October University for Modern

Sciences and Art

Faculty of Computer Science

Graduation Project

Live Human Face expression Augmentation by Projection

Supervisor:DR. Ahmed Farouk

Name:Mohanad Mohamed Ibrahim

ID:182533

Abstract

we going to develop a system for human expression augmentation by using projection mapping. technique by searching many approaches and algorithms that were widely used in projection mapping we decided on using deep learning approach to mimic the human face expression. We estimate the blend shape values of the input expression by using a Convolution Neural Network (CNN) algorithm. We take a human facial expression image as an input and the model output the blend shapes of human facial expression (the input image) then we take these blend shape values and run it into the 3d model to projects the mask with the expression of human face.

Contents

List of Figures

List of Tables

1 Chapter1: Introduction

1.1 Introduction

Before technology, we used colors and painting tools to deliver an artistic message through the buildings, objects, and people's faces. We can still do that, we are still in need for a creative outlet and a way to represent our culture and art through the objects around us and our faces, only this time, we are doing it using technology. By the help and use of projection mapping techniques. Let us introduce what is projection mapping; it is a computer graphic technique to display images or videos on real-world surfaces. We can transform our entire reality into an Spatial Augmented Reality. We can even do things in the augmented reality that we can't do in actual reality, like restoring the original color of an old object without affecting the physical object itself [figure 1] [1].

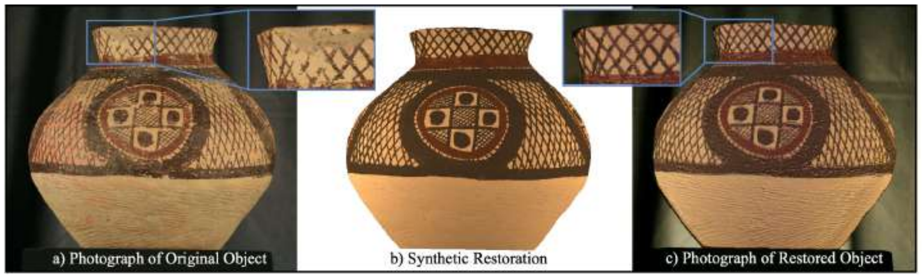

a) Photograph of Original Object
b) Synthetic Restoration
c) Photograph of Restored Object

Figure 1: restoring original color of old object [1]

and in this case, we managed to preserve our history and still present it in its real authentic look and value. It's also possible to affect the appearance of a moving object by using multiple projectors to track the object and change its appearance by blending multiple projected images[4]. There are many applications and magic that we could do with projection mapping, but our main focus in the project is to manipulate the appearance of a human face[2] to highlight the emotional expressions in the face to make art. We will be able to achieve this by developing a system that take a image of the human face expression from a direct angel,then will apply it into a 3D model that we developed using

Artificial intelligence, and the output texture will be projected on the user's face to highlight the expression [figure 2][2]. Nowadays projection mapping also has

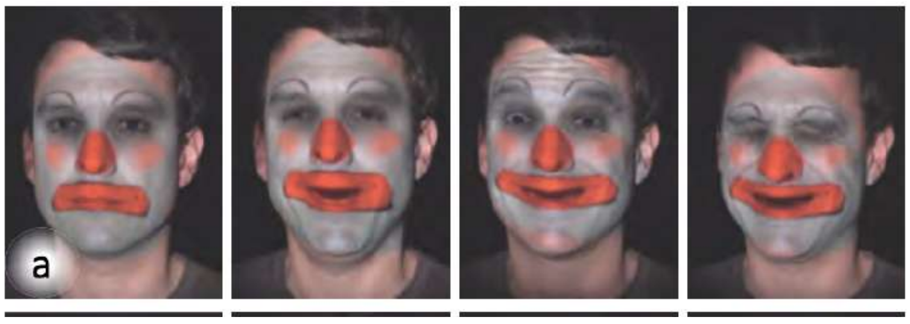


Figure 2: the output of Augmentation of Human Faces [2]

some entertaining applications like RoomAlive[4] it transforms our living room reality into Spatial Augmented Reality see [fig 3][4], the user can interact with the game by shooting and touching the projected game, we favor the way they used projection mapping to deliver a fun and unique experience, they developed a new system by making the projected game content dynamically adapts to any living room and the system also captures the player interaction through Microsoft Kinect. However there are many different challenges in RoomAlive, one of them is system latency and how it can negatively affect the user interaction with the game.

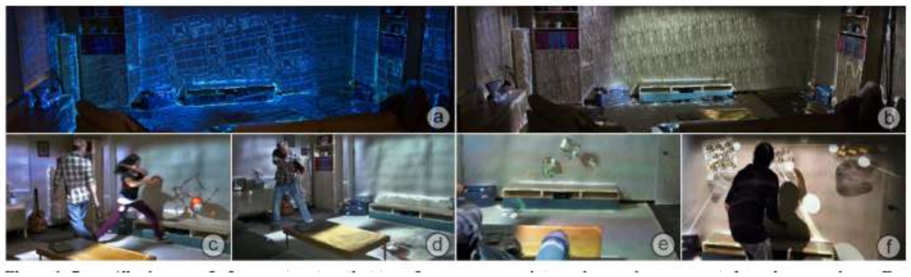


Figure 3: user's interacting with RoomAlive [4]

1.2 Problem Statement

Using projection mapping on the human face to change its appearance holds many challenges; The main challenge we face (in Spatial Augmented Reality) is ability to mimic the human expression whether it is showing a happy, angry or any other facial expressions to allow us to project the appropriate image on it. Another problem we face is system latency because our system needs to mimic the human facial expressions and then it generates a project-able image of a texture mask and place it on the captured positions of the human face. This process takes time and by the time our system is projecting the appropriate image, the subject has already moved his/her face and made another facial expression that our system needs to mimic it. Resulting in projecting the image on the old subject facial position issue. We try to overcome this issue by using the human face tracking system[5] or by estimating human face positions[2], it helps to track human face faster and recognized it [figure 4][2].

