ARISTOTLE UNIVERSITY OF THESSALONIKI

DEPARTMENT OF INFORMATICS

CREATION OF A FACIAL EXPRESSIONS DATASET COMPRISING OF AUDIOVISUAL DATA AND 3D MODELS

Written by: Georgia Michou,
ID: 3828

Supervisor:

Nikolaos Nikolaidis, Associate Professor

«Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα πτυχιακή εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στο πλαίσιο αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.»

THANKS

The present thesis was carried out at the Aristotle University of Thessaloniki at the Department of Informatics, Faculty of Sciences. I would like to thank my professor, Associate Professor Nikolaidis Nikolaos for the trust he showed me for the assignment of this thesis and for the continuous and direct support that he offered me throughout the entire duration of it. Also a big thank you is due to Charis Simeonidis (PhD student of the Department) for his help and the excellent cooperation we had on the research and programming parts.

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία στοχεύει στη δημιουργία ενός ολοκληρωμένου οπτικοακουστικού συνόλου δεδομένων εκφράσεων προσώπου που περιλαμβάνει κινούμενα τρισδιάστατα μοντέλα μαζί με εικόνες και βίντεο που έχουν καταγραφεί από διάφορες οπτικές γωνίες και αποστασεις. Ένα επόμενο στάδιο εστιάζει στην εφαρμογή μιας μεθόδου που έχει αναπτυχθεί για την αναγνώριση εκφράσεων στο παράγωγο σύνολο δεδομένων προκειμένου να μελετηθεί η επίδραση διαφορετικών παραμέτρων όπως οι συνθήκες φωτισμού, οι θέσεις των καμερών και οι αποστάσεις στην ακρίβεια και απόδοση του αλγορίθμου.

Αρχικά, πραγματοποιήθηκε αναζήτηση σε σύνολα δεδομένων που απεικονίζουν άτομα με διαφορετικές εκφράσεις προσώπου/συναισθήματα μέσω βίντεο. Μεταξύ αυτών που εξετάστηκαν, η τελική επιλογή ήταν το σύνολο δεδομένων RAVDESS καθώς περιλαμβάνει τόσο οπτικά όσο και ηχητικά δεδομένα.

Μετά τη μετατροπή των δεδομένων RAVDESS σε μορφή κατάλληλη για την επόμενη φάση, η ανάπτυξη χωρίστηκε σε τρία βασικά συστατικά. Το πρώτο από αυτά περιελάμβανε τη δημιουργία ενός συνόλου δεδομένων από μια σειρά τρισδιάστατων μοντέλων, τα οποία ανακατασκευάστηκαν από καρέ βίντεο χρησιμοποιώντας δύο διαφορετικές προσεγγίσεις: είτε θεωρώντας ένα πρόσωπο και το αντίστοιχο συναίσθημα όπως παρουσιάζεται στη βάση δεδομένων είτε συνδυάζοντας τη γεωμετρία και την υφή του προσώπου ενός ατόμου με το συναίσθημα ενός άλλου ατόμου. Αυτό το στάδιο υλοποιήθηκε χρησιμοποιώντας το λογισμικό DECA.

Τα τρισδιάστατα μοντέλα ενσωματώθηκαν στον προσομοιωτή Webots για να δημιουργηθούν κινούμενες απεικονίσεις των προσώπων που αντιπροσωπεύουν τα διαφορετικά συναισθήματα. Κατά τη διάρκεια αυτής της διαδικασίας, υλοποιήθηκε κώδικας που καταγράφει καρέ βίντεο του ανθρώπινου κεφαλιού. Τα καρέ συλλέχθηκαν από ένα πλέγμα εικονικών καμερών, τοποθετημένων σε συνολικά 25 γωνίες (-60°...+60° στο pan με βήματα 30° και -30°...+30° στο tilt με αυξήσεις 15°) και 3 αποστάσεις (0.5, 0.75 και 1 μέτρο). Επιπλέον, για να προσομοιώσουν διαφορετικές συνθήκες φωτισμού, οι εικόνες βίντεο συλλέχθηκαν σε "σκοτεινές" και "φωτεινές" συνθήκες. Κατά τη διάρκεια αυτής της επεξεργασίας, προστέθηκε ένα αρχείο ήχου συμβατό με το πρόσωπο που αναπαρίσταται κάθε στιγμή. Αυτή η διαδικασία αποτελεί την τελική διαμόρφωση του οπτικοακουστικού συνόλου δεδομένων κατάλληλου για μεθόδους αναγνώρισης εκφράσεων.

Τέλος, για να αξιολογηθεί το σύνολο δεδομένων που δημιουργήθηκε, εκπαιδεύτηκε σε αυτό μια μέθοδο αναγνώρισης συναισθημάτων με οπτικοακουστικά δεδομένα σε 4 πειραματικές εκδοχές. Στόχος ήταν να αξιολογηθεί η επίδραση της χρήσης συνθετικών εικόνων από λήψεις που δεν είναι διαθέσιμες στο πραγματικό σύνολο δεδομένων κατά την εκπαίδευση αλλά και να εξαχθούν και επιπρόσθετα συμπεράσματα.

ABSTRACT

This thesis aims to create a comprehensive multimodal (audiovisual) facial expressions dataset that includes animated 3D models along with images and videos captured from various viewpoints. A subsequent stage focuses on applying methods developed for expression recognition in the derived dataset in order to study the effect of different parameters such as lighting conditions, camera position and distance on the accuracy and performance of the expression recognition algorithm.

Initially, a search was conducted for a dataset depicting individuals with different facial expressions / emotions through videos. Among those examined, the final choice was the RAVDESS dataset since it includes both visual and audio data.

Following the conversion of RAVDESS data into a format suitable for the next phase, the development was divided into three basic stages. The first of these involved creating a dataset of a series of three-dimensional models, reconstructed from video frames using two different approaches, in order to increase the number of samples: either by considering a face and the corresponding emotions as presented in the dataset or by combining the facial geometry and texture of one individual with the emotions of another individual. This stage was implemented using the DECA software module.

The 3D models were integrated into the Webots simulator to produce animations depicting the faces representing the different emotions. During this process, code was implemented that captures video frames of the human head. The frames were collected from a grid of virtual cameras, positioned at a total of 25 angles (-60°...+ 60° in a pan with 30° steps and -30°... + 30° in tilt with increments of 15°) and 3 distances (0.5, 0.75 and 1 m). In addition, in order to simulate different lighting conditions, the video images were collected in "dark" and "bright" lighting. During this processing, an audio file compatible with the person represented at each moment is added. This process constitutes the final configuration of the audiovisual dataset suitable for expression recognition methods.

Finally, in order to evaluate the created dataset, it was trained in a method for audiovisual emotion recognition in 4 experimental versions. The aim is to evaluate the effect of using synthetic images from shots not available in the real-world dataset during training.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

1. Αναγνώριση εκφράσεων προσώπου, 2. Τρισδιάστατα μοντέλα 3. Webots

KEYWORDS

1. Facial expression recognition 2. 3D models 3. Webots

List of Figures

List of Tables

1. Introduction

1.1 Topic of the thesis

The thesis delves into the creation of an audiovisual facial expression dataset that depicts individuals posing facial expressions recorded from a grid of virtual cameras, positioned at a total of 25 angles and 3 distances, under 2 lighting conditions. The initial phase involved searching for a dataset encompassing facial images portraying diverse emotional states. Leveraging advanced software tools, facial characteristics such as geometric attributes, texture nuances, and fine-grained details like wrinkles were extracted, towards the generation of detailed 3D models depicting these facial expressions. Subsequent animation of these models using appropriate simulation software yielded a dataset comprising images and videos derived from these simulations. To evaluate the correctness and usefulness of the new dataset, experiments were conducted in which it was used for training and testing of an audiovisual emotion recognition method.

1.2 Purpose

This thesis focuses on the construction of an audiovisual dataset, which can be used to train and test expression recognition methods. The dataset includes videos with appropriate audio, captured from different shooting angles, under two lighting conditions and three distances. The dataset characteristics make it suitable for active facial expression recognition, where camera movement can improve the accuracy of recognition. Understanding and interpreting facial expressions is crucial in several areas, such as detecting emotions in online environments, enhancing security measures through expression recognition, and improving human-computer interfaces. This scientific effort aims to improve these applications by providing a new synthetic dataset.

A significant aspect of this method involves using three-dimensional facial models. These models aim to accurately capture facial nuances, such as geometric and textural details, essential for portraying individuals and their emotional expressions. Addressing these complex aspects presents challenges that underscore the importance of this research.

The objectives of this study are twofold: firstly, to create an audiovisual dataset suitable for training/testing expression recognition methods and secondly, to evaluate the effect of various viewing angles and distances on the performance of such methods.

1.3 Key points

The methodology devised for the expression recognition dataset creation unfolds across three key stages:

1. Data Selection for 3D Model Construction: The process begins with the acquisition of facial data essential for building the requested 3D models. The primary objective is to procure facial images or videos that encapsulate a diverse spectrum of emotions, ensuring that each facial detail is carefully captured. Consequently, images or videos focusing solely on the face, devoid of any distracting body elements, are sought after. An ideal dataset would also include audio content, enriching the final simulation with auditory cues.After careful consideration of these criteria, which will also be analysed in the next chapter, the RAVDESS dataset [1] emerged as the optimal choice.
2. 3D Model Construction: Following the processing and conversion of the dataset into the desired image format, the subsequent phase involves the construction of the 3D facial models from images. Through a comprehensive study, the DECA [2] model emerged as the frontrunner for this task, due to its accuracy in 3D face reconstruction and accurate modelling of dynamic changes in facial features, including wrinkles induced by expressions. Through the manipulation of specific code fragments provided by the model, the 3D models were generated, ensuring precision and fidelity in representation.
3. Visualisation and Realistic Simulation: The final stage involves visualising the constructed models and importing them into a simulation environment. The objective is to animate the 3D models, with the addition of corresponding sound. The Webots simulator [3] was used for this task, configured and supplemented with the appropriate code. The generated videos feature simultaneous audio playback and multiple camera shots from 25 angles, positioned at 3 different distances and under 2 different lighting conditions. As a result, each sequence of 3D models derived from a RAVDESS AV file is represented in animation format, enriched with corresponding audio files.

2. Methodology

As previously outlined, this thesis focuses on the creation of 3D facial models, which, through processing, lead in the generation of a new dataset of images and videos, suitable for expression recognition. The objective of this endeavour is to leverage the research outcomes for further experimentation. The software tools and methodologies employed, along with their overarching objective, squarely align with the domain of computer vision. This association stems from the utilisation of techniques for generating 3D models, while the applications of facial recognition are inherently categorised within the purview of computer vision.

2.1 Computer vision

Computer vision [5], a field within machine learning, concentrates on the analysis and interpretation of images and videos to enable computers to perceive and utilise visual information for tasks traditionally performed by humans. Through training, computer vision models identify features and contextual elements in visual data, facilitating their ability to analyse images and videos and make informed predictions or decisions. Although related, computer vision is distinct from image processing. Image processing involves the modification or enhancement of images to achieve specific results, such as adjusting brightness, contrast, and resolution, or blurring sensitive information and cropping. Unlike computer vision, image processing does not inherently require the recognition or understanding of image content.

The domain of computer vision is dedicated to creating artificial systems that can analyse and interpret visual information. This visual information includes a wide range of formats such as video sequences, inputs from multiple cameras, and data from 3D scanners or medical imaging devices. In this framework, the goal of the present study is to produce 3D models from videos to precisely depict facial expressions.

For a better understanding and deeper insight into usability of computer vision [6], some apps and examples (Figure 1) where it is used are the following:

- Self-Driving Cars: Computer vision enables autonomous vehicles to process images from multiple cameras in real-time, allowing them to detect road edges, interpret signage, identify and recognize pedestrians. This capability allows safe navigation and passenger transport.
- Augmented and Mixed Reality: Computer vision helps augmented reality systems in smartphones and wearable devices to recognize surfaces like tabletops, ceilings, and floors, accurately overlaying digital content onto the real world.
- Object Detection: In manufacturing, computer vision is used to detect and catalogue defects or locate broken equipment and other objects by classifying and analysing images.
- Categorization of Images: Computer vision is used to classify and analyse images for various purposes such as sorting products, ensuring quality control, and managing inventory.
- Retrieval of Images Based on Their Content: Computer vision retrieves images based on actual content, enabling efficient search and exploration in large data sets and facilitating automatic image annotations.
- Facial Expression Recognition: Computer vision analyses facial features to identify and categorise emotional expressions, automating emotion detection in real-time.

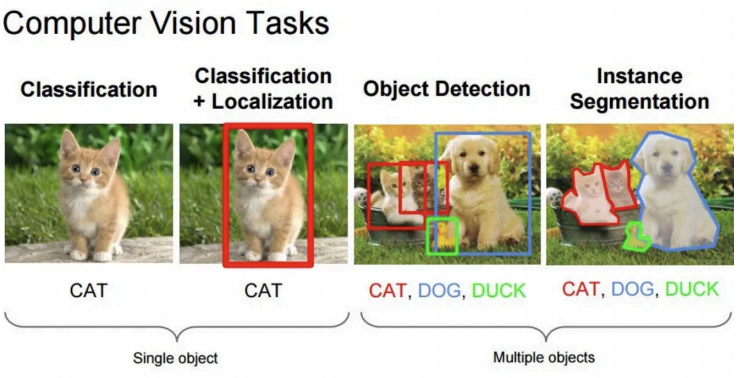


Figure 1: Examples of Computer Vision Tasks. Source: Everything You Ever Wanted To Know About Computer Vision by Ilija Mihajlovic, https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e

2.2 Facial Expression Recognition methods and motivation of this study

Facial expression recognition is a technology that involves the automated detection and analysis of facial movements to infer human emotions or intentions. It relies on computer algorithms to identify key facial features such as eyebrow position, eye movement, mouth shape, and overall facial muscle contractions. These features are then analysed to classify expressions like happiness, sadness, anger, surprise, or disgust. Facial expression recognition systems typically use techniques from computer vision and machine learning, such as deep learning models trained on large datasets of facial images. As a recognition method, it involves some key steps to analyse and interpret emotions from facial images or video frames. These steps typically include face detection, feature extraction and expression recognition.

Facial expression recognition is widely applied [9] [10] across several fields within computer science and beyond:

- Healthcare: Facial expression recognition is utilised for various purposes such as monitoring patient emotions during clinical assessments, assisting in pain management, and aiding in psychological evaluations.
- Security and Surveillance: Facial expression recognition enhances security systems by enabling the detection of suspicious behaviours or emotional states in real-time. It is employed in surveillance applications for monitoring public spaces and in access control systems for authentication purposes.
- Education and Learning: Educational technologies leverage facial expression recognition to personalise learning experiences based on

students' engagement levels and emotional responses. This adaptation helps in creating more effective and tailored educational environments.

- Marketing and Consumer Behavior: In marketing research, facial expression recognition is used to analyse consumer reactions to advertisements, products, or services. By understanding emotional responses, marketers can optimise campaigns and strategies to better resonate with target audiences.

Humans [8] have a basic set of emotions, which are communicated via universal facial expressions. The ability to automatically recognize emotions in images and videos can be achieved by developing an algorithm that detects, extracts and evaluates these expressions in real time. In social settings, facial expressions are powerful means of communicating personal emotions and intentions, playing an important role in human social interaction, which makes the ability to recognize them important. Emotions can be expressed through a variety of means, including words, hand and body movements, and facial expressions. Therefore, the ability to extract and understand emotions is critical for effective communication between humans and machines.

Facial expressions [11] play an important role in recognition of emotions and are used in the process of non-verbal communication, as well as to identify people. They are very important in daily emotional communication, just next to the tone of voice. They are also an indicator of feelings, allowing a man to express an emotional state. People can immediately recognize an emotional state of a person. As a consequence, information on facial expressions are often used in automatic systems of emotion recognition. The human face, as the most exposed part of the body, allows the use of computer vision systems (usually cameras) to analyse the image of the face for recognizing emotions. Light conditions and changes of head position are the main factors that affect the quality of emotion recognition systems using cameras.

