Chang, and P. Flynn, “A survey of approaches and challenges in 3D and multi-modal 3D+2D face recognition,” *Computer Vision and Image Understanding*, vol. 101, no. 1, pp. 1–15, 2006. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S1077314205000822>
- [138] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar, “Acquiring the reflectance field of a human face,” *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*, pp. 145–156, 2000. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=344779.344855>
- [139] H. Rahmani and A. Mian, “3D Action Recognition from Novel Viewpoints,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1506–1515, 2016. [Online]. Available: <http://ieeexplore.ieee.org/document/7780536/>

- [140] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Nets,” *Advances in Neural Information Processing Systems 27*, pp. 2672–2680, 2014. [Online]. Available: <http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf>
- [141] Z. Zhang, Y. Song, and H. Qi, “Age Progression/Regression by Conditional Adversarial Autoencoder,” 2017. [Online]. Available: [http://web.eecs.utk.edu/~zzhang61/docs/papers/2017\\_{-}CVPR\\_{-}Age.pdf](http://web.eecs.utk.edu/~zzhang61/docs/papers/2017_{-}CVPR_{-}Age.pdf)
- [142] N. M. Difilippo, M. K. Jouaneh, and S. Member, “Characterization of Different Microsoft Kinect Sensor Models,” vol. 15, no. 8, pp. 4554–4564, 2015.
- [143] K. Essmaeel, L. Gallo, E. Damiani, G. De Pietro, and A. Dipand??, “Temporal denoising of Kinect depth data,” *8th International Conference on Signal Image Technology and Internet Based Systems, SITIS 2012r*, pp. 47–52, 2012.
- [144] M. A. Farooque and J. S. Rohankar, “Survey on Various Noises and Techniques for Denoising the Color Image,” *International Journal of Application or Innovation in Engineering and Management*, vol. 2, no. 11, pp. 217–221, 2013.
- [145] J. Diebel and S. Thrun, “An application of markov random fields to range sensing,” *Advances in neural information processing systems*, vol. 18, pp. 291–298, 2005. [Online]. Available: <http://papers.nips.cc/paper/2837-an-application-of-markov-random-fields-to-range-sensing>
- [146] Q. Yang, R. Yang, J. Davis, and D. Nistr, “Spatial-Depth Supre Resolution for Range Images,” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1 – 8, 2007.
- [147] D. Chan, H. Buisman, C. Theobalt, and S. Thrun, “A Noise-Aware Filter for Real-Time Depth Upsampling,” *ECCV Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications*, pp. 1–12, 2008.



- [148] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele, “Joint bilateral upsampling,” *ACM Transactions on Graphics*, vol. 26, no. 3, p. 96, 2007. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1276377.1276497>
- [149] J. Park, H. Kim, Yu-Wing Tai, M. S. Brown, and I. Kweon, “High quality depth map upsampling for 3D-TOF cameras,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1623–1630, 2011.
- [150] K. He, J. Sun, and X. Tang, “Guided image filtering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.
- [151] C. Wu, M. Zollhöfer, M. Nießner, M. Stamminger, S. Izadi, and C. Theobalt, “Real-time shading-based refinement for consumer depth cameras,” *ACM Transactions on Graphics*, vol. 33, no. 6, pp. 1–10, 2014. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2661229.2661232>
- [152] X. Zhang and R. Wu, “Fast depth image denoising and enhancement using a deep convolutional network,” *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2499–2503, 2016. [Online]. Available: <http://ieeexplore.ieee.org/document/7472127/>
- [153] Y. Xu, T. Mo, Q. Feng, P. Zhong, M. Lai, and E. I.-C. Chang, “Deep learning of feature representation with multiple instance learning for medical image analysis,” *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, no. 1, pp. 1626–1630, 2014. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6853873>
- [154] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2016.
- [155] X.-J. Mao, C. Shen, and Y.-B. Yang, “Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip

- Connections,” no. Nips, 2016. [Online]. Available: <http://arxiv.org/abs/1603.09056>
- [156] S. Nah, T. H. Kim, and K. M. Lee, “Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring,” 2016. [Online]. Available: <http://arxiv.org/abs/1612.02177>
- [157] W. Bae, J. Yoo, and J. C. Ye, “Beyond Deep Residual Learning for Image Restoration: Persistent Homology-Guided Manifold Simplification,” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2017-July, pp. 1141–1149, 2017.
- [158] B. Mildenhall, J. T. Barron, J. Chen, D. Sharlet, R. Ng, and R. Carroll, “Burst Denoising with Kernel Prediction Networks,” 2017. [Online]. Available: <http://arxiv.org/abs/1712.02327>
- [159] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, “BlenSor: Blender Sensor Simulation Toolbox,” vol. 6939, no. Isvc, pp. 199–208, 2011.
- [160] J. D’Errico, “Surface fitting using gridfit,” 2016. [Online]. Available: <https://uk.mathworks.com/matlabcentral/fileexchange/8998-surface-fitting-using-gridfit>
- [161] J. Stam, “Diffraction Shaders,” *In Practice*, pp. 101–110, 1991.
- [162] E. Games, “Unreal Engine 4,” 2016.
- [163] J. Sell and P. O. Connor, “The Xbox One System on a Chip and Kinect Sensor,” pp. 44–53, 2014.
- [164] O. Wasenm, “Comparison of Kinect v1 and v2 Depth Images in Terms of Accuracy and Precision,” pp. 1–12, 2016.
- [165] B. Choo, M. Landau, M. DeVore, and P. Beling, “Statistical Analysis-Based Error Models for the Microsoft Kinect™ Depth Sensor,” *Sensors*, vol. 14, no. 9, pp. 17 430–17 450, 2014. [Online]. Available: <http://www.mdpi.com/1424-8220/14/9/17430/>