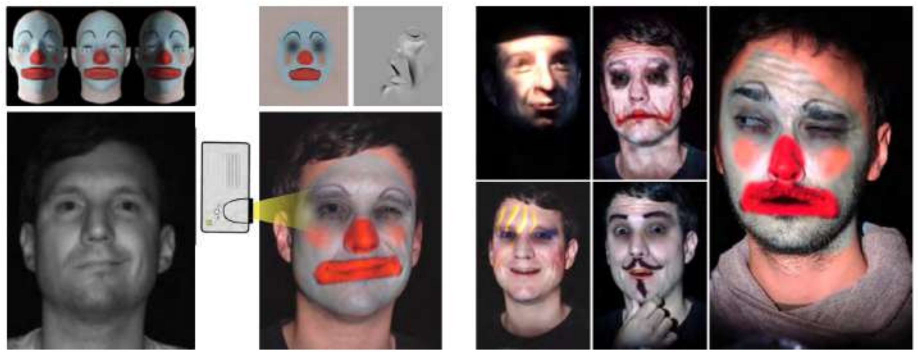


Figure 4: Projection Mapping in human face [2]

1.3 Objective

this project should deliver an artistic message by using the augmentations of the human face to change its appearance in order to highlight the emotional expressions. The project will take a image of the human face from only one view using a direct angle, after that it will analyze the facial expression and will deform our 3d model to generate the projected image for projection[2].

1.4 Motivation

Nowadays we are able to use projection mapping in theatrical performances to give the audience a new fascinating theatre experience. Projection mapping is also used in the movie industry, they use it to render faces and different makeup on actors without the need for them to go through the hassle of reapplying makeup and face costumes, thus resulting in saving a lot of production time and giving the viewers a better movie experience. Also it has a powerful impact on advertising as it can be used to change the appearance of an object for better presentation[1] . These are few examples of the many applications such as its use in museums to restore the original color of ancient monuments.

1.5 Thesis layout

In conclusion, we now know what is projection mapping and how it can be helpful in museums, art, or even in movies. In chapter two; we are going to dive in its background to see its application and algorithms and approach. In chapter three we will propose our system method and how it could work.In chapter four we going to our system overview and implementation phases and how our system running and see limitation,In chapter five we going to evaluate our system and testing it,In chapter six we are going to discuss our technical issues and scientific issues and our future work.

2 Chapter 2:Background and Literature Review

2.1 Background

there are many approaches and algorithm that are widely used in projection mapping. One of them is Dynamic spatial augmented reality, this approach requires the 3d position of the object or the human face, to obtain this information this approach needs a machine Learning algorithm to find the posture, the initial position of object, and the edge-based tracker see [fig 5][3].We can use a deep learning approach to get the blend shape weights to deform the 3d model to give the 3d model the same expression as the human face and to recognize the human face expression. In order to get a semi-accurate estimation of the human face head position there is Kalman filtering algorithm[2]. Additionally the Levenberg–Marquardt algorithm, can help in solving non-linear problems found in 3d human face parametric model[5]

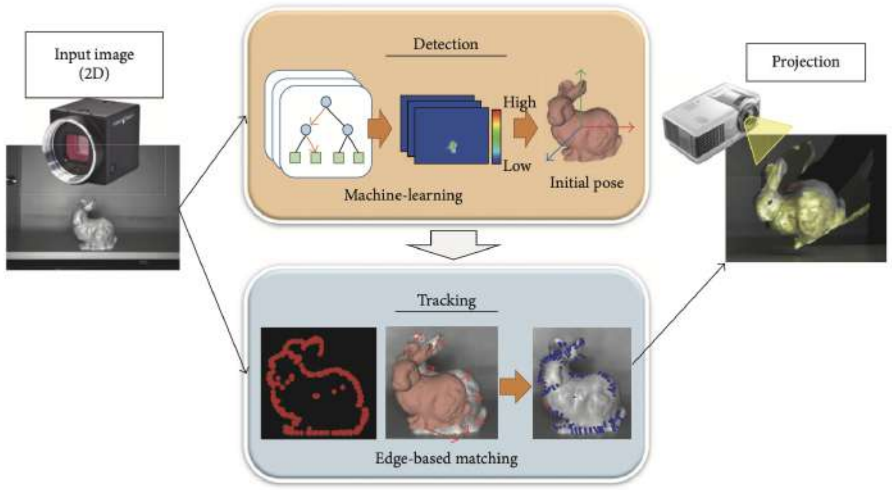


Figure 5: process of Dynamic spatial augmented reality[3]

2.1.1 Machine Learning

Machine Learning is a part of Artificial Intelligence that depends on training data. In order to teach the machines to perceive and act by itself, We use a Decision tree method to obtain and to collect the posture and the 3d position of human face or object. The Machine Learning has two-layer only before it needs to collect the training data of different rendering of 3d model from different aspect which are classified to use it see [figure 6][3]

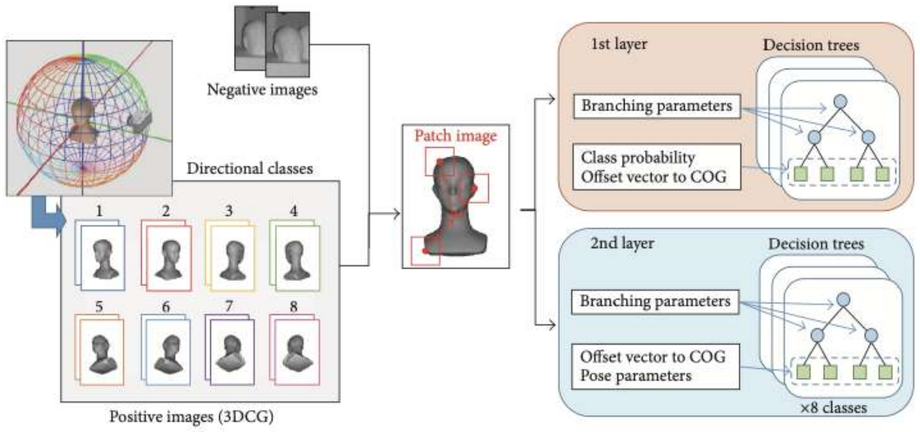


Figure 6: Machine Learning Algorithm Process[3]

2.1.2 Edge-based tracker

Edge-based tracker depends on edges so we must make the 3d model edge match the real target edge to find the position of an object and to find the initial position of the object see [figure 7] [3]

2.1.3 Deep Learning

Deep Learning is a part of Artificial intelligence that simulates learning like a human brain to recognize and detect the expression of the human face to make a decision. Deep Learning has many classes; one of them is Convolutional Neural Network (CNN) that are used for video and image detection. We can use

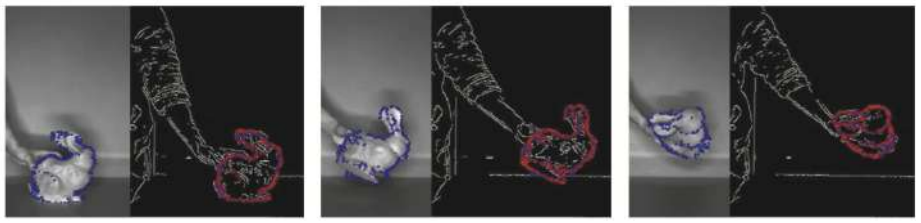


Figure 7: Edge-Based tracker [3]

Convolutional Neural Network to detect the emotional expression of the human face to set the weights of 3d model to mimic it.