The integration of 3D models into facial expression recognition methods presents many advantages for increasing the accuracy and robustness of sentiment analysis systems. Unlike conventional 2D approaches, 3D models offer additional depth information, capturing the fine contours and nuances of facial features that are important for understanding complex emotional expressions. These depth data not only enhance the accuracy of emotion detection but also ensures independence from the point of view, mitigating the impact of variations in lighting conditions and shooting angles. Furthermore, the accurate tracking of facial landmarks in three dimensions facilitated by 3D models allows for differentiated interpretation of emotions, leading to more accurate recognition results. The ability to render 3D models with high fidelity contributes to the creation of realistic representations of facial expressions, thus enhancing the interpretability and generalisability of emotion recognition algorithms. The flexibility of 3D models allows for data augmentation, enabling the creation of synthetic data to enrich the training process and improve the performance of the models in different scenarios. In summary, the incorporation of 3D models in facial expression recognition methodologies not

only increases the accuracy and reliability of facial analysis, but also contributes to the creation of more effective applications related to human-computer interaction, virtual reality and emotional computing domains.

2.3 The problem

The methodology employed in this thesis for creating the audiovisual dataset is based on the conversion of 2D facial images into sequences of 3D models, aimed at recognizing the expressions within them. The primary goal is to utilise existing 2D image datasets as source material, extracting rich facial data to construct corresponding three-dimensional models. This transformation from 2D to 3D representation not only enriches the dataset with depth information but also facilitates a more nuanced understanding of facial expressions, capturing details often overlooked in flat images. By leveraging advanced computer vision techniques, each 2D image is converted into its corresponding three-dimensional form, thus enriching the dataset with essential depth cues necessary for robust emotion recognition. The central challenge lies in identifying and utilising appropriate datasets and software tools required for the effective execution of this data transformation. The final objective is to create a dataset that includes audiovisual data captured from various camera positions and angles, making it suitable for use in active expression recognition methods.

As the 3D model sequences are generated, we had to identify a suitable simulator to complement the process of generating the targeted dataset. In this step, the aim is to simulate the behaviour of the constructed 3D models in dynamic environments, thus facilitating the creation of the datasets. This process seeks to capture the 3D models under a variety of conditions, which include different lighting conditions and shots from a range of camera angles and distances. The use of a suitable simulator will allow for the reproduction of realistic conditions under which the 3D models will be recorded under these different scenarios.

In conclusion, the present challenge can be delineated into three main components: selecting suitable datasets with facial expressions, using software tools that support the construction of 3D face representations derived from the input data, and selecting a simulation framework for creating a new dataset suitable for use in facial expression recognition methods.

2.4 The approaches

The strategy employed to accomplish the goals entailed using new facial models within a realistic scene setting. The primary challenge addressed involved accurately capturing facial features and expressions. This required detailed rendering of facial characteristics during simulation, dynamically adjusting to ensure precise representation. This process was facilitated through advanced techniques, simulating natural facial movements to deliver comprehensive portrayals similar to video sequences. Factors such as lighting, shadowing, and

perspective angles significantly influenced the outcomes. Thus, capturing a diverse array of images with varying settings was important to create a new dataset.

In the next chapter, a detailed exposition will be provided regarding the utilisation of the DECA model [2], which enables the integration of facial texture into models during their developmental phase. This feature prompted the exploration of two distinct methodologies for crafting 3D facial representations. In both instances, adjustments were made to the RAVDESS dataset to serve as a resource for capturing emotions in image form. This transformative process yielded two datasets: one comprising image files and another featuring singular portraits of each face actor. The use of the DECA model led to the creation of two new datasets: the first involved constructing models depicting actors posing their own facial expressions, while the second involved creating models employing the facial geometry of an actor and the expression of another. Subsequent to model construction, the process of model animation and capturing of the image and audio data was executed utilising Webots, resulting in the generation of two supplementary datasets showcasing images of the modelled outcomes. The produced data were in video format, accompanied by corresponding audio files. To evaluate the effectiveness of the new dataset, it was used to test a method for audiovisual emotion recognition.

3. RAVDESS

3.1 Description

In order to create the dataset for facial expression recognition methods, a search of the scientific literature was conducted. The overarching goal was to identify a dataset that met specific criteria. First, a search was conducted for a dataset involving multiple individuals reflecting diverse demographics, each exhibiting a spectrum of facial features and expressions. It was necessary that this dataset captured individuals portraying a rich array of facial expressions. The ideal representation of these emotions encompassed their dynamic / temporal evolution. This portrayal could be exemplified through videos showcasing the onset of emotions, such as a person transitioning from a neutral facial expression to anger, with each successive frame precisely capturing the shift in emotional state. In accordance with the criteria, the necessary data were procured from the RAVDESS dataset (The Ryerson Audio-Visual Database of Emotional Speech and Song) [1], as shown in Figure 2.

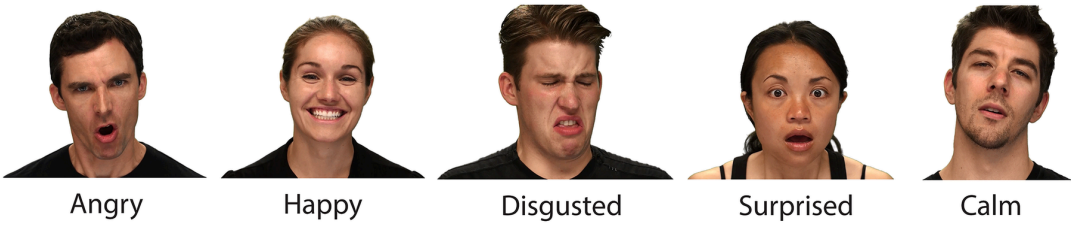


Figure 2: Cover photo of RAVDESS data: https://zenodo.org/communities/ravdess/records?q=&l=list&p=1&s=10&sort=newest

The RAVDESS dataset comprises a total of 7356 files in three formats: full-AV, video-only, and audio-only. These files feature performances by 24 professional actors, evenly split between genders (12 women and 12 men), delivering two lexically equivalent statements (01 = "Kids are talking by the door", 02 = "Dogs are sitting by the door") in a neutral North American accent. Each statement is presented at two intensity levels: normal and intense, alongside an additional neutral expression. Emotions are conveyed through two different ways: through speaking and singing. Speaking data encompass 8 emotions: neutral, calm, happiness, sadness, anger, fear, disgust, and surprise, while singing data include 5 emotions: calm, happiness, sadness, anger, and fear. The actors are filmed against a neutral white background, facing the camera directly to express their emotions. The images are in high-resolution colour (1280 x 720 pixels).

3.2 Modification of data

In order to achieve the synchronised playback of sound and image during the reconstruction of the models, the use of full-AV files was chosen. However, due to the large volume of files, the focus was only on speech and not on singing. All other features of the videos were kept unchanged. Therefore, the number of videos that were used is equal to

24 actors x 8 emotions x 2 emotional intensities x 2 statements x 2 repetitions = 1536 videos

After selecting these files from the dataset, the subsequent step is to structure them systematically. Following the organisational framework established in the RAVDESS dataset, each actor is allocated to an individual folder identified by a unique numeric label. These folders are sequentially numbered from 01 to 24, corresponding to the actors represented in the dataset. The folders with odd numbers feature videos portraying male actors, while those with even numbers showcase female actors. Within each actor's designated folder, a file is assigned for every video featuring that actor.

As outlined in the dataset selection criteria mentioned above, to facilitate comprehensive expression analysis, it is essential to capture emotions frame by frame. Consequently, a code in Python was developed to convert each available

video into a sequence of images. To ensure data integrity, all generated images were preserved in the order of their creation.

In accordance with the RAVDESS data naming convention, a specific folder was established to accommodate a series of files, each housing images extracted from processed videos. Each file name consists of 8 components connected by the lower-colon symbol. Beginning with the data modality, always denoted as '01' representing full-AV, followed by the vocal channel, where '01' signifies speech. Subsequent components include the emotion captured, ranging from '01' for neutral to '08' for surprise, and the level of emotional intensity, with '01' indicating normal and '02' indicating strong. The statement and repetition are then identified, with '01' representing the phrase 'Kids are talking by the door' and '02' for 'Dogs are sitting by the door,' and '01' and '02' representing the first and second repetitions, respectively. The actor's number, ranging from 01 to 24, is also included. Moreover, to emphasise the encountered emotion in each instance, it is reiterated at the end of the filename with a descriptive word. An example of this nomenclature is shown in Table 1 for the file 01_01_02_01_01_01_09_calm. Finally, the content of the image folder adheres to this naming convention, with each filename concluding with a three-digit frame number.

01'	’01'	’02'	’01'	’01'	’01'	’09'	calm
Modality: full-AV	Vocal channel : speech	Emotion : calm	Emotional intensity: Normal	Statement: ”Kids are talking by the door"	Repetition: 1st	Actor: 09	Name of emotion

Table 1: Description of the file 01_01_02_01_01_01_09_calm

Regarding the final data volume for the creation of 3D models, a total of 1440 videos, equivalent to 1440 image files, were considered. The difference from the previously calculated number (1536) arises from the utilisation of only one modality, full-AV. As per the RAVDESS file descriptions, speech files containing solely one modality and neutral emotion one emotional intensity, are calculated as such

24 actors x (7 emotions x 2 emotional intensities x 2 statements x 2 repetitions) + (neutral emotion x 2 statements x 2 repetitions) = 1440 folders.

Considering the number of frames in each video, all videos have a duration of 3-4 seconds with a consistent frame rate of 30 frames per second (fps). Therefore, the expected range of frames per file is from 3 seconds × 30 fps = 90 frames to 4 seconds × 30 fps = 120 frames.

3.3 Structure

Having structured the dataset in this specified format, the next step involved organising the data into files with a structure to facilitate subsequent procedures. This entailed the creation of two distinct folders: 'exp' and 'shape' (Figure 3). The 'exp' folder encompasses a total of 1,440 files, each containing frames extracted from the videos and converted into individual images, as previously outlined. In contrast, the 'shape' folder contains a collection of single frames, each capturing an actor with a neutral facial expression. Each of these frames is uniquely identified by a two-digit serial number, allowing for precise reference and organisation.

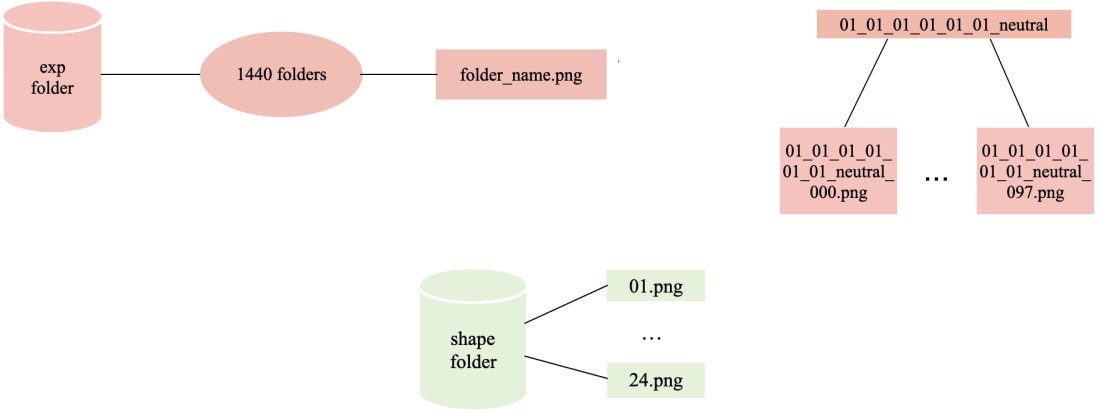


Figure 3: The layouts illustrate the final structure of the RAVDESS data folders and one directory example of exp folder

4. DECA

4.1 Introduction

After gathering the dataset into the desired format, the next step involves searching for a tool capable of creating 3D models from this dataset. A primary criterion is the ability to preserve detailed facial features without distorting the emotional expressions they convey. Therefore, a new research was conducted to examine available software tools capable of capturing these detailed expressions while also providing detailed facial geometry. Another goal was to seamlessly incorporate speech into each successive series of images.

Presently, monocular methods employed for the reconstruction of three-dimensional (3D) facial structures demonstrate proficiency in capturing intricate geometric nuances, yet they are fraught with several inherent limitations. Certain methodologies yield facial models that, despite their precision in geometric

fidelity, exhibit deficiencies in realism within animated contexts. This deficiency stems from their failure to adequately account for the dynamic alterations in facial features, particularly how wrinkles manifest with varying expressions. Conversely, alternative techniques, although trained on datasets of high-quality facial scans, grapple with the challenge of effectively extrapolating to real-world image datasets. After careful consideration we ended up selecting DECA.

4.2 Description of DECA

Current monocular 3D face reconstruction techniques can capture fine geometric details, but they have several drawbacks. Some methods produce faces that lack realistic animation capability because they do not account for the variation in wrinkles with different expressions. Others are trained on high-quality face scans and fail to generalise to in-the-wild images. The DECA (Detailed Expression Capture and Animation) model [2] addresses these issues by estimating 3D facial shapes and animatable details that are unique to an individual and vary with expressions. It generates a UV displacement map from a low-dimensional latent representation that includes both individual-specific detail parameters and general expression parameters. A regressor is trained to predict detail, shape, albedo, expression, pose, and illumination parameters from a single image. The model introduces a novel detail-consistency loss, which separates individual-specific details from expression-dependent wrinkles, allowing realistic synthesis of individual-specific wrinkles by adjusting expression parameters while keeping individual-specific details unchanged. DECA is trained using in-the-wild images without paired 3D supervision and achieves state-of-the-art shape reconstruction accuracy on two benchmarks. Qualitative results on in-the-wild data show DECA’s robustness and its ability to distinguish between identity- and expression-dependent details, enabling the animation of reconstructed faces. The model and code are publicly available.

The main idea of DECA is to learn to estimate a parameterized face model with geometric detail exclusively from in-the-wild training images. After training, DECA can reconstruct the 3D head with detailed facial geometry from a single image. The parameterization of the reconstructed details allows for the animation of the detailed reconstruction by adjusting FLAME's expression and jaw pose parameters, creating new wrinkles while maintaining individual-specific details unchanged.

4.3 Description of FLAME model

The FLAME model (Faces Learned with an Articulated Model and Expressions) [16] is designed to integrate seamlessly with existing graphics software and to be easily adaptable to data. FLAME employs a linear shape space trained from 3800 human head scans. This linear shape space is combined with an articulated jaw, neck, and eyeballs, pose-dependent corrective blend-shapes, and additional global

expression blend-shapes. The model's pose and expression-dependent movements are learned from 4D face sequences in the D3DFACS dataset, along with other 4D sequences. By accurately registering a template mesh to the scan sequences, D3DFACS registrations are provided for research purposes. Overall, the model is trained on over 33,000 scans. FLAME is low-dimensional yet more expressive than the FaceWarehouse model and the Basel Face Model. When compared to these models by fitting them to static 3D scans and 4D sequences using the same optimization method, FLAME demonstrates significantly greater accuracy and is available for research purposes.