- [166] K. Khoshelham and S. O. Elberink, “Accuracy and resolution of kinect depth data for indoor mapping applications,” *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.
- [167] T. Bagautdinov, C. Wu, J. Saragih, P. Fua, and Y. Sheikh, “Modeling Facial Geometry using Compositional VAEs,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018)*, pp. 3877–3886, 2018. [Online]. Available: [http://openaccess.thecvf.com/content\\_{-}cvpr\\_{-}2018/papers/Bagautdinov\\_{-}Modeling\\_{-}Facial\\_{-}Geometry\\_{-}CVPR\\_{-}2018\\_{-}paper.pdf](http://openaccess.thecvf.com/content_{-}cvpr_{-}2018/papers/Bagautdinov_{-}Modeling_{-}Facial_{-}Geometry_{-}CVPR_{-}2018_{-}paper.pdf)
- [168] N. Zeng, H. Zhang, B. Song, W. Liu, Y. Li, and A. M. Dobaie, “Facial expression recognition via learning deep sparse autoencoders,” *Neurocomputing*, vol. 273, pp. 643–649, 2018.
- [169] R. Walecki, O. Rudovic, V. Pavlovic, B. Schuller, and M. Pantic, “Deep structured learning for facial action unit intensity estimation,” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5709–5718, 2017.
- [170] O. F. Osman, R. M. I. Elbashir, I. E. Abbass, C. Kendrick, M. Goyal, and M. H. Yap, “Automated Assessment of Facial Wrinkling: a case study on the effect of smoking,” pp. 1081–1086, 2017. [Online]. Available: <http://arxiv.org/abs/1708.01844> <http://dx.doi.org/10.1109/SMC.2017.8122755>
- [171] A. K. Davison, M. H. Yap, and C. Lansley, “Micro-Facial Movement Detection Using Individualised Baselines and Histogram-Based Descriptors,” *Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, pp. 1864–1869, 2016.
- [172] G. L. Hassner and Tal, “Age and Gender Classification using Convolutional Neural Networks,” *2008 8th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2008*, vol. 24, no. 3, pp. 2622–2629, 2015. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4761364> <http://portal.acm.org/citation>.

cfm?doid=1180639.1180711{ }5Cnhttp://www.ncbi.nlm.nih.gov/pubmed/21193381{ }5Cnhttp://dx.doi.org/10.1016/j.patcog.2012.09.011

- [173] Visage, “Visage,” 2018. [Online]. Available: <https://www.visagetechologies.com>
- [174] Autodesk, “Autodesk Character Generator,” 2014. [Online]. Available: <https://charactergenerator.autodesk.com/>
- [175] Mixamo, “Face Plus,” San Francisco, 2016. [Online]. Available: <https://www.mixamo.com/faceplus>
- [176] R. Huang, S. Zhang, T. Li, and R. He, “Beyond Face Rotation: Global and Local Perception GAN for Photorealistic and Identity Preserving Frontal View Synthesis,” *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 2458–2467, 2017.
- [177] A. Odena, C. Olah, and J. Shlens, “Conditional Image Synthesis With Auxiliary Classifier GANs,” 2016. [Online]. Available: <http://arxiv.org/abs/1610.09585>
- [178] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks,” *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 2242–2251, 2017.
- [179] Z. Yi, H. Zhang, P. Tan, and M. Gong, “DualGAN: Unsupervised Dual Learning for Image-to-Image Translation,” *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 2868–2876, 2017.
- [180] N. P. Jouppi, A. Borchers, R. Boyle, P.-I. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, C. Young, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C. R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, N. Patil, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, D. Killebrew, A. Koch, N. Kumar,

S. Lacy, J. Laudon, J. Law, D. Patterson, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, G. Agrawal, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, J. Ross, M. Ross, A. Salek, R. Bajwa, E. Samadiani, C. Severn, G. Sizikov, M. Snelham, J. Souter, D. Steinberg, A. Swing, M. Tan, G. Thorson, B. Tian, S. Bates, H. Toma, E. Tuttle, V. Vasudevan, R. Walter, W. Wang, E. Wilcox, D. H. Yoon, S. Bhatia, and N. Boden, “In-Datacenter Performance Analysis of a Tensor Processing Unit,” *ACM SIGARCH Computer Architecture News*, vol. 45, no. 2, pp. 1–12, 2017. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=3140659.3080246>