2.1.4 Kalman filter

the advantage of the Kalman filter algorithm is to infer and find the unknown values using known values. It can help reducing system latency time by estimating the human head position if the user gets off the camera vision it can help to predict the next position of the head. Kalman filter is also used in navigation in map application [2]

2.1.5 Levenberg–Marquardt

levenberg–marquardt algorithm is a combination of Gaussian newton and gradient descent used to solve non-linear problems, it can help us to measure and calculates the translation and rotation scales in 3d parametric human face model[5].

2.2 Previous Work

2.2.1 Makeup Lamps: Live Augmentation of Human Faces via Projection

2.2.1.1 Strategy and Structure

the researchers' goals are to track the human face and analyze its expression to highlight the emotion of human face by using projection mapping. In favor of better projection they are reducing the system latency by deforming and rendering the facial rig of 3d model. To restore these expressions in position table and the video is captured using a high-speed camera under infrared light, the facial expression and landmark are estimated by Kalman filter algorithm after that they deform 2d mesh and project it [figure 8][2].

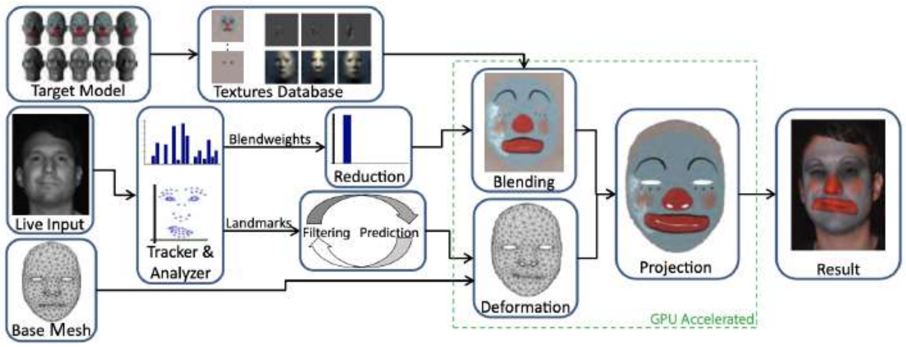


Figure 8: Structure of Makeup Lamps system [2]

2.2.1.2 Data

the data they are using are a 3d blend-shape model that have weights to set for facial configuration, they have 6 blendshapes rendered in different positions and expressions. They collected and stored the albedo map of the 3d model to reduce the size of data, resulting in them using only 6 expressions of the 6 blendshapes[figure 9][2]

2.2.1.3 Method Evaluation

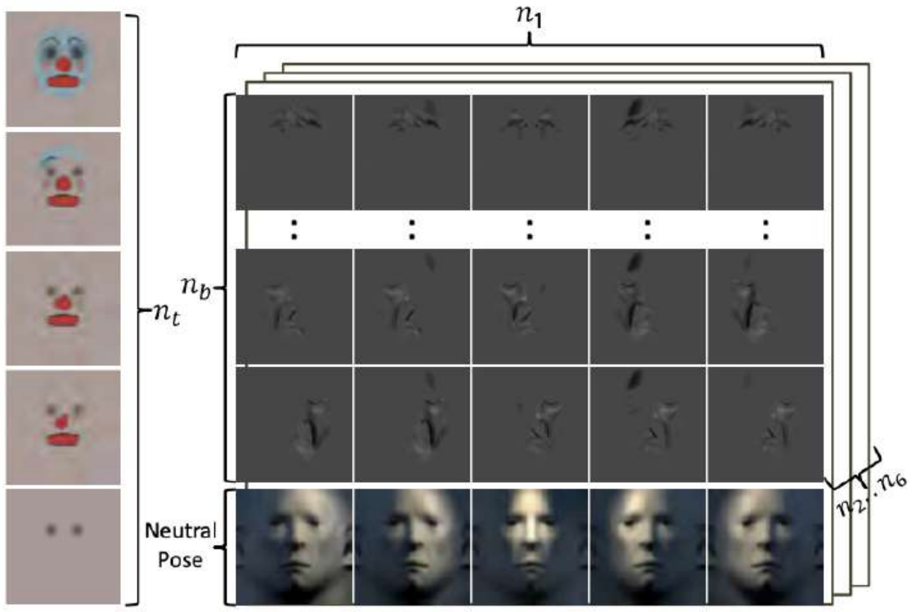


Figure 9: albedo map [2]

the first non-rigid human projection system that is using the Kalman filter algorithm has improved the performance because it can estimate the position of the human face saving time and reducing errors. The Christie Mirage 4K35 DLP projection system is being modified and used, they modified it by using more lights for better resolution to reduce latency [figure 10] [2] the weaknesses of the system is the facial tracker because the use off-shelf tracking system had not recognized all facial expression and the tracking noise[2].

2.2.1.4 Results Evaluation

the overall average system latency take 9.8ms and by using CPU and GPU parallel the details in the table as shown [Table 1][2]

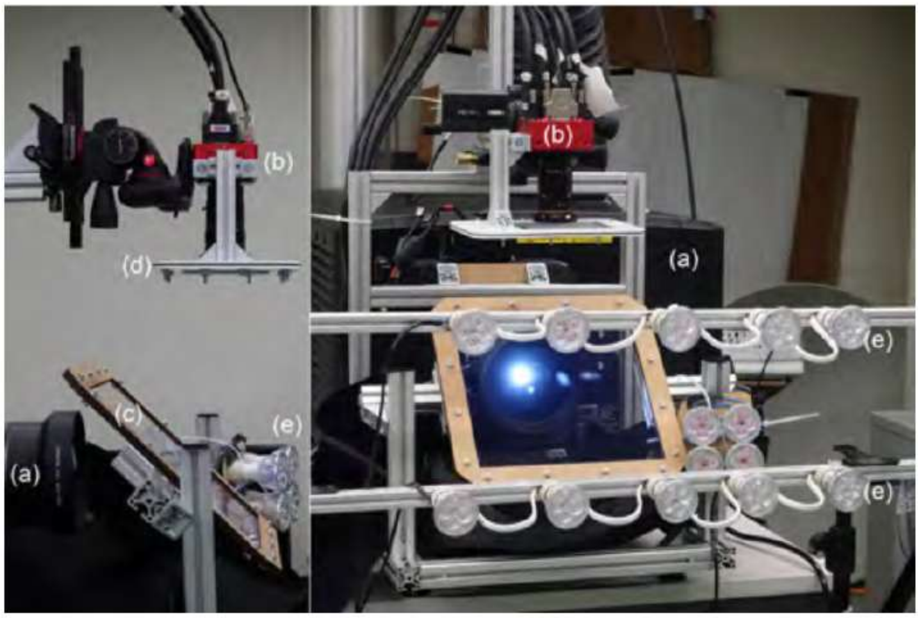


Figure 10: the customized projector [2]