In the methods that DECA employes, FLAME (Figure 4) serves as a statistical 3D head model that integrates distinct linear identity shape and expression spaces with linear blend skinning (LBS) and pose-dependent corrective blend-shapes, enabling the articulation of the neck, jaw, and eyeballs.Given parameters of facial identity $\boldsymbol{\beta} \in R^k$, pose $\theta \in R^{3k+3}$, (with k = 4 joints for neck, jaw and eyeballs), and expression $\boldsymbol{\psi} \in R^{\psi}$, FLAME outputs a mesh with n = 5023 vertices. The model is defined as

$$M(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}) = W(T_P(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}), \mathrm{J}(\boldsymbol{\beta}), \boldsymbol{\theta}, \mathrm{W}),$$

with the blend skinning function W (T, J, $\boldsymbol{\theta}$, W) that rotates the vertices in $\mathrm{T} \in R^{3n}$ around joints $\mathrm{J} \in R^{3k}$, linearly smoothed by blend weights $\mathrm{W} \in R^{kxn}$. The joint locations J are defined as a function of the identity $\boldsymbol{\beta}$.
Further,

$$T\mathrm{p}(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}) = \mathrm{T} + B\mathrm{s}(\boldsymbol{\beta};\mathrm{S}) + B\mathrm{p}(\boldsymbol{\theta};\mathrm{P}) + B_E(\boldsymbol{\psi};\mathrm{E})$$

denotes the mean template T in "zero pose" with added shape blendshapes $B\mathrm{s}(\boldsymbol{\beta};\mathrm{S}) : R^k \mathrm{x}\rightarrow R^{3n}$,pose correctives $\mathrm{Bp}(\boldsymbol{\theta};\mathrm{P}) : R^{3k+3} \rightarrow R^{3n}$, and expression blend shapes $B_E(\boldsymbol{\psi};\mathrm{E}) : R^{|\psi|} \rightarrow R^{3n}$, with the learned identity, pose, and expression bases (i.e. linear subspaces) S, P and E.

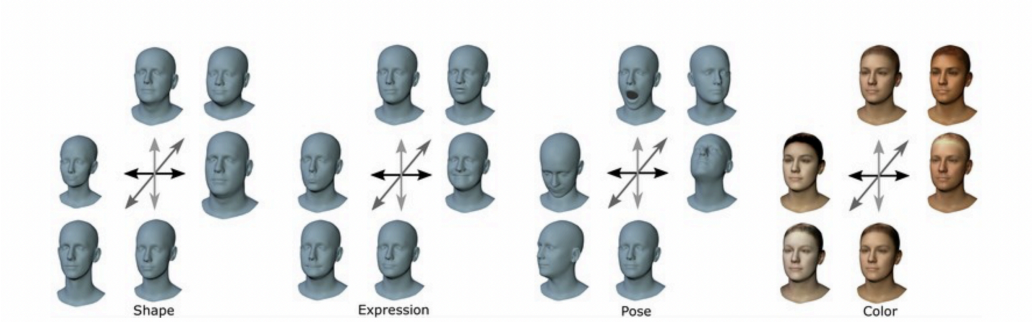


Figure 4: Different versions of the FLAME model illustrate variations in shape, expression, pose, and appearance. For shape, expression, and appearance, the first three principal components are shown at ±3 standard deviations. Pose variations are depicted at ±π/6 for head movements and at 0, π/8 for jaw movements. Source: https://flame.is.tue.mpg.de/

4.4. DECA method

The DECA approach focuses on two main aspects: coarse reconstruction and detailed reconstruction. Initially, a 2D image is processed, encoded into a latent code, and then decoded to generate a synthesised 2D image , with the goal of minimising the difference between the synthesised image and the original input. Using an expression image (e) to extract facial expression parameters and one or more shape images (s) to obtain the shape parameters, including texture, the program generates multiple three-dimensional face models in various (e, s) combinations. These models display the shape and texture of the shape image (s) while capturing the facial expression from the expression image (e). Following this, the detailed reconstruction phase improves the coarse FLAME geometry by adding a detailed UV displacement map DD, with values ranging from -0.01 to 0.01.

4.4.1 Coarse reconstruction

For the coarse reconstruction (Figure 5), an encoder Ec is trained, consisting of a ResNet50 [He et al. 2016] network followed by a fully connected layer, to generate a low-dimensional latent code. This code comprises FLAME parameters β, ψ, θ (i.e. representing the coarse geometry), albedo coefficients a, camera c, and lighting parameters l. More specifically, the coarse geometry uses the first 100 FLAME shape parameters (β), 50 expression parameters (ψ), and 50 albedo parameters (a). In total, E predicts a 236 dimensional *c* latent code. Given a dataset of 2*D* face images Ii with multiple images per subject, corresponding identity labels *ci* , and 68 2*D* keypoints k*i* per image, the coarse reconstruction branch is trained by minimising

$$L_{coarse} = L_{lmk} + L_{eye} + L_{pho} + L_{id} + L_{sc} + L_{reg},$$

with landmark loss L_{lmk} , eye closure loss L_{eye} , photometric loss L_{pho} , identity loss L_{id} , shape consistency loss L_{sc} and regularisation L_{reg}.

Important metrics used in this reconstruction include:

1. Landmark re-projection loss: quantifies the discrepancy between the actual 2D facial landmarks k*i* and the corresponding landmarks on the surface of the FLAME model $M_i \in R^3$, which are projected into the image using the estimated camera model. The landmark loss is defined as

$$L_{lmk} = \sum_{i=1}^{68} \|\mathbf{k}_i - s\Pi(M_i) + \mathbf{t}\|_1 .$$

2. Eye closure loss: calculates the relative offset between landmarks k*i* and k*j* on the upper and lower eyelids, comparing it to the offset of the corresponding landmarks on FLAME's surface *Mi* and *Mj* when projected into the image. Given E as the set of upper/lower eyelid landmark pairs, the loss is defined as follows:

$$L_{eye} = \sum_{(i,j)\in E} \left\|\mathbf{k}_i - \mathbf{k}_j - s\Pi(M_i - M_j)\right\|_1$$

3. Photometric loss: calculates the discrepancy between the input image *I* and the rendering Ir as

$$\text{LphoD} = \|\text{VI} \odot (\text{I} - \text{Ir}')\|_{1,1}$$

In this context, *V*I represents a facial mask where the value is 1 within the facial skin area and 0 outside, determined through an established face segmentation technique [Nirkin et al., 2018], and ⊙ denotes the element-wise product (Hadamard product).

4. Identity loss: The face recognition network *f* generates feature embeddings for both the rendered images and the input image. The identity loss quantifies the cosine similarity between these two sets of embeddings. Formally, the loss is defined as

$$L_{id} = 1 - \frac{f(I)f(I_r)}{\|f(I)\|_2 \cdot \|f(I_r)\|_2}.$$

By evaluating the difference between the embeddings, the loss ensures that the rendered image retains key aspects of the individual's identity, making certain that the rendered image resembles the same person as the input image.

5. Shape consistency loss: The aim is to ensure that the rendered images accurately represent the real person. If the method has correctly estimated the facial shape in two images of the same individual, then exchanging the shape parameters between these images should result in rendered images that are indistinguishable. To achieve this, the photometric and identity loss are applied to the rendered images with swapped shape parameters, minimising the following

$$L_{sc} = L_{coarse}(I_i, \mathcal{R}(M(\boldsymbol{\beta}_j, \boldsymbol{\theta}_i, \boldsymbol{\psi}_i), B(\boldsymbol{\alpha}_i, \mathbf{l}_i, N_{uv,i}), \mathbf{c}_i))$$

6. Regularisation: Lreg regularises shape $E\beta = \| \beta \|_2$, expression $E\psi = \| \psi \|_2$, and albedo $E\alpha = \| \boldsymbol{\alpha} \|_2$.

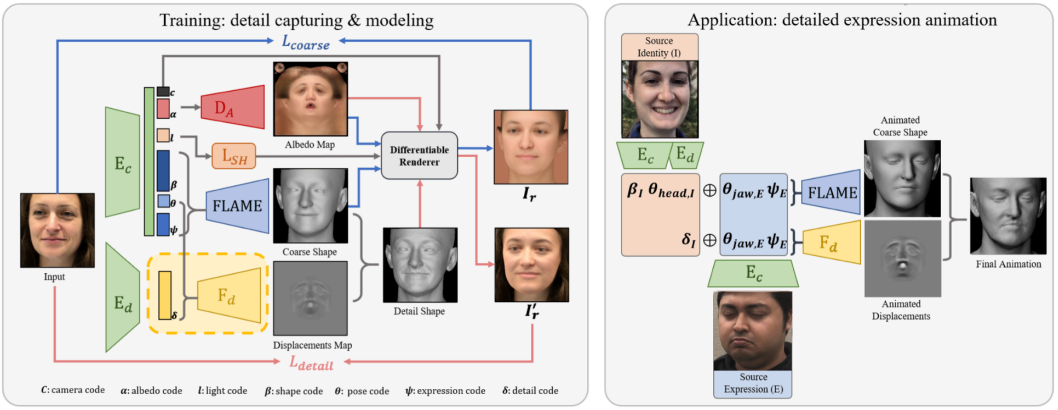


Figure 5: DECA Training and Animation Process. In the training stage (left box), DECA determines parameters to reconstruct facial shapes for each image, utilising shape consistency data (blue arrows). It also develops an expression-based displacement model by using detailed consistency data (red arrows) from multiple images of the same person. The novelty of DECA is highlighted in the yellow box area. After training, DECA animates a face (right box) by merging the reconstructed shape, head pose, and detail code of the source identity with the jaw pose and expression parameters of the reconstructed source expression, resulting in an animated coarse shape and displacement map. Finally, DECA generates an animated detailed shape. Source: https://files.is.tue.mpg.de/black/papers/SIGGRAPH21_DECA.pdf

4.4.2. Detail Reconstruction

The detail reconstruction enhances the basic FLAME geometry by incorporating a detailed UV displacement map $D \in [-0.01,0.01]^{dxd}$. Similar to the coarse reconstruction, an encoder E_d (with the same architecture as E_c) is trained to encode the input image I into a 128-dimensional latent code δ, which captures subject-specific details. The latent code δ is then concatenated with FLAME's expression ψ and jaw pose parameters θ_{jaw} and decoded by Fd to D.

Some key points of this reconstruction are about the decoder, the rendering and some losses:

1. Detail decoder: The detail decoder is described by the equation $D = F_d(\delta, \psi, \theta_{jaw})$, where the detail code $\delta \in R^{128}$ controls the static person-specific details. The expression $\boldsymbol{\psi} \in R^{50}$ and jaw pose parameters $\boldsymbol{\theta}_{jaw} \in R^{3}$ from the coarse reconstruction phase are used to capture the dynamic expression wrinkle details. For rendering purposes, D is converted into a normal map.
2. Detail rendering: The detail displacement model enables the generation of images with mid-frequency surface details. To reconstruct the detailed geometry M', we convert M and its surface normals N to UV space, denoted as $M_{uv} \in R^{3xdxd}$ and $N_{uv} \in R^{3xdxd}$, and combine them with D as $M'_{uv} = M_{uv} + D \odot N_{uv}$. By calculating normals N' from M', we obtain the detail rendering I_r' by rendering M with the applied normal map as $I_r' = R(M, B(\boldsymbol{\alpha}, l, N'), c)$. The detail reconstruction process is optimised by minimising the loss $L_{detail} = L_{phoD} + L_{mrf} + L_{sym} + L_{dc} + L_{regD}$, which includes the photometric detail loss L_{phoD}, ID-MRF loss L_{mrf}, soft symmetry loss L_{sym}, and detail regularisation L_{regD}. Since the estimated albedo is generated using a linear model with 50 basis vectors, the rendered coarse face image captures only low-frequency information such as skin tone and basic facial features. High-frequency details in the rendered image mainly stem from the displacement map. Therefore, by comparing the rendered detailed image with the real image, F_d is compelled to model detailed geometric information.
3. Detail photometric losses: By applying the detail displacement map, the rendered images Ir′ exhibit some geometric details. Similar to the coarse rendering, a photometric loss is employed, where V_I is a mask indicating the visible skin pixels. The losses are of the type:
$$L_{phoD} = ||V_I \odot (I - I_r')||_{1,1}$$
4. ID-MRF loss: The ID-MRF loss functions by extracting feature patches from various layers of a pre-trained network, using both the input image and the detail rendering. It then minimises the discrepancy between corresponding nearest neighbour feature patches from these images. This loss regularises the generated content to match the original input at the local patch level, enabling DECA to capture high-frequency details. The loss is computed on the layers *conv*3_2 and *conv*4_2 with the type
$$L_{mrf} = 2L_M(conv4\_2) + L_M(conv3\_2)$$

where L_M (*layer*h) denotes the ID-MRF loss that is employed on the feature patches extracted from Ir′ and I. Similar to the photometric losses, LmrfLmrf is calculated only for the face skin region in UV space.

5. Soft symmetry loss: To regularise the non-visible parts of the face, a soft symmetry loss is incorporated. This loss aims to minimise discrepancies using the following method, where Vuv represents the face skin mask in UV space and "flip" denotes the horizontal flip operation. Without L_{sym}, extreme poses can result in visible boundary artefacts in occluded areas. The loss is computed with the type
$$L = \| V \odot (D - \text{flip}(D)) \|$$
6. Detail regularisation: The detail displacements are regularised by $L_{regD} = \| D \|_{1,1}$ to reduce noise.

4.5 Detail disentanglement

Another pivotal aspect of the DECA model methodology involves its final stage: the detailed decomposition process. By optimising Ldetail, the model aims to reconstruct faces with detailed mid-frequency features. However, to enable these detailed reconstructions to be animated effectively, it's essential to differentiate between person-specific details, such as moles, pores, eyebrows, and wrinkles that remain consistent regardless of facial expression, controlled by δ, and expression-dependent wrinkles, those that vary with different facial expressions, controlled by FLAME's expression and jaw pose parameters ρ and θjaw. A critical observation is that across two images of the same individual, there should be consistency in both the overall facial structure and the personalised details.

In particular, when comparing the rendered detailed images, exchanging the detail codes between two images of the same individual should not change the rendered outcome. This concept is illustrated in Figure 6. Here, the jaw and expression parameters are taken from image *i*, the detail code is derived from image *j*, and these are combined to compute the wrinkle detail. Ensuring realism means that when detail codes are swapped between different images of the same person, the resulting outputs should remain consistent.

The detail consistency loss, as shown in Figure 6, plays an important role in separating identity-specific details from expression-dependent details. Without this loss, the detail code δ would encompass both identity and expression-related details. Consequently, reconstructed details would not accurately reflect changes in FLAME jaw pose and expression variations. Given two images Ii and Ij of the same subject (i.e. $c_i = c_j$), the loss is defined as

$$L_{dc} = L_{detail}(I_i, R(M(\beta_i, \theta_i, \psi_i), A(\boldsymbol{\alpha} i), Fd(\boldsymbol{\delta} j, \boldsymbol{\psi} i, \boldsymbol{\theta}_{jaw}, i), l i, c i)),$$

where βi, θi, ψi, θjaw,i , ai, li and ci are the parameters of Ii, while δj is the detailed code of Ij.

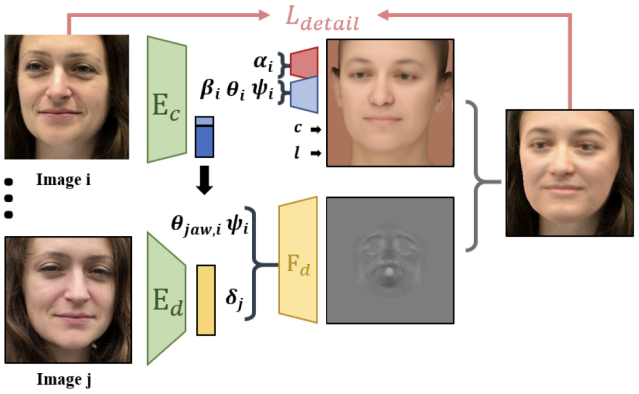


Figure 6: Detail consistency loss. DECA employs multiple images of the same individual during training to separate static, person-specific details from those that vary with expressions.Source:https://files.is.tue.mpg.de/black/papers/SIGGRAPH21_DECA.pdf

4.5 Limitations of DECA

DECA achieves very good results in reconstructing facial shapes and introduces novel animatable details but faces certain limitations. Firstly, its ability to render detailed meshes is restricted by the albedo model derived from BFM, necessitating an albedo space devoid of existing shading to effectively separate facial coloration from geometric intricacies. Secondly, DECA does not explicitly model facial hair, attributing skin tone to the lighting model, which can lead to facial hair being misrepresented due to shape deformations. While generally robust, DECA may struggle with extreme head poses and challenging lighting conditions, although it can handle typical occlusions found in standard datasets. However, it faces difficulties with severe occlusions, underscoring the need for more diverse training data to improve its robustness.