Task	Average Time	Standard Deviation
Image processing (CPU)	195.6 s	22.9 s
Face tracking (CPU)	5671.3 s	2851.9 s
Blendshape reduction(CPU)	25.4 s	9.1 s
Data filtering (CPU)	95.8 s	27.8 s
transform(GPU)	100.2 s	2.0 s
Textured rendering (GPU)	180.7 s	4.6 s

Table 1: The proposed system latency of Makeuplamps

2.2.2 FaceForge: Markerless Non-Rigid FaceMulti-Projection Mapping

2.2.2.1 Strategy and Structure

An overview of the system they use multi-projection and high camera RGB-d camera to make construction and track the human face. The colored image is being used only in the Creation phase. The tracking algorithm depends on depth image they deploy a face model but it's a 3d parametric model. For example morphable model[5] are using Levenberg-Marquardt algorithm to count the scales of translation and rotation see output [figure 11] [5]

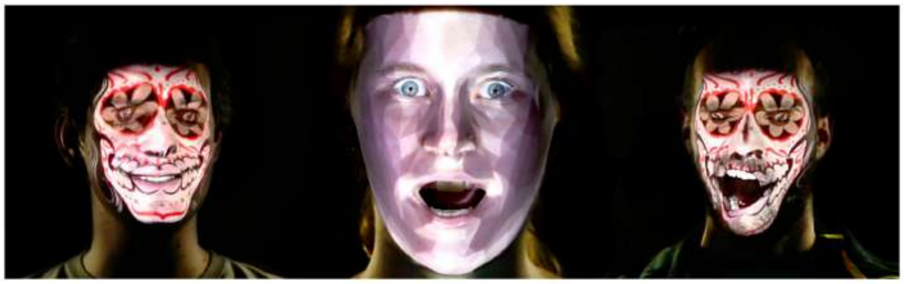


Figure 11: output of FaceForge [5]

2.2.2.2 Data

the researcher had used a 3d human face model but it's a parametric model. The benefits of the parametric model is that it can share the same texture size even in other human faces and it can render from a different view and it also has 76 blendshape weights for facial expressions. The average of albedo was collected to configure the data and it has 80 identities shape the size of the model is 106466 triangles and 53490 vertices see [figure 12] [5]

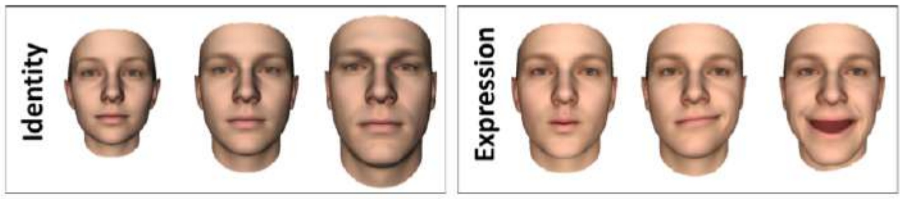


Figure 12: FaceForge 3d parametric model [5]

2.2.2.3 Method Evaluation

The first non-rigid tracking multi-projection mapping approach was proposed by them, they usage of the 3d parametric model had a great impact in run time, construction, and in rendering. The multi-projection technique had helped covering the whole human face. The zebra technique was used to give a good light balance to solve the override light problem. The disadvantage of this method is system latency because the use of multi-projection and it is not able to recognize all facial expressions[figure 13][5]

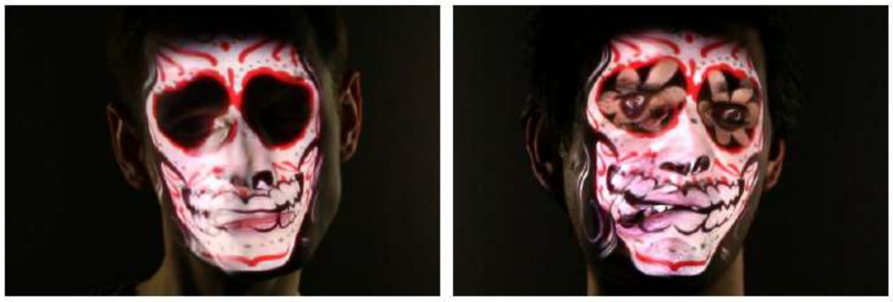


Figure 13: FaceForge facial expressions errors [5]

2.2.1.4 Results Evaluation

To Evaluate the result we must identify the main task. First the human face tracker takes an average 10.4 millisecond and to solve the light scene it takes 6.2 milliseconds and the rendering the projected image it takes 2.7 milliseconds the overall system take 19.3 milliseconds [5]

3 Chapter 3: Material and Methods

3.1 Materials

3.1.1 Data

by using a 3d model in unity editor we are able to edit the expression, the 3d model we are using has a 74 blendshapes weights that enable us to control the expression. We managed to download 854 picture from the internet, each person has 7 expressions poses that includes happy, angry, natural, disgusted, afraid, sad, and surprised as shown in [figure 14] and [figure 15] All pictures are in "JPG" format and the 3d model in [figure 16]

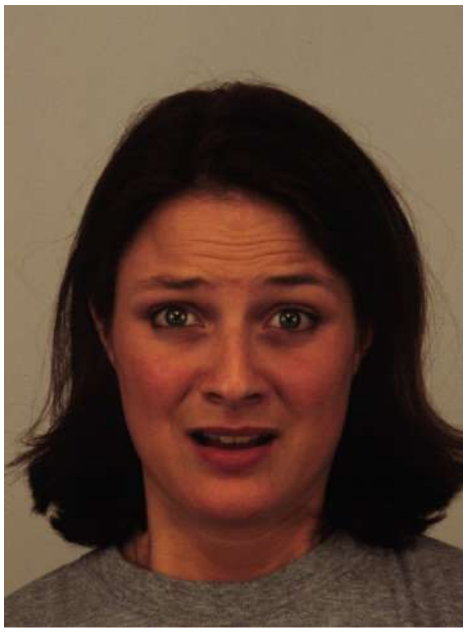


Figure 14: the sample data of afraid person expression

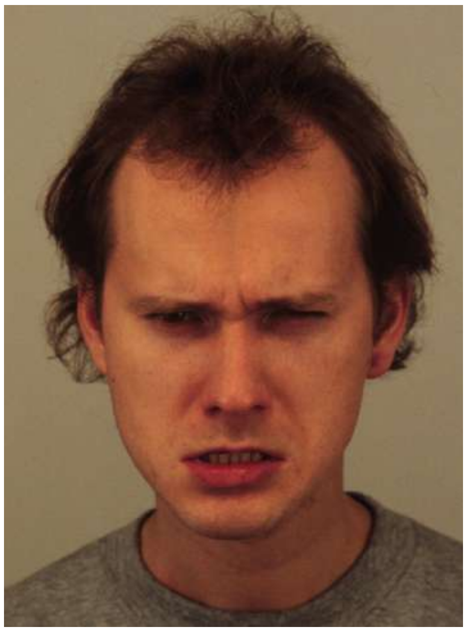


Figure 15: the sample data of angry person expression

3.1.2 Tools

-projector

-Unity : game engine to set weights of blendshape of the 3d model.

-C#: programming language.

-Google Colab : is a online cloud platform for developing and applying AI and deep learning model.

-paython:programming language.

-numpy; a library to help in deep learning.

-Anaconda: is a software for developing and applying AI and deep learning models.

-RGB camer(xbox 360 kienct device) for better tracking.

3.1.3 Environment

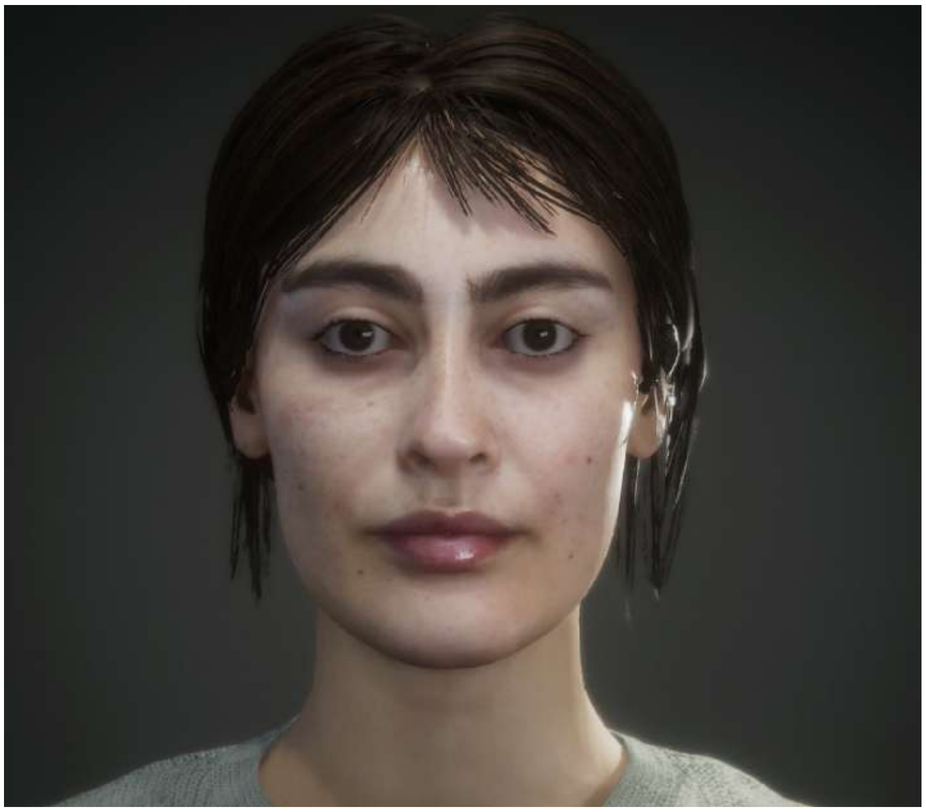


Figure 16: the 3d model

Processor :1.4 GHz Quad-Core Intel Core i5

Ram :8 GB 2133 MHz LPDDR3

Graphics :Intel Iris Plus Graphics 645 1536 MB

TensorFlow Cloud: to develp the CNN network on it

3.2 Methods

3.2.1 System architecture Overview

Before we started planning the system architecture, we needed to gather the data and process it. After that, our system starts by taking an image (input) and use our deep learning approach with Convolutional Neural Network algorithm (CNN) to generate the 74 weights of blendshape that enable our 3d model to change and mimic expressions of the input and accomplish the best similarity. After training our neural network will be able to manipulate any 3d model and its belndshape weights to get a perfect position that matches the user expression.[figure 17][5][2]

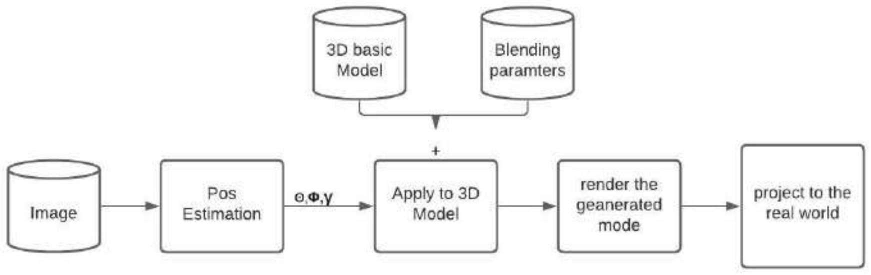


Figure 17: our system architecture

In [figure 17] Our system architecture starts with image and then it estimate the human facial expression after that we apply it into the 3d model then we render the generated mode finally we project the texture into the real world.