Furthermore, the training dataset contains a significant proportion of low-resolution images, which enhances robustness but may introduce noisy details. Conversely, existing high-resolution datasets offer less diversity. Training DECA on these datasets results in a model less adept at handling typical in-the-wild images but capable of capturing finer details. Additionally, the limited size of high-resolution datasets poses challenges in disentangling details dependent on expression and identity.

To further advance research in this field, a model trained exclusively on high-resolution images (DECA-HR) has been introduced. DECA-HR improves

visual fidelity and reduces noise in reconstructed details. However, this enhancement compromises robustness, particularly in handling low-resolution images, extreme head poses, extreme expressions, and similar conditions.

DECA currently operates with a weak perspective camera model. To extend its application to "selfies" and recover head geometry, integrating focal length into the method is essential. While focal length can sometimes be directly acquired from the camera for specific uses, inferring both 3D geometry and focal length from a single image under perspective projection, especially for in-the-wild images, remains a challenging task.

5. 3D models

5.1 DECA code

The next step involves usingDECA at the code level to build 3D models. To conduct experiments with the DECA model [2], the open-source software was configured on a Linux machine. Developed in Python and leveraging the PyTorch library [15], the software is designed to capitalise on the computational power of machines equipped with CUDA [16] and high-performance graphics processors. For our work, CUDA 11.0 and an Nvidia RTX 2080 were utilised. Additionally, accompanying the base code are two demo scripts. Some changes were made to the file to adapt it to the data of the RAVDESS dataset in which the experiments are performed.

The core code includes two demo scripts designed for various use cases outlined in Chapter 4, where DECA is introduced. To achieve the research objectives of the project, modifications were necessary to the demo code. The authors' initial approach for this specific demo code is as follows: Given an expression image (e), from which the program derives facial expression parameters, and one or more shape images (s), from which the program derives the shape parameters of the face, including texture, multiple combinations (e, s) of three-dimensional face models are generated. These models exhibit the shape and texture of image (s) while expressing the facial expression depicted in image (e). In practice, models are generated based on specific facial images, with an expression derived from another facial image.

However, this methodology does not support animation creation. Therefore, the code underwent modifications to enable the pairing of multiple expression images (e) with a facial shape image (s). This pairing facilitates the generation of animations featuring the facial shape and texture (s) alongside a sequence of expression images (e). This code is executed two times, with a different expression application mode in each case: the first was performed with each actor and themselves, without emotion transfer, while the second was conducted with emotion transfer from other individuals.

5.2 Methods and their results

In the initial phase of models creation, each actor's facial features are applied to every set of frames in which they appear. For instance, using an image from the 'shape' folder depicting an actor with a neutral expression identified as ‘09’, the process involves applying this facial representation to all corresponding image files portraying the actor with various emotional states (e.g., calm, angry), rather than utilising a different actor's image (e.g., ‘10’). This is achieved by initialising the DECA model with predefined configuration settings. Subsequently, the images structured as described above are loaded and processed using DECA. Notably, a control mechanism is implemented to accommodate the 1-1 actor combination scenario outlined in this thesis. The transfer of facial expression information between images is executed by adjusting the pose and expression parameters.

The final results yield a folder for each combination, containing a model (.obj) for every frame from the 'exp' subfolder, along with a corresponding JSON file delineating the facial key points represented in the model. Additionally, each folder includes a texture image (.png) and its corresponding image depicting normal vectors (normals.png), accompanied by an MTL file referencing the texture image of the model. Each folder represents a sequence of models corresponding to the 'exp - shape' combination and an example of this can be seen in Figure 7. In total, 1440 folders, each averaging 100 models, were collected, resulting in 1440 sequences of facial models or a grand total of 144,000 models.

Figure 7: Example of the first method which combines an exp folder and a shape face with the corresponding naming and structure of the new folder.

In the second phase of 3D models creation, the existing structure of the 'exp' and 'shape' files is retained and utilised, as previously outlined in the code, but with a variation in the combination of faces. Specifically, the DECA method mentioned earlier is tested on different actors within the dataset.

With 1440 folders in the 'exp' directory and a total of 24 faces in the 'shape' folder, pairing each face with every image folder results in a significant number of combinations, totaling 34,560 folders. Estimating an average of 100 obj models per folder, this yields a substantial 345,600 models, which can be cumbersome and time-consuming in subsequent steps of the method's development. To address this, specific criteria were established to prevent a person from the 'shape' folder from being paired with a folder from the 'exp' directory where they are already represented, thereby reducing the number of resulting combinations. The expression transfer strategy was implemented, involving the transfer of expressions (from one random intensity level for each emotion) extracted from four randomly selected individuals of the same gender to each actor.

Based on this strategy, 24 (subjects) x 8 (expressions) x 2 (statements) x 4 (subjects)= 1536 sequences of facial 3D models. However, an actor of the dataset may not have as many videos as someone else with these specific criteria, a fact that led to 1456 sequences of 3D models. In total, considering that on average each folder has 100 facial models, the new dataset of 3D models contains about 145,600 of them.

In terms of the nomenclature assigned to the newly generated data by DECA for both cases (the first with no emotion transfer and the second with emotion transfer), a specific naming convention was devised to differentiate between the combinations of two individuals at a time. In the initial scenario, with 1-1 actor combination, the merging of a folder derived from 'exp' with a frame from the 'shape' folder determines the name of the resultant folder. This results in the 'shape_exp' pattern being followed, where:

- shape: frame from an actor
- exp: expression folder with frames from 1 video

The names of the models in each folder resulting from the DECA method and the json files are numbered in 3-digit order, like the frames in the corresponding folder in exp. The texture images and mtl (wavefront Material Template Library file format) files are numbered in the same way.

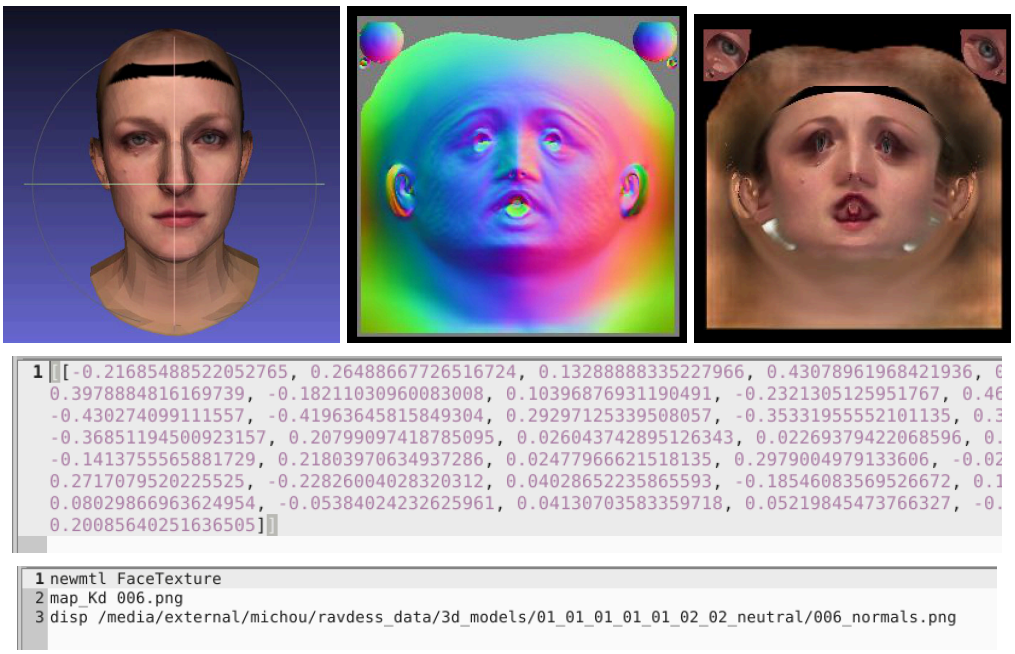


Figure 8: Example of the folder 01_01_01_01_01_02_02 with its content. From left to right, there are the 006.obj, 006_normals.png, 006.png, 006.json and 006.mtl files.

In the second scenario, with emotion transfer, a distinct approach is taken to determine the nomenclature of the data, as shown in Figure 9. This involves segmenting the file name retrieved from the 'exp' folder into distinct components using the hyphen symbol as a delimiter. A new naming pattern emerges, which diverges from the aforementioned 'shape_exp' convention. Instead, the naming convention follows a structure that is tailored to capture specific attributes or characteristics inherent to the data: 01_ExpressionID_Intensity_Statement_Repetition_Actor1_Actor2_Expression where:

- ExpressionID (01-08): neutral, calm, happy, sad, angry, fearful, disgust, surprised
- Intensity (01-02): Normal, Strong
- Statement (01-02): “Kids are talking by the door”, “Dogs are sitting by the door”
- Repetition: (01-02): 1st or 2nd repletion
- Actor_1 (01-24): 3D face of the corresponding actor.
- Actor_2 (01-24): Expression performed from the corresponding actor.
- Expression: neutral, calm, happy, sad, angry, fearful, disgust, surprised

Within each newly generated sequence, the models are arranged in a manner consistent with the frames from 'exp,' each bearing a unique three-digit serial number. Similarly, the JSON files follow the same pattern. The key distinction from the first case, where the expression transfer method was applied, lies in the

texture and normal vector images. In this scenario, only one texture image and one normal vector image are present, with each named after the actor selected from the 'shape' folder. The same principle applies to the mtl files.

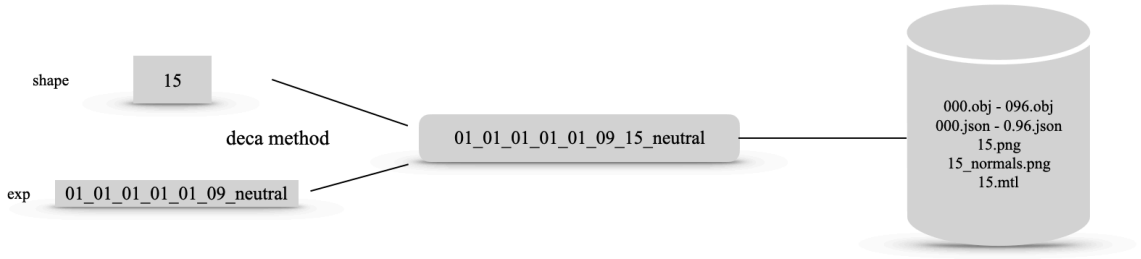


Figure 9: Example of the second method which combines an exp folder and a shape face with the corresponding naming and structure of the new folder

5.3 Correction of texture

The facial sequences generated by the DECA model in both scenarios accurately capture facial expressions, requiring no further refinement at this stage. Furthermore, upon examination of the images, it's evident that the facial key points have been aptly captured without the need for interpolating additional external elements.

The textures applied to the models are directly influenced by the input images provided, specifically the expression envelope (referred to as "e" in the code). Each image is framed by white surroundings, and the camera angle in each shot is a determining factor. Models depicting the front of the face capture textures with minimal issues, resulting in accurate representations. However, in models where textures are obtained from the left or right side of the face, such as when the actor rotates their head to convey a particular emotion, external elements like the white background colour become incorporated into the model, as shown in Figure 10. Another observation is that the DECA model, which aims to precisely render facial features, automatically eliminates the presence of hair from the texture image. Consequently, a black area appears on the forehead of each face, altering the texture's realism and diminishing its accuracy.

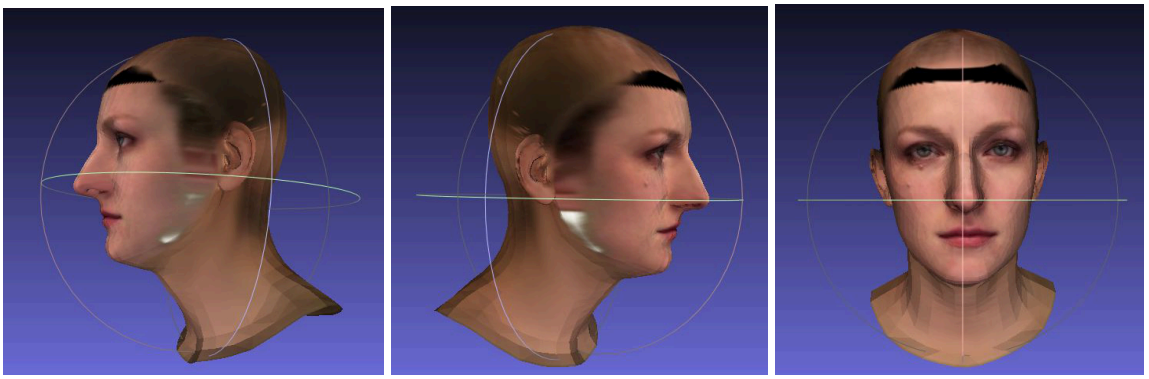


Figure 10: Example of the 3 views of a face - obj model employed by the first method in DECA

To ensure the accuracy and realism of the results, some efforts were made to rectify problematic textures. Initially, white or black sections that had been inadvertently added to the models were removed, replaced with the natural skin colour captured from the rest of the face. Following this adjustment, the subsequent step involved the addition of hair to the top of each model, enhancing their resemblance to real-life counterparts.

The initial method was primarily executed through a series of sequential code scripts. Its primary objective was using a masking technique to render colours for each actor-face with precision. This involved the creation of masks in varying shades of black and white, tailored to each individual's facial contours. Initially, specialised code was devised to solicit user input, allowing for the precise selection of pixels to generate the corresponding masks. The rationale behind not using a universal mask for all faces stemmed from the recognition that each face boasts unique imperfections in different regions. Adopting a blanket, larger mask would inevitably result in the distortion of crucial colours within the existing texture. Following the creation of 24 masks, a subsequent script was employed to infuse the textures with hues using both linear and radial blending techniques. The code, having as inputs a mask, a texture image and an intensity coefficient, successfully assigned skin tones to the white areas of the mask, which were obtained given specific facial coordinates. By executing these 2 blending techniques, 2 different texture images were generated for each actor, i.e. a total of 48 images. Upon comparison, no discernible difference was observed between the resulting textures for each face-actor. To maintain uniformity across all faces, the texture derived from linear blending was selected as the new corrected texture. Through this technique, the wrong colorations on the models' faces were largely corrected and the correction stages are shown through an example in Figure 11.

Figure 11: From left to right: the initial texture of obj model, the new mask and the results in linear and radial blending

The second method employed to rectify the textures involved the addition of hair using a mask and the Photoshop tool. This entailed creating a new mask based on the facial coordinates, replicating the colour and texture of hair to depict its presence on the head when applied. Subsequently, within the Photoshop environment, this mask was applied to the previously generated textures using linear blending techniques, as shown in Figure 12. The last step was to pass the new textures to all the files produced by DECA through appropriate scripts, based on the face represented in each one. The results of this procedure were corrected textures, given an example in Figure 13.

Figure 12: From left to right: the texture of linear blending, the new hair mask and the result of this combination.

Figure 13: The first row shows the frames of a video from the exp folder, the second the corrected textures as images and the third one shows the corrected textures applied in the obj models..

6.Webots

6.1 Introduction

Following the creation of 3D model sequences and their placement into folders, the next step is their transformation into animation. This requires the creation of a frame sequence for the representation of each model sequence. To achieve this, a simulation platform is imperative to visualise the models. Upon the accomplishment of this stage, the transition from images to video sequences ensues, complemented by the incorporation of the audio elements. Webots was used for this purpose.

Webots [3] is a free and open-source 3D robot simulator used in industry, education and research. It has been released under the free and open-source Apache 2 licence. Webots includes a large collection of freely modifiable models of robots, sensors, actuators and objects. In addition, it is also possible to build new models from scratch or import them from 3D CAD software. When designing a robot model, the user specifies both the graphical and the physical properties of the objects. The graphical properties include the shape, dimensions, position and orientation, colours, and texture of the object. The physical properties include the mass, friction factor, as well as the spring and damping constants. Simple fluid dynamics is

present in the software. It uses a fork of the ODE (Open Dynamics Engine) for detecting collisions and simulating rigid body dynamics. The ODE library allows one to accurately simulate physical properties of objects such as velocity, inertia and friction. It includes a set of sensors and actuators frequently used in robotic experiments, e.g. lidars, radars, proximity sensors, light sensors, touch sensors, GPS, accelerometers, cameras, emitters and receivers, servo motors (rotational & linear), position and force sensor, LEDs, grippers, gyros, compass, IMU, etc. The robot controller programs can be written outside of Webots in C, C++, Python, ROS, Java and MATLAB using a simple API. Webots offers the possibility to take screenshots and record simulations. Webots worlds are stored in cross-platform *.wbt files whose format is based on the VRML language. One can also import and export Webots worlds and objects in the VRML format. Users can interact with a running simulation by moving robots and other objects with the mouse. Webots can also stream a simulation on web browsers using WebGL.

6.2 Initial code

To initiate the visualisation of models in animated form, some Python scripts from a diploma project [13] were used and suitably modified, and new scripts were added. Collectively, these scripts constitute the generator project. The aim of the code is to generate a world file (.wbt) for each model sequence, where it records the data, i.e., positions the models within the scene. Prior to the development of the code, two files were created to serve as input data, setting the common grounds for visualising the outcomes.

First of them is world_ravdess.wbt which defines the scene layout and environment settings using VRML (Virtual Reality Modeling Language) in Webots. In particular, it defines the following:

- EXTERNPROTO Statements: These statements define external prototypes that are used in the scene. External prototypes allow reusability of complex objects or components defined in separate files. In this case, it's referencing various textured backgrounds, floors, and appearances from external sources on GitHub.
- WorldInfo: Provides general information about the world, such as its title, basic time step (time interval between simulation steps), and default damping values for objects.
- Viewpoint: Defines the initial viewpoint or camera position and orientation in the scene. It specifies the orientation (as a quaternion) and position coordinates of the viewpoint.

- TexturedBackground and TexturedBackgroundLight: These nodes define the appearance of the background in the simulation. They specify the luminosity of the background.
- CircleArena: This node defines a circular arena or floor in the simulation. It specifies parameters like radius, floor appearance, floor tile size, wall thickness, and wall height.
- SpotLights: These nodes define spotlights in the scene. Each spotlight is positioned at a specific location with a specified direction, intensity, and other properties like attenuation, beam width, and cut-off angle. These spotlights contribute to the lighting of the scene.
- DEF supervy Robot: This node defines the supervisor robot in the scene. It contains a camera and a speaker as children nodes. The camera specifies its field of view, width, and height. The speaker node is positioned at the origin (0, 0, 0).

The next file given in the input is template.txt. This part of the code defines a 3D object named "face_x" using the Solid node in VRML. The key points of this are:

- DEF: This keyword is used to define a named object that can be referenced elsewhere in the code.
- Solid: This node represents a solid object in the scene.
- Translation: Specifies the position of the object in 3D space. In this case, the object is translated to the coordinates (0, -0.07, -1).
- Rotation: Defines the rotation of the object in 3D space. Here, it rotates the object around the x-axis by 1.5 radians.
- Children: Contains child nodes or objects that are part of the "face_x" object.
- Solid (default object): Another Solid node nested within "face_x". This nested Solid represents the default appearance or geometry of the "face_x" object.
- Transform: This node is used to apply transformations like scaling to its children. It scales its children by a factor of 1.6 in all three dimensions.
- Shape: Represents the visual appearance of the object.
- Appearance: Specifies the material and texture of the object.
- Material: Defines the material properties of the object, such as shininess.
- Texture: Specifies the texture applied to the object. In this case, it references an image file “_t".
- IndexedFaceSet (geometry): This node defines the geometry of the object using indexed face sets. It specifies the coordinates of the vertices and texture coordinates for mapping textures onto the object's surface

Overall, the first part of the code (.wbt) sets up the scene environment, including background, arena, lighting, and initial viewpoint, to create the simulation environment in Webots. The second one, defines the appearance, geometry, and positioning of a 3D object named "face_x" in the scene.

The main script used to create a controller for each folder containing a sequence of models is called *supervisor_controller_ravdess*. This controller is crafted to oversee a simulated environment replete with cameras and sound. Its primary task is to capture images from diverse perspectives, regulate illumination, and execute other operations contingent upon simulation parameters. Central to its functionality are two pivotal helper functions: *look_at_rotation(source_pos_val, target_pos_val)*, which computes the rotation matrix for orienting the camera towards a specified target position, and *comp_euler_orientation(source_pos_val, target_pos_val)*, which calculates the Euler angles representing the orientation between two positions. Within this segment of code lies the *MyController* class, furnishing the framework for supervising the simulation and accessing its constituent objects. Notably, the *init* method within this class initialises the controller with various parameters, including whether to incorporate camera and audio elements, the simulation's time step, initialization wait time, image catalogue, and audio path. This segment of the code serves the purpose of initialising variables and retrieving information about individuals within the environment. Within the class structure, the configuration and positioning of cameras are handled based on specific parameters. Additionally, methods are incorporated to facilitate the adjustment of lighting, toggling between faces, switching camera perspectives, capturing images, and managing counters. The *run* method contains the core functionality of the controller, encompassing tasks such as image capture, metre adjustment, face switching, and halting the simulation as needed. In executing the code, the main function undertakes the setup of essential parameters such as the image directory, audio path, inclusion of camera and audio elements, simulation timestep, and initialization waiting time. Subsequently, it initialises an instance of *MyController* with the specified parameters and executes it. Finally, it concludes by terminating the simulation and exiting the program.

The next script, generator_ravdess, has been devised to streamline the creation of a series of simulated environments. Its primary function is to produce a folder structure akin to those found in the DECA dataset (3D models). This structure includes essential components such as a controller, a world file, a texture representing the individual, and an accompanying sound element. Initially, the script utilises two provided files: world_ravdess.wbt and template.txt, as foundational data. Through command line parameters, specific instructions are

provided. The obj_path parameter denotes the location of model sequences sourced from DECA, while img_path links to images generated in subsequent processes. The webots_dir parameter specifies the storage location for data generated during script execution, and audio_dir points to the audio files within the RAVDESS dataset. Lastly, the data_gen parameter relates to the inclusion of a camera-controller system. The data generation process comprises the following steps:

- Verification of Directory Existence: The script checks for the presence of corresponding directories within the designated Webots directory (webots_dir). If absent, it proceeds with the creation of Webots simulation files.
- Template File Utilisation: The script utilises template files (template.txt, world_ravdess.wbt, supervisor controller camera ravdess.py) to initialise the simulation environment.
- Sorting and Index Gathering: .obj files within the current directory are scanned and sorted. Specific lines from the first ".obj" file (000.obj) are processed to gather index information.
- Directory Creation: The script establishes directories within the Webots directory structure.
- Iterative processing of obj files:
    a. Vertex Data Extraction: Vertex data is extracted from each .obj file.
    b. Template Modification: Template lines are adjusted with vertex and texture information.
    c. Texture File Replication: Texture files are duplicated.
    d. Audio File Processing: Audio files are processed and exported in WAV format.
    e. Controller File Modification: Lines pertaining to audio paths in the controller file are adjusted.
- Optional Data Generation Modification: Lines for data generation are optionally modified.
- File Generation: The script generates Webots world and supervisor controller files based on the processed data.

6.2 Executions of generator project

The above scripts composing the project code generator were initially executed twice, due to the 2 different cases used to compose the dataset with the 3D models Despite variances in the internal folder structures, minimal alterations to the underlying code were required. The sole adjustment implemented during each execution instance pertained to the determination of which component within the

obj folder name should be utilised for texture retrieval and corresponding audio file copying. Subsequently, the data_gen parameter, introduced within the code, was initialised as a boolean, accepting either True or False. In instances where the parameter was set to False, a single camera facilitated all downloads, whereas in cases where it was set to True, a camera-controller system was employed. Consequently, four distinct executions were conducted: two for the dataset featuring a 1-1 component combination and another two for datasets with emotion transfer capabilities.

In the scenario of the 1-1 combination, a directory named ravdess_data was established (Figure 14), housing three subdirectories:

- **3d_models**: This directory contains the DECA model results.
- **img_dataset**: This directory is designated for utilisation in subsequent stages.
- **webots_sim**: Here, the results of the Webots simulation are stored, with the data_gen parameter set to False.
- **webots_data_gen**: This directory stores the results of the Webots simulation with the data_gen parameter set to True.

In the scenario involving expression transfer, a folder named ravdess_data_transfer was established (Figure 15), mirroring the structure described previously.

Regarding the structure of the webots_sim and webots_data_gen folders, a consistent layout is observed in both cases, featuring four subfolders. The first, named controllers, has a controller.py code file tailored for visualising the specific model sequence. The worlds subfolder contains an internal file named world.wbt, representing the corresponding world configuration. Additionally, the textures subfolder contains texture files, while the sounds subfolder stores sound files associated with the simulation. A notable distinction arises primarily in the textures subfolder and the internal files of the DECA results. In one instance, the textures correspond to the original frames from the video, resulting in a one-to-one mapping between frames and textures. In contrast, the other scenario entails a single texture image.

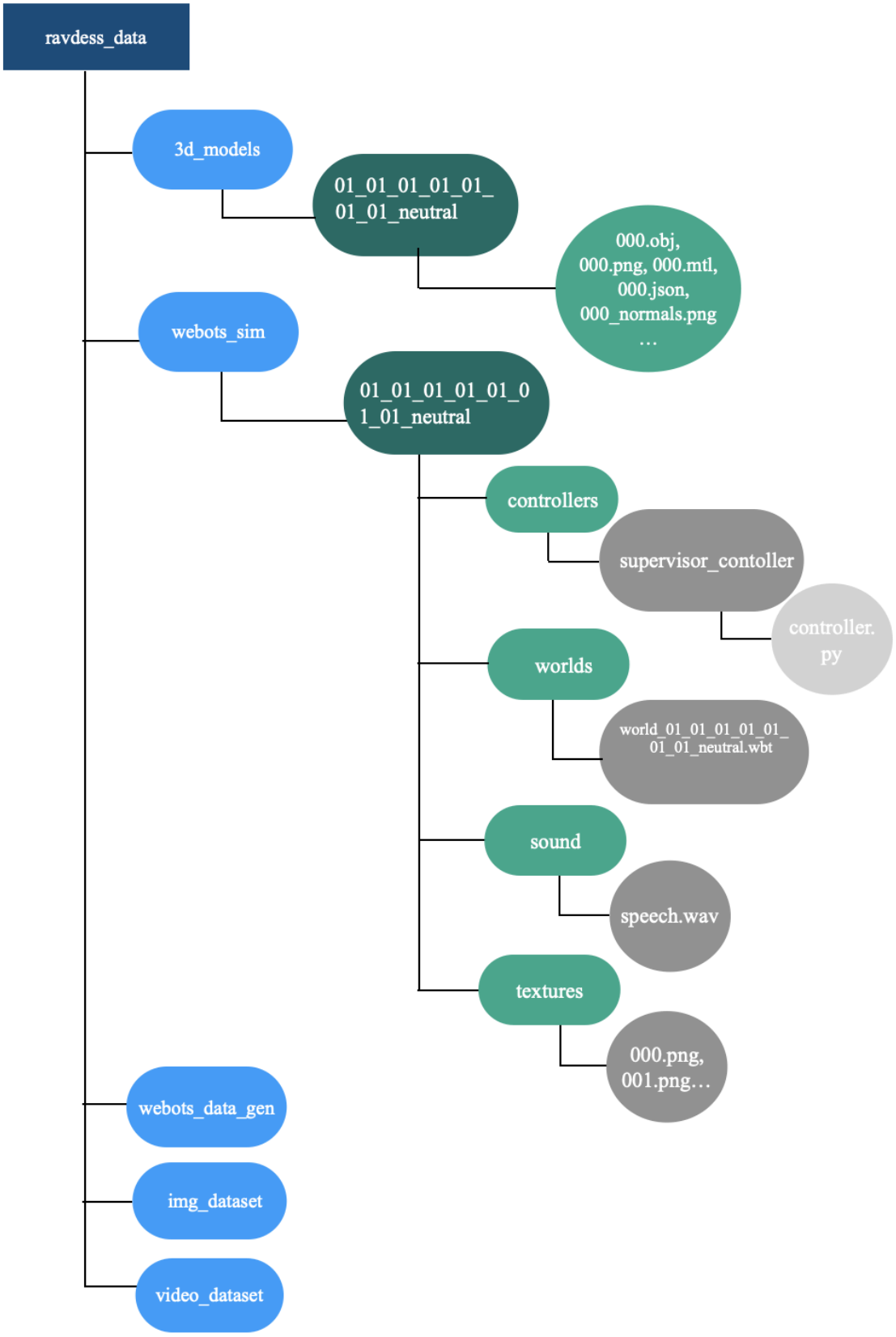


Figure 14: The diagram shows the structure of data in 1-1 combination of generator script.

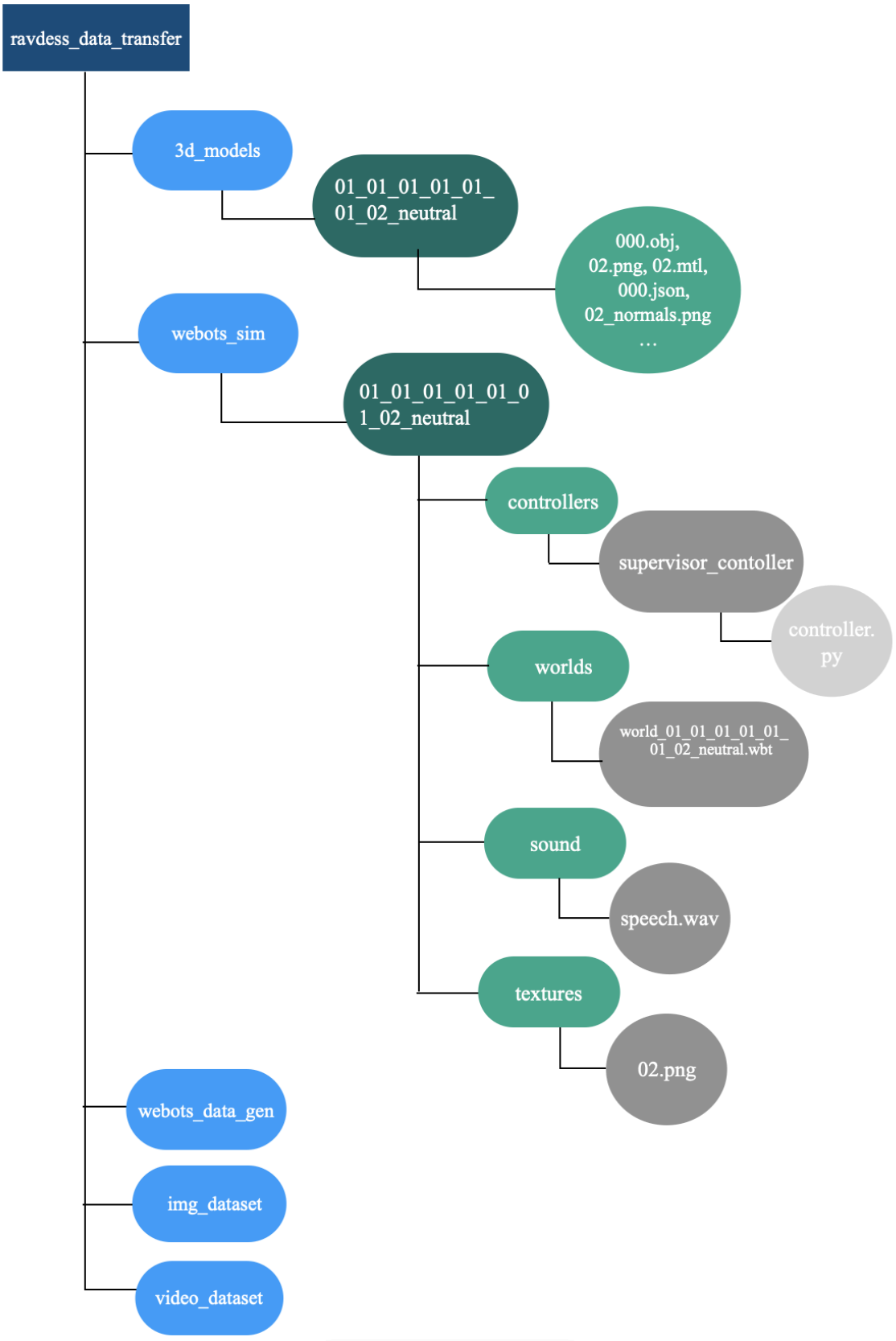


Figure 15: The diagram shows the structure of data in the expression transfer scenario.