4 Chapter 4: System Implementation

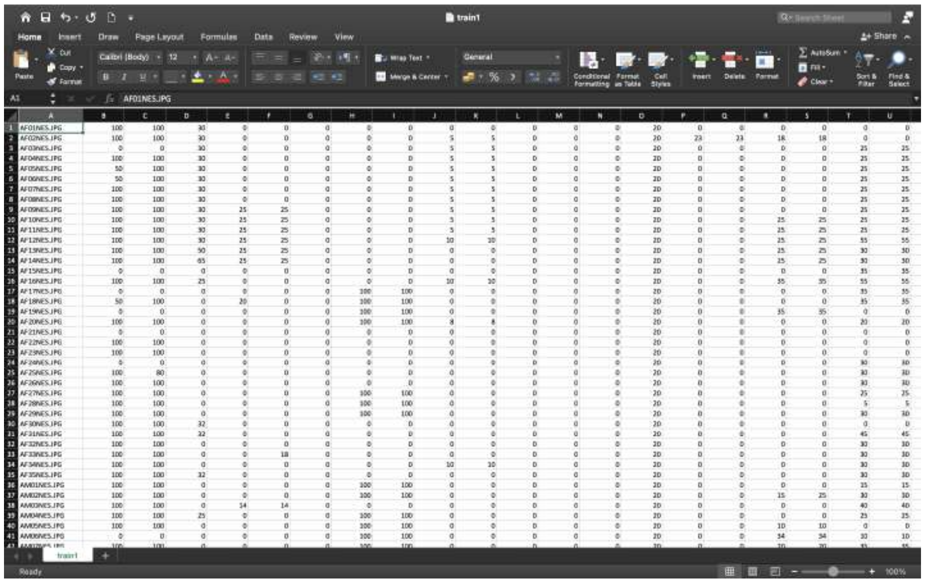
4.1 System Development

the system was developed in different phases. First, we had to ensure that the 3d model blendshapes can read and write the blendshapes weights from C# code so we tried class Blendshape in unity C# to set and get the Blendshapes of the 3d model to move her mouth and eye and jaw and so on. Second, in the second phase we start to build our datasets by using 854 images from internet and the 3d model. We build our dataset by making the 3d model take the same expression of the person by using the 74 blendshapes. We assign to the model the accurate number that represents the expression from range 0 to 100 [figure 19] and store these 74 blendshape numbers in a comma-separated values file [figure 18]. The process was very time consuming. It took time, effort, screen and processing time to achieve the accurate expression of the images we used to build our data-set. The tricky part, was the human face. Humans have fascinating facial expressions that were inflexible to mimic. The reason behind the difficult part of mimicking the human facial expression was the fact that it's all connected, for example, when we clench our jaws in case of anger, our cheek squints and eyebrows slightly go down, all this happens with one expression at the same time. In case of the 3d model, when we clench the jaw, the face still looks happy, not angry, there for, in order to have the accurate feeling, we need to play with different (and often too many) blendshapes, which takes a huge amount of time and multiple trail and error sessions [figure 20]. The third phase is the testing phase. We must test the images and its values on Convolutional Neural Network (CNN) model to ensure that every image is taking its own accurate blendshape.

Then we implement our Convolutional neural network (CNN) model to take input image and the output is the blendshapes values. Eventually, we use our CNN results to create the output and run it through the 3d model. One of the issues was the fact that the data-set images were not all in the same size, the train date imagws were 562×762 and the test data images were 896×896. In order to get around this issue, we had to resize all input images to 400×400 by using Keras APIs [figure 21].

Another challenge was the CNN using only one color channel, therefore, we had to apply a gray-scale filter on all input images [figure 21]. To accomplish everything that we mentioned above, we had to use many programs and platforms (Unity, Tensorflow, Spyder, Google Colab, Visual Studio).

As we see here in [figure 18] in the "CSV", the first column represents the image