Following the execution of the generator, a total of 2896 worlds or personification environments were created in Webots (.wbt file) for both the single-camera and multiple-camera scenarios. Specifically, 1440 worlds were collected for the 1-1 combination, and an additional 1456 were obtained for the emotion transfer case. The representation of a world from the collected data is notably simplified in the former set, where the human figure remains fixed at a certain point. These two specific datasets were retained and integrated into the overall dataset created through this code, synthesising a new dataset. Consequently, so far for the creation of the audiovisual dataset for use in facial expression recognition methods (especially active ones), 2 datasets have been created, one with 3D models and one with the generated world archives from Webots. The next step involves the production of sequences—animations from Webots—followed by their representation in video format. It is important that the data selected is exclusively derived from instances where the data_gen parameter is set to True, indicating the presence of multiple cameras.

6.3 Executor script

The final script written for the Webots personalization environment is named *executor_ravdess*, having the role of generating animations. Initially, the script compiles the names of the produced world file folders into a text file. Subsequently, it parses this designated text file (worlds_path), containing a roster of file paths. For every world path retrieved from this file:

1. The script assembles the directory path by appending '/worlds' to the directory housing the respective world file.
2. It adjusts the world file name to ensure adherence to a prescribed format and verifies whether it has already undergone processing.
3. When the filename format is recognized and has not yet been processed, the script crafts a command to execute Webots with the specified world file utilising xvfb-run—a tool facilitating X applications execution sans display. This command is then executed via os.system(cmd).

In essence, this script streamlines the execution of Webots simulations for each animation scenario identified within the specified directory, leveraging the predefined world files.

As for the image sequences generated through the execution of the executor, their details have already been described by the corresponding controller. In particular, this controller defines the lighting and camera position parameters *controller*. Specifically, this controller initialises the lighting and camera position parameters. With each frame, it ensures the correct object is displayed while concealing the previous one, subsequently capturing an image. These images are formatted as JPG files with a quality setting of 80 and dimensions of 600 x 600 pixels. Each image

follows a naming convention structured as follows: lighting - radius - height - horizontal angle - frame.jpg, where the parameters vary within predefined values:

- Lighting (0.25, 0.75): Dark, Light
- Distance (0.50, 0.75, 1.00): 0.5m, 0.75m, 1.0m
- Tilt (01-05): 01 -> -30.0, 02 -> -15.0, 03 -> 0.0, 04 -> 15.0, 05 -> 15.0
- Pan: (01-05): 01 -> -60.0, 02 -> -30.0, 03 -> 0.0, 04 -> 30.0, 05 -> 60.0

The *executor* script was executed twice: once with data sourced from the ravdess_data folder and once with data from ravdess_data_transfer. Within the executor's operation, a corresponding folder, bearing the same name as the data folder, is created to accommodate the generated image sets. For instance, by taking the folder 01_01_01_01_01_01_01_neutral from webots_data_gen in ravdess_data, a subfolder named 01_01_01_01_01_01_01_neutral is generated within the img_dataset directory. In total, the first run produced 1440 image folders, while the second run generated 1456. The number of frames within each folder typically ranges from 14,000 to 18,000, with an average of 16,000 images. Consequently, the resulting dataset comprises 2896 folders, each containing approximately 16,000 images, culminating in a dataset size of 46,336,000 images.

6.4 Creation of the video dataset

With the new image datasets representing the desired animations created through Webots, the next step is to build the corresponding video representations. To do this, another code script, named videoGenerator, was written. There, through the OpenCV and MoviePy libraries, the process of converting the images into video with the addition of audio is done. Taking the audio files coming from the RAVDESS dataset, the video generation begins through four nested loops, each of which corresponds to a parameter: light, radius, height and angle, as previously discussed. Within these, the script constructs detailed videos with various combinations of parameters, describing the following steps:

1. Folder Initialisation: Creation of output folders for videos with and without accompanying audio.
2. Parameter Iterations: Nested loops traverse through the assorted parameter combinations, constructing paths for both audio and non-audio versions of the videos.
3. Video Writer Initialisation: A video writer is initialised to facilitate frame writing into video files.
4. Frame Iteration: Within a nested while loop, each frame of the video undergoes processing. The script constructs the path to each frame image

based on the current parameter set, reads the image using OpenCV (cv2.imread()), and subsequently writes it into the video file.

5. Audio Integration: The script constructs the audio path by extracting tokens from the subfolder name and assembling the actor directory and audio filename. The audio files are sourced from the RAVDESS dataset. The real-world RAVDESS videos and the newly generated videos share the same number of frames and duration. This ensures that the corresponding audio is accurately aligned with the video content. The script reads the video file without audio using MoviePy (mp.VideoFileClip()) and the corresponding audio file using MoviePy (mp.AudioFileClip()). The video and audio are then merged using video_no_audio.set_audio(audio), and the final video with audio is saved to the output folder.

This systematic approach guarantees the production of videos with many combinations of parameters, supplemented by audio files, realistically capturing the essence of the original image datasets. In total, 2,896 video folders were created, corresponding to the number of input image files and their associated names. Each folder corresponds to a unique sequence of 3D models. Each such folder contains 150 videos resulting from the combination of two lighting conditions, three camera-to-face distances, and twenty-five angles of tilt and pan, ensuring complete coverage over a range of capture parameters. The dataset is presented in MP4 format, with videos at a resolution of 300 x 300 pixels. In addition, each video includes synchronised audio, aligned with the speech of the featured actor. Each video is identified by a file name with a .mp4 extension, following the pattern light_radius_height_angle. In total, the production yielded 434,400 videos with audio, resulting from the multiplication of 2,896 folders and 150 videos per folder. Each video has a duration of 4 seconds.

6.5 Final results

After running the executor_ravdess and videoGenerator scripts, the img_dataset folder within the ravdess_data and ravdess_data_transfer directories was filled with the images created during the process. Two additional subdirectories were then added to these folders: video_dataset, containing videos with synchronised audio, and video_tmp, containing videos without audio. To facilitate future procedures and save space, the video folders were compressed into zip files based on the corresponding sentiment. Consequently, seven zip files were created, each containing 182 subfolders of video with audio. The structure of data is shown in Figure 16. In total, through the process followed to generate the requested data, three datasets have been created: one with the 3D models produced by DECA model, one with the worlds files from Webots simulator and one with the animated

videos in audiovisual format. Some examples of the animated videos are shown in Figures 17, 18, 19 and 20. The datasets are publicly available here: https://zenodo.org/records/10711757

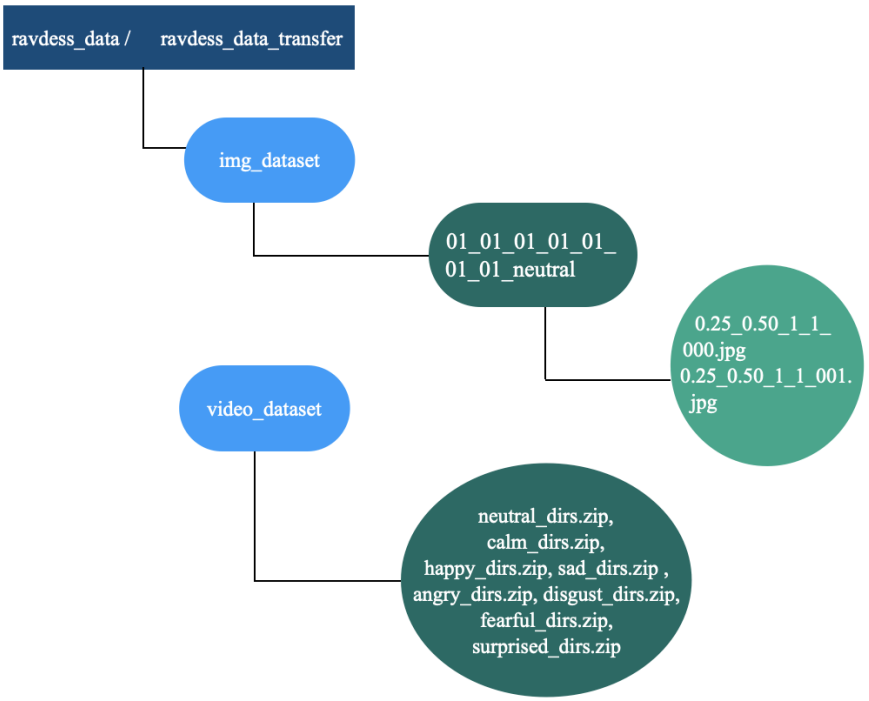


Figure 16: The diagram shows the structure of all data after creating the final datasets of images and videos.

Figure 17: Results of the Webots simulation where the 1-1 combination was used. The shown images are captured from different aspects, with distances of 0.5 and 0.75 to the camera, and in a dark mode.

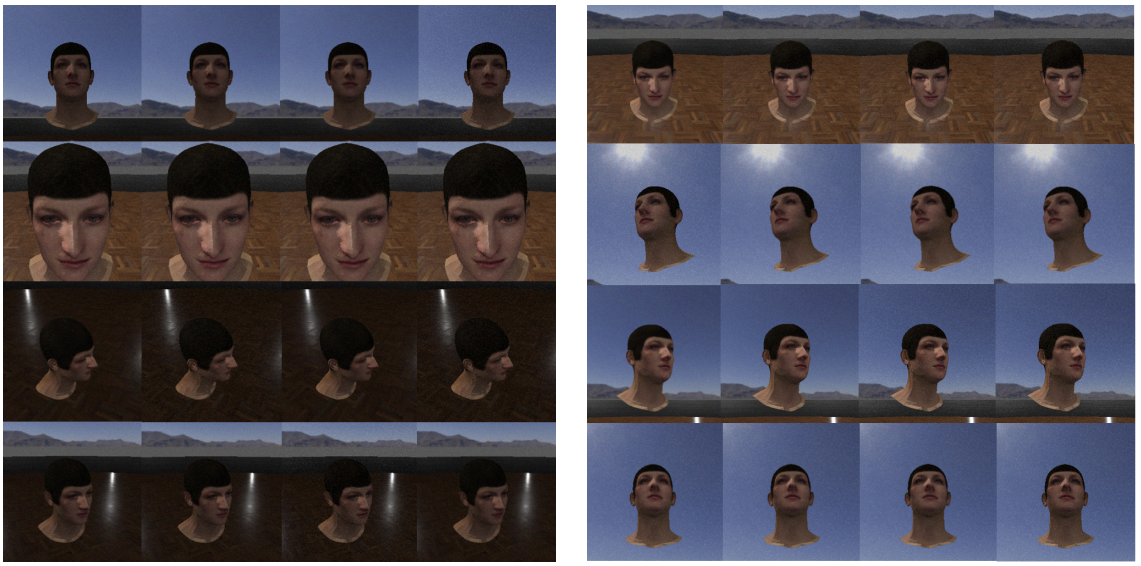


Figure 18: Results of the Webots simulation where the 1-1 combination was used. The shown images are captured from three different distances from the camera, two lighting modes, and various combinations of tilt and pan aspects.

Figure 19: Results of the Webots simulation where the method with transferred texture was used. The shown images are captured at a distance of 0.5 from the camera, with pan = 0 and different values of tilt.

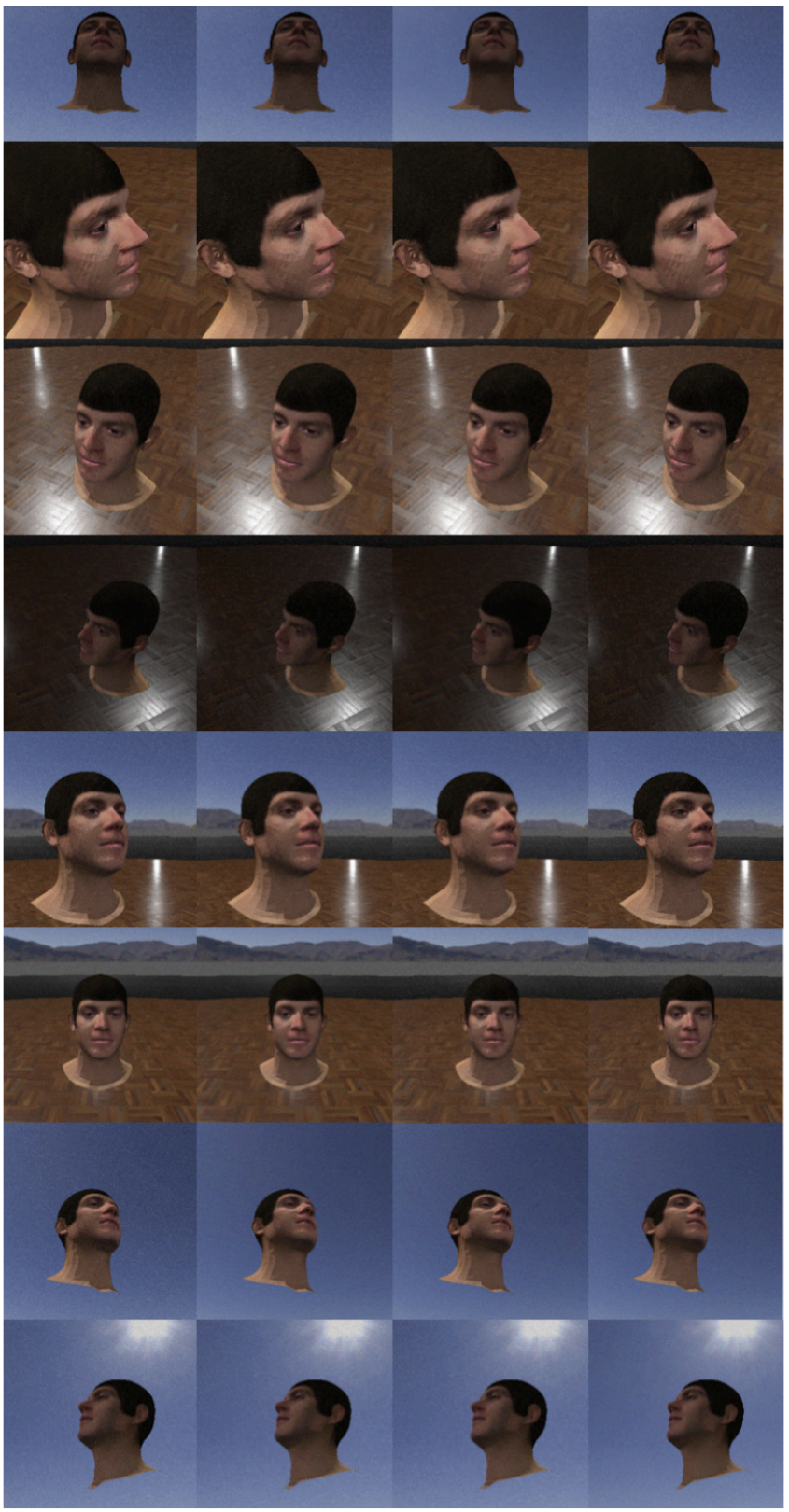


Figure 20: Results of the Webots simulation where the method with transferred texture was used. The shown images are captured at three distances from the camera and with different combinations of tilt and pan aspects.

7. Experiments

7.1 Introduction

In this thesis, the objective is to construct a dataset of audiovisual data generated from 3D models. We believe that based on human expressions, this dataset can be utilised in facial expression recognition applications and methods. A series of experiments were conducted, employing a state-of-the-art audiovisual emotion recognition method, aiming to assess the utility of the generated dataset.

7.2 Method description

The method [17] with which experiments with the new dataset were performed, creates a model that consists of two branches responsible for learning auditory and visual features, with the fusion units placed either at the end or in the middle of the two branches, depending on the type of feature fusion. Both branches use 1D convolutional blocks applied in a single time dimension. The vision branch consists of visual feature extraction from video frames and learning a joint representation for the entire sequence. Unlike traditional methods, feature extraction is integrated directly within the pipeline and optimised with the multimodal fusion module. Each video frame is processed independently by a 2D feature extractor, resulting in a vector descriptor for each frame. These descriptors are concatenated and processed temporally using 1D convolutional blocks, which are computationally efficient and leverage pre-trained 2D extractors from larger datasets. The vision branch can be adapted to use various feature types, such as deep features from pre-trained models or emotion recognition features like facial landmarks. Four convolutional blocks are used, each with a 1D convolutional layer, batch normalisation, and ReLU activation, to learn temporal representations. The audio branch processes feature representations, either pre-computed or optimised jointly, through four blocks of 1D convolutional layers. Each block includes a convolutional layer, batch normalisation, ReLU activation, and max pooling. Mel-frequency cepstral coefficients are used as the primary features, as other representations like chroma features or spectrograms did not show any additional benefit.

This section outlines the fusion approaches considered in the model, focusing first on the late transformer fusion method similar to those in existing literature, followed by two proposed intermediate fusion methods.

1. **Late Transformer Fusion**: Features learned from the audio and vision branches are fused using transformer blocks. Each branch employs a transformer that integrates features from the other modality. The outputs of these transformer blocks are concatenated and passed to the final prediction

layer. Specifically, the transformer's keys and values are obtained from the vision branch features, while the queries are derived from the audio branch features. The fused output is then used for the final prediction.

2. **Intermediate Transformer Fusion**: This approach uses transformer blocks for fusion at intermediate feature layers, specifically after the first stage of feature extraction (after two convolutional layers). Each branch integrates features from the other modality early in the architecture, allowing for joint learning of meaningful features during subsequent layers.
3. **Intermediate Attention-Based Fusion**: This method utilises dot-product similarity for fusion, leveraging the attention mechanism within transformer blocks. Queries and keys are computed with learned weights from two feature representations of different modalities. The scaled dot-product similarity, followed by softmax activation, emphasises important attributes, assigning importance scores to keys relative to queries. This process highlights relevant attributes of one modality based on similarity with the other modality, resulting in an attention vector that identifies the most relevant features. Unlike direct feature fusion, this approach uses attention scores to identify and emphasise key attributes, promoting feature agreement between modalities while maintaining their independence.

Modality dropout is used and addresses the issue of missing or unreliable data in multimodal learning by randomly masking or attenuating one modality during training. Three variants are proposed: zero-masking one modality to simulate missing data, scaling one modality by a random factor to prevent reliance on a single modality, and replacing one modality's data with random noise to handle noisy inputs. This approach improves performance even when both modalities are available.

One of the datasets used for training in this method was RAVDESS. The performance of the model was tested in three modes: only on audio data, only on video data, and on audiovisual data. When tested on audiovisual data, the model achieves accuracy from 76% to 81.58%, on only-audio data, from 15.16% to 59.16%, and on only-video data, from 17.33% to 74.92%.

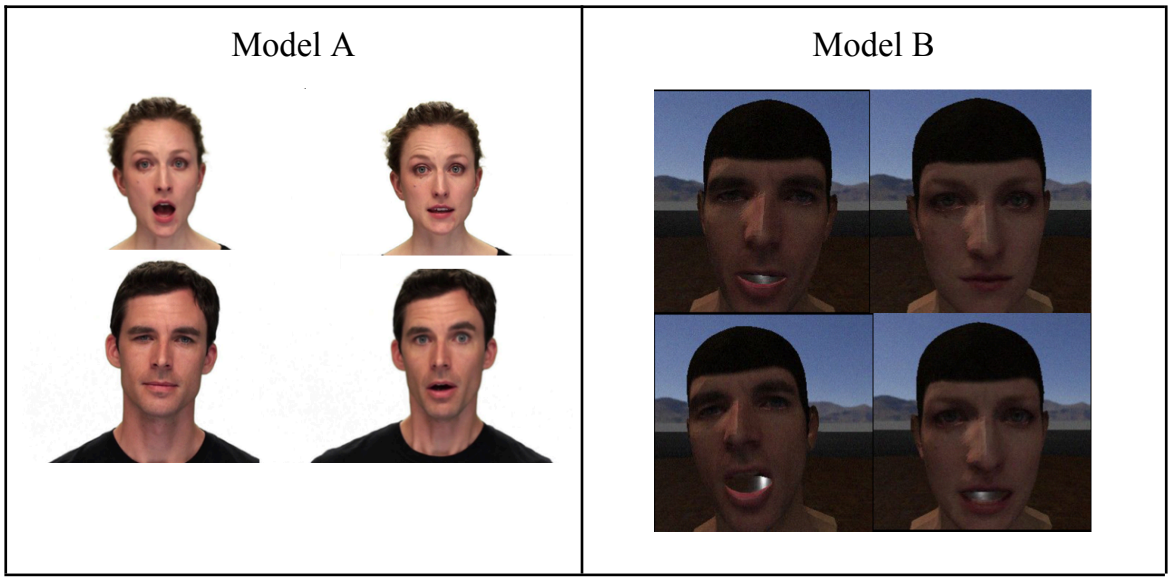
7.3 Training and evaluating on the new dataset

To assess the effectiveness of the generated dataset when used as training data, four training setups were created. The method presented in [19] was trained on each setup, resulting in four distinct models. From the fusion methods, IA (Intermediate Attention fusion) was used and from modality dropout, zero drop was employed.

The selected data did not include videos with emotion transfer but those from the 1-1 combination. These synthetic data, in video format, contain images - shots from 3 distances and 25 angles. The aim is to examine the impact of using synthetic images from viewpoints not available in the real-world training data. Regarding the actors in the dataset, actors 1-4 are used for testing, 5-8 for validation, and 9-24 for training in all four setups . In each of the four setups, separate training/validation sets were created for the training process of the corresponding model. An abstract visualisation of the corresponding training/validation sets is depicted in Figure 20. More specifically,

- The first model, model A, was trained using the training and validation sets of the real-world RAVDESS dataset which includes frontal images.
- The second model, model B, was trained using the training and validation sets that include only the close-up frontal (zero pan and tilt and distance equal to 0.5) shots of the synthetic data, meaning shots from angles and distances similar to those of the real-world RAVDESS.
- The third model, model C, was trained with all the synthetic data, namely videos captured from all angles (25 angles) and distances (3 distances).
- The fourth model, model D, was trained with a mix of real and synthetic data, using all data from the real-world RAVDESS and from the synthetic ones those not included in the former, i.e., videos from all angles and distances except the frontal close-ups(zero pan and tilt and distance 0.5).

The four models were evaluated on the same testing set, which is entirely synthetic and includes videos collected from all distances and angles. This approach emulates a real-case scenario where a trained model is deployed in images captured from diverse viewpoints. Finally, each trained model is assessed under three conditions based on the provided input: audiovisual, audio-only, and video-only.



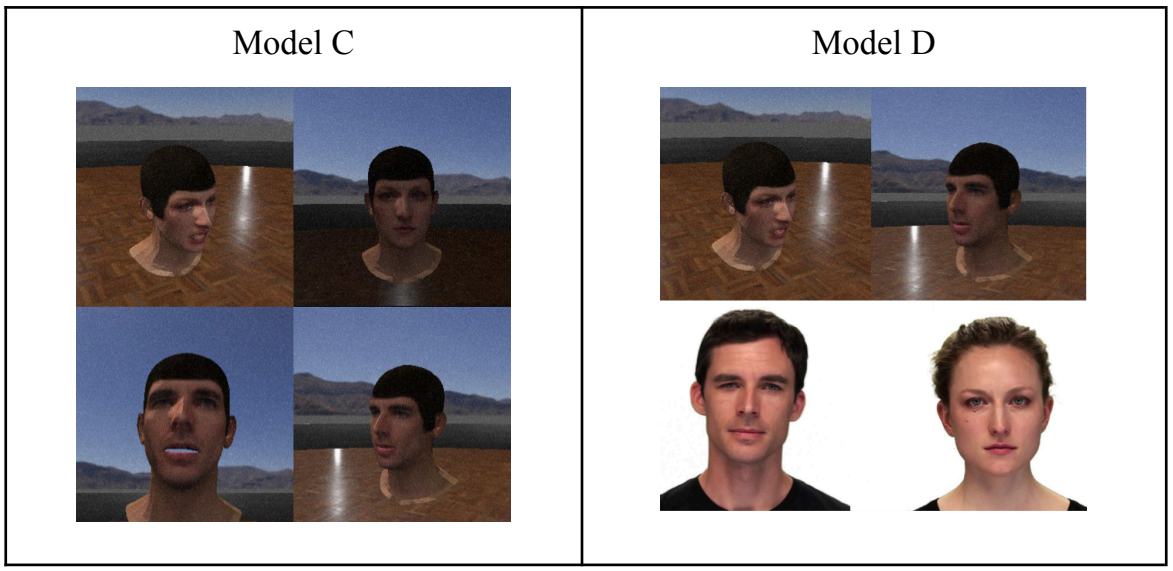


Figure 21: Visualisation of the training/validation sets of the four trained models.

7.4 Experimental results

To evaluate the performance of the four models and derive insights, their behaviour was examined based on specific parameters and conditions. This includes assessing performance, on frontal close-up shots, on data captured from multiple angles and distances, as well as combinations of these conditions. The results will be presented separately for each version: audio-only, video-only, and audiovisual.

Table 2 depicts the accuracy rates of the four trained models, separately for each input modality, tested on the entire test set that includes videos from all angles and distances. Comparing the expression recognition accuracy rates in the AV modality, models C and D, which were trained using recordings from multiple viewpoints, show higher rates by 20-25% compared to models A and B, which were trained using only frontal close-up shots, which was expected. The deviation between the performances achieved by models C and D is small (< 1%) and may be a consequence of domain shift. Both the test set and the training set of model C consist of synthetic data captured from multiple angles, while the training set of model D includes both synthetic and real data. For models A and B, where the test set includes data from different distances and angles not present in the training set, performance is better in the audio-only modality, with rates of 65.93% for model A and 56.7% for model B. Regarding the results in the video-only modality, models C and D perform better than models A and B, with a difference of around 30%, which is again expected since these models were trained on data similar to those in the test set. Similarly, Table 3 depicts the accuracy rates of the four trained models, separately for each input modality, tested on the frontal close-up shots, while Table 4 presents the accuracy rates when the models are tested on non-frontal shots. In both tables, the performance of the models is similar to that in Table 2, with models C and D achieving the highest accuracy on the AV modality. Between the only-video and only-audio modalities, the second performs better and, as expected,

is minimally affected by whether the test data are frontal or not. Comparing the rates between frontal close-up and non-frontal shots, models A and C achieve better rates when tested on frontal close-up shots, while models B and D perform better on non-frontal data across all modalities. It is interesting to note the performance models B and C in the video-only case: the performance of B (trained on frontal data) drops significantly when tested on non-frontal data, whereas C, which was trained on all view angles and distances, has similar performance on both frontal and non-frontal test data. Overall, the best rates in the table are achieved by models C and D using the audiovisual (AV) modality, with values of 73.94% and 73.5%, respectively. The following experiments will be conducted exclusively in the AV modality.

Model	**Accuracy (all shots)**		
	AV	**A**	**V**
A	49.64%	65.93%	20.52%
B	47.21%	56.7%	19.51%
C	73.94%	57.41%	53.32%
D	73.5%	58.99%	51.27%

Table 2: The success rates of models A, B, C, and D across different modalities (Audio-Only, Video-Only, and Audiovisual) tested on the entire dataset.

Model	**Accuracy (frontal close-up shots)**		
	AV	**A**	**V**
A	47.27%	65.45%	25.9%
B	65%	58.18%	41.82%
C	75.9%	59.09%	57.72%
D	70.9%	59.54%	47.27%

Table 3: The success rates of models A, B, C, and D across different modalities (Audio-Only, Video-Only, and Audiovisual) tested on frontal close-up shots.

Model	**Accuracy (non-frontal shots)**		
	AV	**A**	**V**
A	49.38%	65.82%	19.02%
B	46.63%	56.65%	17.42%
C	73.62%	57.07%	52.76%
D	73.96%	58.74%	51.79%

Table 4: The success rates of models A, B, C, and D across different modalities (Audio-Only, Video-Only, and Audiovisual) tested on non-frontal shots.

Table 5 depicts the percentage gains of model D compared to model A (D-A), presented separately for each combination of tilt and pan angles. Similarly, Table 6 shows the percentage gains of model C compared to model B (C-B), also presented

separately for each combination of tilt and pan angles. Although all distance values were considered in the analysis, they were not utilised in deriving the conclusions. This analysis focused solely on the variations in tilt and pan angles to determine the percentage gains. Starting with Table 5, it is evident that model D performs better than model A in all tilt and pan angles, as the difference D-A has a positive sign in all cases. In particular, in images depicting human faces frontally (i.e., with tilt and pan equal to zero), the performance gain is lower, at 17.42%, compared to cases where human faces are captured from large pan and tilt angles. This is again expected since D is trained on, and thus can recognize better than A, facial expressions captured from non-frontal views. As depicted in Figure 21, big performance gains are observed in images captured from cameras positioned at -60 degrees pan (i.e., in the first column) or at 30 degrees tilt (i.e., in the last row).

Similarly in Table 6, the positive differences between models C and B indicate that model C performs better than model B. The smallest value is observed for tilt = 0 and pan = 0, whereas for other pan and tilt values the gains are larger and there is a (non-consistent) tendency to increase for large values. This is once more expected as model B has not "seen" during training subjects from non-frontal views.

	Pan = -60	Pan = -30	Pan = 0	Pan = 30	Pan = 60
Tilt = -30	28.09%	23.53%	18.88%	23.92%	22.67%
Tilt = -15	27.36%	26%	22.08%	21.25%	21.11%
Tilt = 0	29.16%	25.82%	17.42%	17.37%	23.02%
Tilt = 15	21.59%	24.22%	22.28%	22.77%	22.38%
Tilt = 30	25.53%	27.78%	27.48%	28.02%	26.93%

Table 5: Differences in accuracy between Model A and Model D (D-A) for all combinations of tilt and pan values.