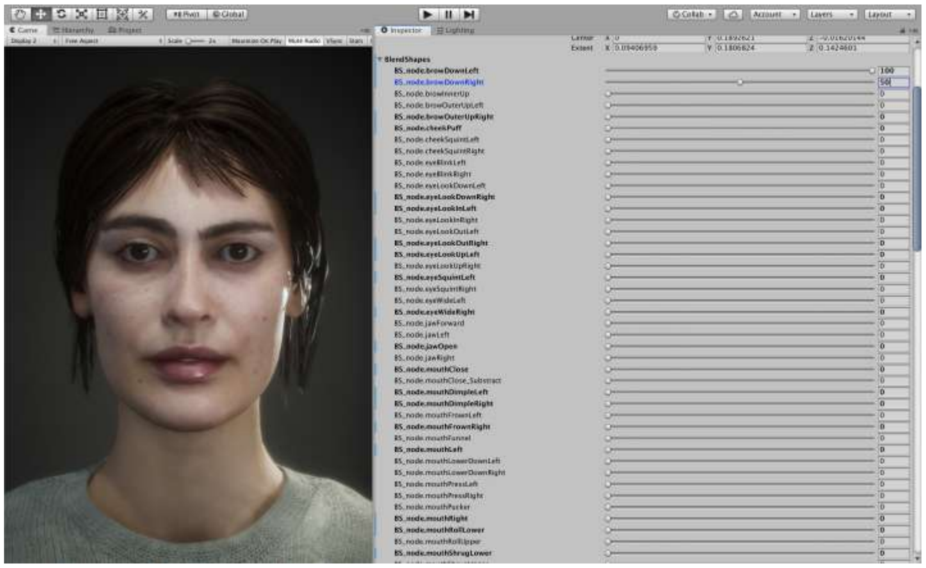

train1

A1 AF01NES.JPG

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	AF01NES.JPG	100	100	30	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0
2	AF02NES.JPG	100	100	30	0	0	0	0	0	5	5	0	0	0	20	23	23	18	18	0	0
3	AF03NES.JPG	0	0	30	0	0	0	0	0	5	5	0	0	0	20	0	0	0	0	25	25
4	AF04NES.JPG	100	100	30	0	0	0	0	0	5	5	0	0	0	20	0	0	0	0	25	25
5	AF05NES.JPG	50	100	30	0	0	0	0	0	5	5	0	0	0	20	0	0	0	0	25	25
6	AF06NES.JPG	50	100	30	0	0	0	0	0	5	5	0	0	0	20	0	0	0	0	25	25
7	AF07NES.JPG	100	100	30	0	0	0	0	0	5	5	0	0	0	20	0	0	0	0	25	25
8	AF08NES.JPG	100	100	30	0	0	0	0	0	5	5	0	0	0	20	0	0	0	0	25	25
9	AF09NES.JPG	100	100	30	25	25	0	0	0	5	5	0	0	0	20	0	0	0	0	25	25
10	AF10NES.JPG	100	100	30	25	25	0	0	0	5	5	0	0	0	20	0	0	25	25	25	25
11	AF11NES.JPG	100	100	30	25	25	0	0	0	5	5	0	0	0	20	0	0	25	25	25	25
12	AF12NES.JPG	100	100	30	25	25	0	0	0	10	10	0	0	0	20	0	0	25	25	55	55
13	AF13NES.JPG	100	100	50	25	25	0	0	0	0	0	0	0	0	20	0	0	25	25	30	30
14	AF14NES.JPG	100	100	65	25	25	0	0	0	0	0	0	0	0	20	0	0	25	25	30	30
15	AF15NES.JPG	0	0	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	35	35
16	AF16NES.JPG	100	100	25	0	0	0	0	0	10	10	0	0	0	20	0	0	35	35	55	55
17	AF17NES.JPG	0	0	0	0	0	0	100	100	0	0	0	0	0	20	0	0	0	0	35	35
18	AF18NES.JPG	50	100	0	20	0	0	100	100	0	0	0	0	0	20	0	0	0	0	35	35
19	AF19NES.JPG	0	0	0	0	0	0	100	100	0	0	0	0	0	20	0	0	35	35	0	0
20	AF20NES.JPG	100	100	0	0	0	0	100	100	8	8	0	0	0	20	0	0	0	0	20	20
21	AF21NES.JPG	0	0	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0
22	AF22NES.JPG	100	100	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0
23	AF23NES.JPG	100	100	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0
24	AF24NES.JPG	0	0	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	30	30
25	AF25NES.JPG	100	80	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	30	30
26	AF26NES.JPG	100	100	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	30	30
27	AF27NES.JPG	100	100	0	0	0	0	100	100	0	0	0	0	0	20	0	0	0	0	25	25
28	AF28NES.JPG	100	100	0	0	0	0	100	100	0	0	0	0	0	20	0	0	0	0	5	5
29	AF29NES.JPG	100	100	0	0	0	0	100	100	0	0	0	0	0	20	0	0	0	0	30	30
30	AF30NES.JPG	100	100	32	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0
31	AF31NES.JPG	100	100	32	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	45	45
32	AF32NES.JPG	100	100	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	30	30
33	AF33NES.JPG	100	100	0	0	18	0	0	0	0	0	0	0	0	20	0	0	0	0	30	30
34	AF34NES.JPG	100	100	0	0	0	0	0	0	10	10	0	0	0	20	0	0	0	0	30	30
35	AF35NES.JPG	100	100	32	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	30	30
36	AM01NES.JPG	100	100	0	0	0	0	100	100	0	0	0	0	0	20	0	0	0	0	15	15
37	AM02NES.JPG	100	100	0	0	0	0	100	100	0	0	0	0	0	20	0	0	15	25	30	30
38	AM03NES.JPG	100	100	0	14	14	0	0	0	0	0	0	0	0	20	0	0	0	0	40	40
39	AM04NES.JPG	100	100	25	0	0	0	100	100	0	0	0	0	0	20	0	0	0	0	25	25
40	AM05NES.JPG	100	100	0	0	0	0	100	100	0	0	0	0	0	20	0	0	10	10	0	0
41	AM06NES.JPG	0	0	0	0	0	0	100	100	0	0	0	0	0	20	0	0	34	34	30	10

Figure 18: Dataset in CSV file

name, the other 74 columns represent the weights of the 74 blendshape in the 3d model. Each row represents an image name and its 74 blendshape weights.



BlendShapes

BS_node.browDownLeft	100
BS_node.browDownRight	50
BS_node.browInnerUp	0
BS_node.browOuterUpLeft	0
BS_node.browOuterUpRight	0
BS_node.cheekPuff	0
BS_node.cheekSquintLeft	0
BS_node.cheekSquintRight	0
BS_node.eyeBlinkLeft	0
BS_node.eyeBlinkRight	0
BS_node.eyeLookDownLeft	0
BS_node.eyeLookDownRight	0
BS_node.eyeLookInLeft	0
BS_node.eyeLookInRight	0
BS_node.eyeLookOutLeft	0
BS_node.eyeLookOutRight	0
BS_node.eyeLookUpLeft	0
BS_node.eyeLookUpRight	0
BS_node.eyeSquintLeft	0
BS_node.eyeSquintRight	0
BS_node.eyeWideLeft	0
BS_node.eyeWideRight	0
BS_node.jawForward	0
BS_node.jawLeft	0
BS_node.jawOpen	0
BS_node.jawRight	0
BS_node.mouthClose	0
BS_node.mouthClose_Substract	0
BS_node.mouthDimpleLeft	0
BS_node.mouthDimpleRight	0
BS_node.mouthFrownLeft	0
BS_node.mouthFrownRight	0
BS_node.mouthFunnel	0
BS_node.mouthLeft	0
BS_node.mouthLowerDownLeft	0
BS_node.mouthLowerDownRight	0
BS_node.mouthPressLeft	0
BS_node.mouthPressRight	0
BS_node.mouthPucker	0
BS_node.mouthRight	0
BS_node.mouthRollLower	0
BS_node.mouthRollUpper	0
BS_node.mouthShrugLower	0

Figure 19: 3d model blendshape from 0 to 100

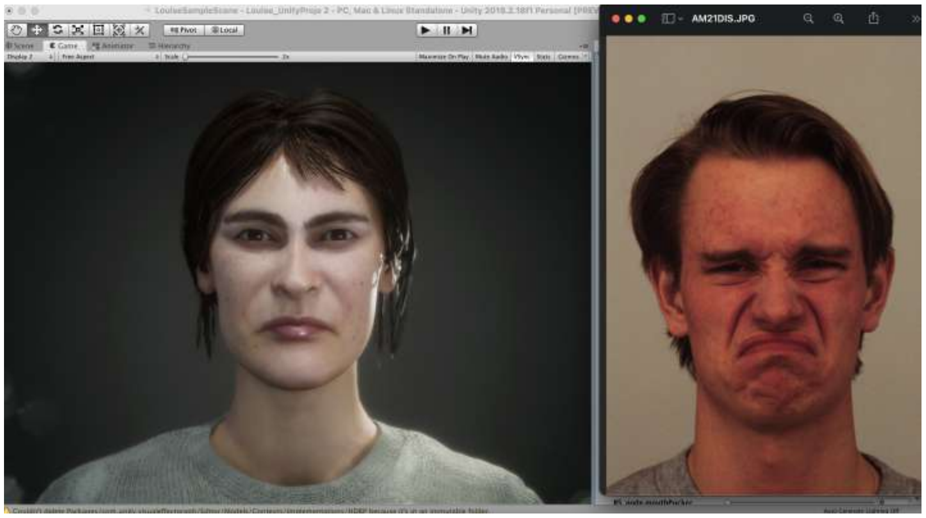


Figure 20: 3d model and human expression

4.2 System Structure

Initially, we had to make a function that takes the input of blendshapes from the CNN results written in C# coding language. Then, we had to build our