	Pan = -60	Pan = -30	Pan = 0	Pan = 30	Pan = 60
Tilt = -30	19.19%	20.45%	19.86%	39.22%	24.06%
Tilt = -15	16.11%	31.57%	31.25%	40.14%	35.42%
Tilt = 0	24.03%	25.29%	14%	32.76%	34.54%
Tilt = 15	31.56%	26.85%	15.6%	27.53%	31.33%
Tilt = 30	28.78%	27.16%	18.85%	25.22%	26.37%

Table 6: Differences in accuracy between Model B and Model C (C-B) for all combinations of tilt and pan values.

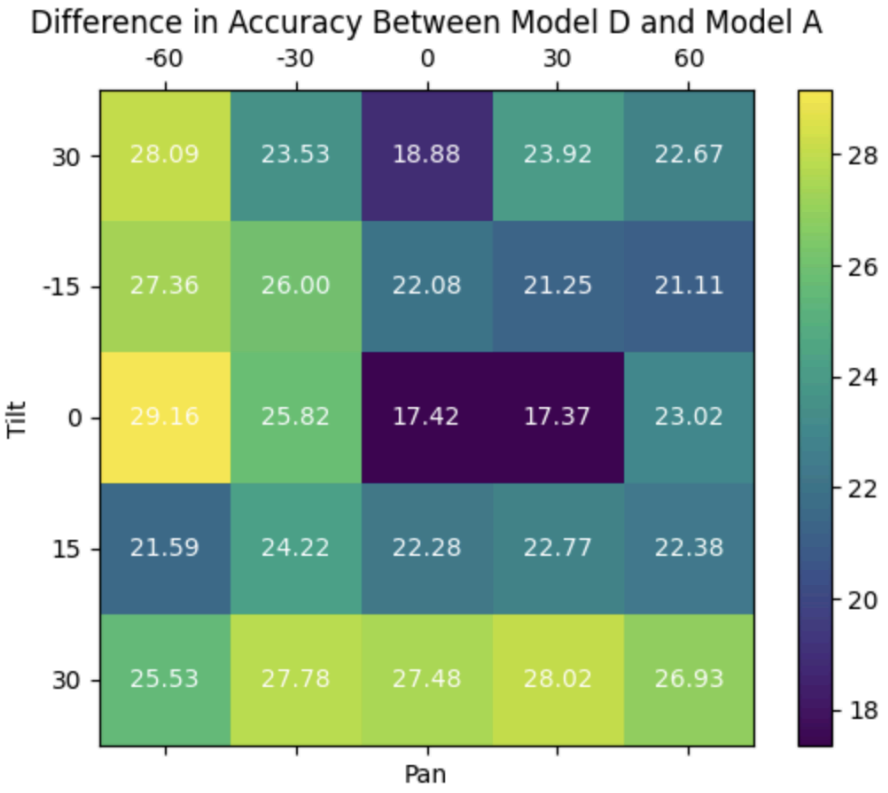


Figure 22: Visualisation (heatmap) of the rates presented in Table 5.

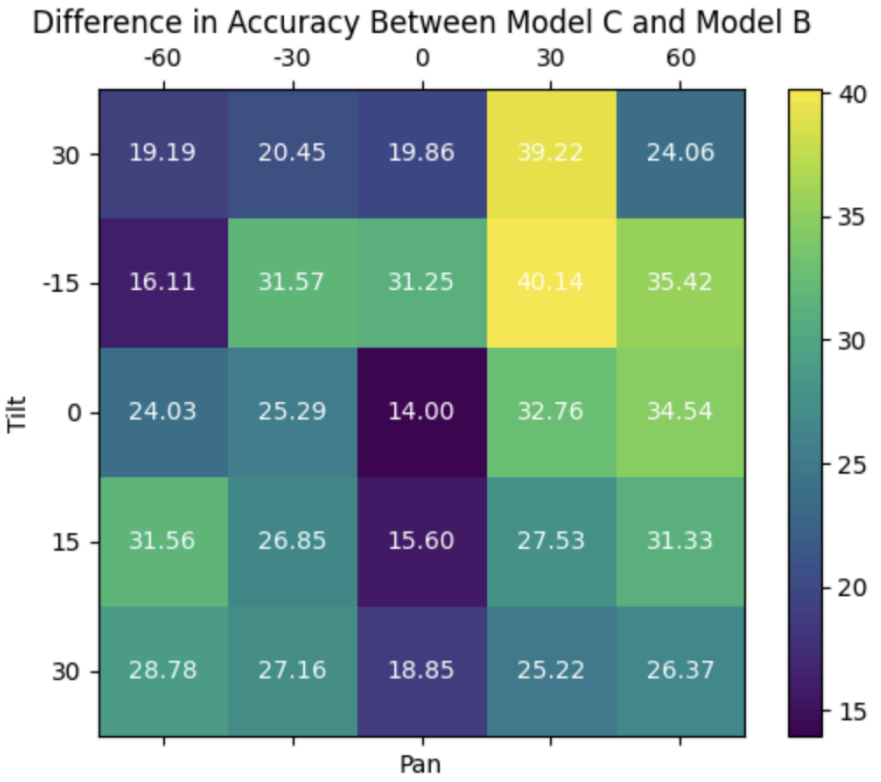


Figure 23: Visualisation (heatmap) of the rates presented in Table 6.

Table 7 depicts the accuracy rates achieved by each model on different camera-subject distance values (i.e., distances of 0.5, 0.75, and 1.00). All tilt and pan values were considered in the experiments. It is evident that the distance between the camera and the face doesn't have a significant impact on the performances of the models, since the accuracy changes when varying the distance are less than 3% (absolute difference) and, in most cases, significantly lower. Also, there is no clear trend in the performance change as distance increases. This can be first of all attributed to the fact that the changes in distance are rather small. Moreover, it can be attributed to the method employed, where a detector crops the face before expression recognition is applied. Thus distance changes affect the resolution of the face but, for the selected values, this has little (and inconsistent) effect on the performance. To conclude, distance does not influence facial emotion recognition accuracy, unlike previous tables where tilt and pan were shown to play a quite significant role.

	Model			
Distance	A	B	C	D
0.5	47.76%	45.5%	73.88%	72.45%
0.75	50.63%	48.6%	74.1%	74.04%
1.00	50.39%	47.39%	73.82%	73.92%

Table 7: Recognition accuracy for different distance values, models A to D (all combinations of tilt and pan values).

Having completed the experiments and analysed the results one can conclude that:

- The highest recognition accuracy rates are achieved when using audiovisual data.
- Models C and D (which are trained on frontal and non-frontal images) demonstrate strong performance both on frontal close-ups and shots from various angles and distances.
- Camera tilt and pan with respect to the face significantly influence model performance, whereas distance does not show notable impact.
- Models C and D perform better than models A and B when non-frontal views are considered.
- Enriching real frontal facial data with non-frontal synthetic data (model D) can significantly improve facial expression recognition performance on non-frontal data.

8. Discussion

During the course of this study, various methods related to 3D model creation from images, texture correction, and other aspects were examined, towards creating the audiovisual dataset. The results were particularly interesting and, when an audiovisual emotion recognition method was applied on the data, they were shown to be suitable for the task. However, several challenges were encountered during the course of the work.

8.1 Challenges and issues faced

The first challenge encountered pertains to the 3D models produced by DECA. As previously mentioned, there were issues with the facial textures, such as the inclusion of background elements in certain areas. To address these problems, the masking method was used, which significantly improved the models and corrected the textures to a large extent. However, creating a mask for each actor resulted in 24 masks, making the process time-consuming, especially if it is to be applied to more faces. An alternative solution is to use a larger mask that can be applied to all faces. This approach was avoided to ensure corrections were limited as much as possible to the erroneous regions, preventing the mask from affecting other areas of the face and ensuring smooth colour transitions. Moreover, the addition of hair was necessary for the realistic rendering of faces. Overall, the results were enhanced by using both methods in conjunction.

The creation of animation sequences in the Webots simulatιov environment posed another challenge. As described in Chapter 6, the adopted approach details how a world is created with all objects pre-prepared. A specific program collects data from all the obj files, integrates them into the scene, and incorporates them into the world while generating the corresponding controller. The disadvantage of this method is that it produces large .wbt world files, which occupy significant disk space and are slow to load. This is a problem in tasks like the current one, where the number of world files exceeds 1000. A suggested solution might involve using simulation data that corresponds to the obj files but requires less storage space.

A further drawback appeared in the final animation files, resulting from the Webots environment. Despite realistically modelling the scene with appropriate environment and lighting, as the subjects open their mouth to utter a sentence, the background colour of the scene appears in the mouth gap, due to the fact that the 3D face model is not a closed shape.. An attempt was made to fix this by placing a black shape in this area via code in Webots, but it yielded more problematic results. Due to the variety of faces and emotion transfers, the mouth is not always in a consistent location across all models. Consequently, in a few models, the issue

seemed corrected, while in most cases, the black shape appeared in an incorrect position on the face. Therefore, it was preferred not to use this solution and to leave the animation sequences as they were.

As already mentioned, the goal was to create a final dataset for use in facial expression recognition methods that consists of videos rather than images. Webots, as we configured it, captures images of the models at different distances and angles, resulting in a set of images. As described in Chapter 6, these images were compiled into videos using a single script, and these videos needed to include audio, something Webots did not directly provide. Adding sound required special treatment, during the emotion transfer process. In this case, the sound had to be taken from the depicted face (and not from the subject from which the expression was taken) to avoid issues such as a male subject speaking with a female voice.Obviously the speech data were synchronised with the expression spoken in the video.

8.2 Conclusions

Methods for human expression recognition are constantly evolving, with an increasing number of tools being developed due to their necessity in various fields. The aim of this work was to create a data generation process that would provide a new dataset to be used with such methods. In real-world applications where expression recognition is examined, close-up shots alone are not sufficient. Enriching them with shots that vary in distance and angle can significantly improve their performance.

Using the DECA model, 3D facial models were generated from the RAVDESS dataset, capturing detailed expressions and facilitating expression recognition. This tool allows the separation of face, expression, and texture, which led to the development of two methods for creating the new dataset: reconstruction of expressive models from a certain subject and generation of models that feature the geometry and texture of one subject and the expression of another one (expression transfer). By doing so we managed to increase the number of data, with respect to RAVDESS.

The Webots simulator was the software that assisted the models' animation and the production of the final dataset. In this environment, each sequence of models was placed in a scene with two different lighting conditions. By placing the virtual camera in three different distances from the subject and 25 different view angles, multiple shots were captured, in the form of video frames (images). Finally, through a script, the images were converted into videos. Audio corresponding to the depicted face was added, forming the final publicly available dataset containing audiovisual files. The 3D models and the Webots simulations were also made available. The fact that videos from various distances and view angle are included

makes the dataset suitable for research on active or view-invariant facial expression recognition, but also, potentially, for face detection and recognition

With the new dataset available, experiments were conducted to evaluate the performance of an audiovisual emotion recognition method. The experiments examined the data that did not include the emotion transfer method but rather the one reconstructing 3D models of the same face. The collected results led to the conclusion that enriching datasets with those produced here, which include not only frontal close-up data but also data from different angles and shots, can improve existing methods. Specifically, comparing the model containing only frontal close-up shots from RAVDESS with the one enriched with the audiovisual dataset produced in this work, the performance in the audiovisual testing set was significantly better. The experiments also led to other interesting results and conclusions.

In summary, the method developed in this thesis for constructing a new dataset proves to be highly useful for generating data for facial expression recognition methods and other related applications. The synthetic data produced here is capable of complementing real data in a dataset, thereby enhancing performance in application evaluation. By following these steps, our aim is to contribute to the development of similar datasets and improve expression recognition systems.

Bibliography

1. Steven R. Livingstone, Frank A. Russo. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. PLoS ONE 13(5): e0196391. 2018. https://zenodo.org/records/1188976
2. Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. DECA Learning an Animatable Detailed 3D Face Model from In-The-Wild Images. ACM Transactions on Graphics (ToG), Proc. SIGGRAPH 20 Αυγούστου 22. https://github.com/yfeng95/DECA
3. Cyberbotics Ltd. Webots (Version 2023a) [Computer software], 2023. https://github.com/cyberbotics/webots
4. Sumit Saha. A Comprehensive Guide to Convolutional Neural Networks. Towards Data Science, Dec 15, 2018.
5. Gaudenz Boesch. What Is Computer Vision: Applications, Benefits and How to Learn It. Viso.ai, January 20, 2023.
6. Hivi Dino, Subhi R. M. Zeebaree, Maiwan B. Abdulrazzaq, Amira Bibo Sallow. Facial Expression Recognition based on Hybrid Feature Extraction Techniques with Different Classifiers. Research Gate, ISSN: 0193-4120 Page No. 22319-22329, May-June 2020.
7. Lijun Yin, Xiaochen Chen, Yi Sun, Tony Worm, Michael Reale. A High-Resolution 3D Dynamic Facial Expression Database. The 8th International Conference on Automatic Face and Gesture Recognition. 17-19 September 2008 (Tracking Number: 66)
8. Anil Audumbar Pise, Mejdal A. Alqahtani, Priti Verma, Purushothama K, Dimitrios A. Karras, Prathibha S, and Awal Halifa. Methods for Facial Expression Recognition with Applications in Challenging Situations. ACM, Computational Intelligence and Neuroscience, Volume 2022, 01 January 2022.
9. Konstantina Vemou, Anna Horvath. Facial Emotion Recognition. European Data Protection Supervisor, Issue 1, 2021.
10. Facial Expression Recognition. Raydiant, 22 August 2022. https://www.raydiant.com/blog/facial-expression-recognition
11. Paweł Tarnowski, Marcin Kołodziej, Andrzej Majkowski, Remigiusz J. Rak. Emotion recognition using facial expressions. ScienceDirect, Volume 108, Pages 1175-1184, 2017.
12. OpenDR - Audiovisual Emotion Recognition Model: https://github.com/opendr-eu/opendr/tree/3676a2652ebd82adf4fc773213d2efe14c84dc0c/src/opendr/perception/multimodal_human_centric/audiovisual_emotion_learner

13. Michail Chatzakis. 3D Face models dataset creation for expression recognition applications. Diploma thesis, Aristotle University of Thessaloniki, 2022.
14. Tianye Li, Timo Bolkart, Michael J. Black, Hao Li,Javier Romero. FLAME, Learning a model of facial shape and expression from 4D scans. ACM Transactions on Graphics (TOG), Volume 36, Issue 6, Article No.: 194, Pages: 1-17, 20 November 2017.
15. PyTorch library: https://pytorch.org/
16. CUDA ToolKit: https://developer.nvidia.com/cuda-toolkit
17. Kateryna Chumachenko1, Alexandros Iosifidis and Moncef Gabbouj. Self-attention fusion for audiovisual emotion recognition with incomplete data. Cornell University, 26 Jan 2022.
18. Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In Advances in Neural Information Processing Systems (NeurIPS), Pages:331-340, 2018.
19. Rebecca Caroll. What is Computer Vision? IBM, July 14, 2021. https://www.ibm.com/blog/computer-vision/
20. Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2D & 3D face alignment problem? (and a dataset of 230,000 3D facial landmarks). In IEEE International Conference on Computer Vision (ICCV). 1021–1030, 2017.
21. Andries Engelbrecht. Engineering Applications of Artificial Intelligence. Stellenbosch University, Volume 120, April 2023.
22. Gaurav Gupta. Webots with ROS. Black Coffee Robotics, July 17 2023. https://www.blackcoffeerobotics.com/blog/webots-with-ros-simulation-overview
23. Stuart Russell and Peter Norvig. Artificial Intelligence A Modern Approach (3rd edition), April 2011.
24. Richard Szeliski. Computer Vision: Algorithms and Applications 2nd Edition, January 5, 2022.
25. Ajaykumar Devarapalli, Jora M. Gonda. Investigation into facial expression recognition methods: a review. Indonesian Journal of Electrical Engineering and Computer Science 31(3):1754, September 2023.