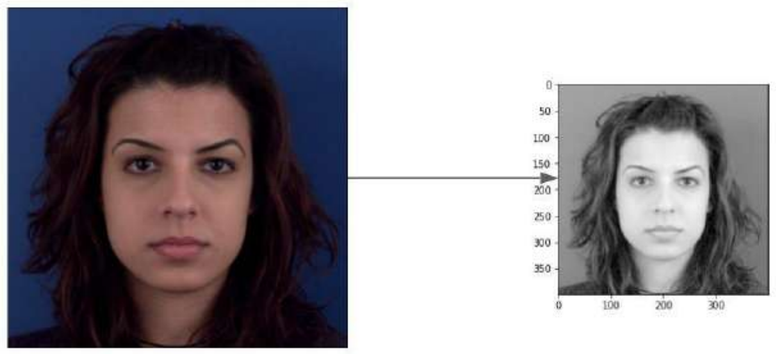


Figure 21: resize and make it gray

dataset by using the 3d model and the images from the internet, In addition, we had to resize the dataset and make it in a gray-scale image to be used by our Convolutional Neural Network (CNN) model to predict the blendshapes of the 3d model. In conclusion, the input is image and the output is belndshape of the image to be used by the 3d model.

4.2.1 System Overview

our system is composed of three essential stages. The first stage is about using a modified function that uses Blendshape class in unity to take the input we got from the CNN model of the blendshapes values in C#. After that we create the dataset by gathering a collection of images and process it. We process the images by making our 3d model mimic the human facial expression in the image. the first phase of dataset processing, is making the 3d model expression natural, the second phase is trying the 74 blendshapes by changing the number of the blendshape from range 0 to 100 to get the best expression of 3d model that match the image, the third phase is saving these blendshapes values into a comma-separated values (CSV) file[Figure 18]. The second stage is to prepare the dataset for Convolutional Neural Network (CNN) model by resizing it and change the RGB color to the gray-scale image. First, we resize the image to 400×400 by using Keras APIs.Second, we change the RGB color to gray-scale color. The third stage is implementing the Convolutional Neural Network (CNN) model to take an input image and output its blendshapes values [figure 29]. Finally, we take the output and put it in the 3d model in unity see [figure 23]. in [figure 22] it show our CNN model architecture that start with

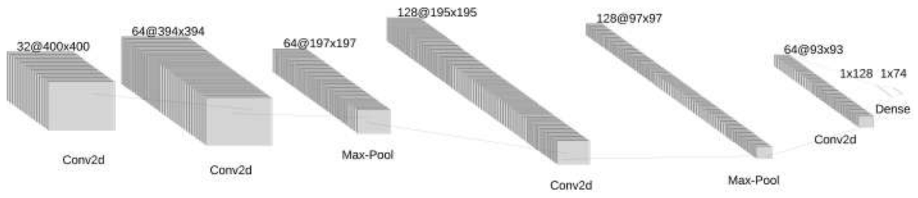


Figure 22: Our CNN model architecture

convolution layer with RELU activation function and 32 filters with kernel size (5,5) and it contain another convolution layer with 64 filters,kernel size (3,3) and with activation RELU and it contains Max-Pooling layer with pool size (2,2) it also has dropout layer 0.25 to avoid overfitting and it had another convolution layer with 128 filters and kernel size (3,3) with RELU activation function and another Max-pooling layer with pool size (2,2) and another convolution layer with 64 filters and karnel size (5,5) with RELU activation function as well as flatten layer to make output 1D and one dense layer with 128 and RELU activation function and the final layer is dense layer with RELU activation function. In [figure 23] our system overview starts with input as an image of human facial expression that we process to be ready to use by our CNN model. Finally, we take the result and run it in 3d model.

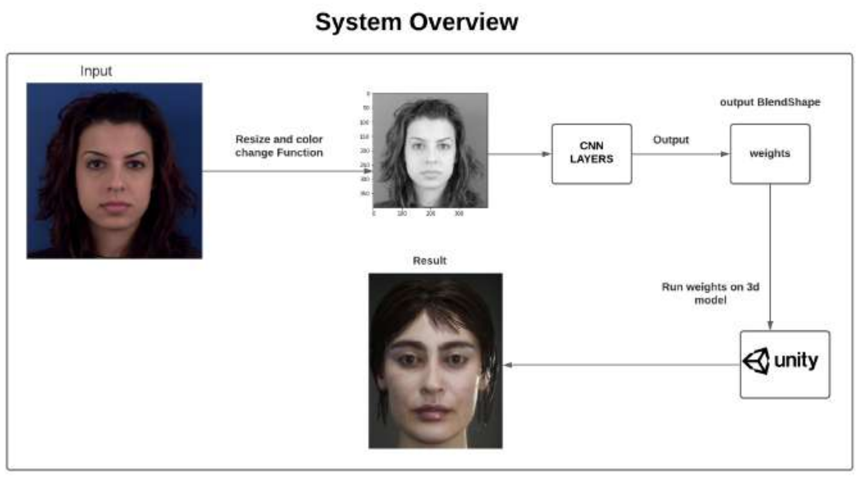


Figure 23: System Overview

4.2.2 Class Diagram or TensorBoard

The following figures illustrate our Convolution Neural Network (CNN) model. It shows that first,it contains convolution layer with RELU activation function and 32 filters with kernel size (5,5) and it contain another convolution layer with 64 filters,kernel size (3,3) and with activation RELU also it contains Max-Pooling layer with pool size (2,2) then it has dropout layer 0.25 to avoid over-fitting and it had another convolution layer with 128 filters and kernel size (3,3) with RELU activation function along with Max-pooling layer with pool size (2,2) and another convolution layer with 64 filters,kernel size (5,5) with RELU activation function and flatten layer to make output 1D and dense layer with 128,RELU activation function Finally the final layer is dense layer with a 74 with RELU activation function [figure 24][figure 25][figure 26].

As we see in [figure 24], first the input shape is (400,400,1), then there are strides and casts after that submission to the next layer. We can see in the figure that we have Mean Squared Error "MSE" loss function and Adam optimizer. In [figure 25] we see the ArgMax then after it the cast, the resize, and another

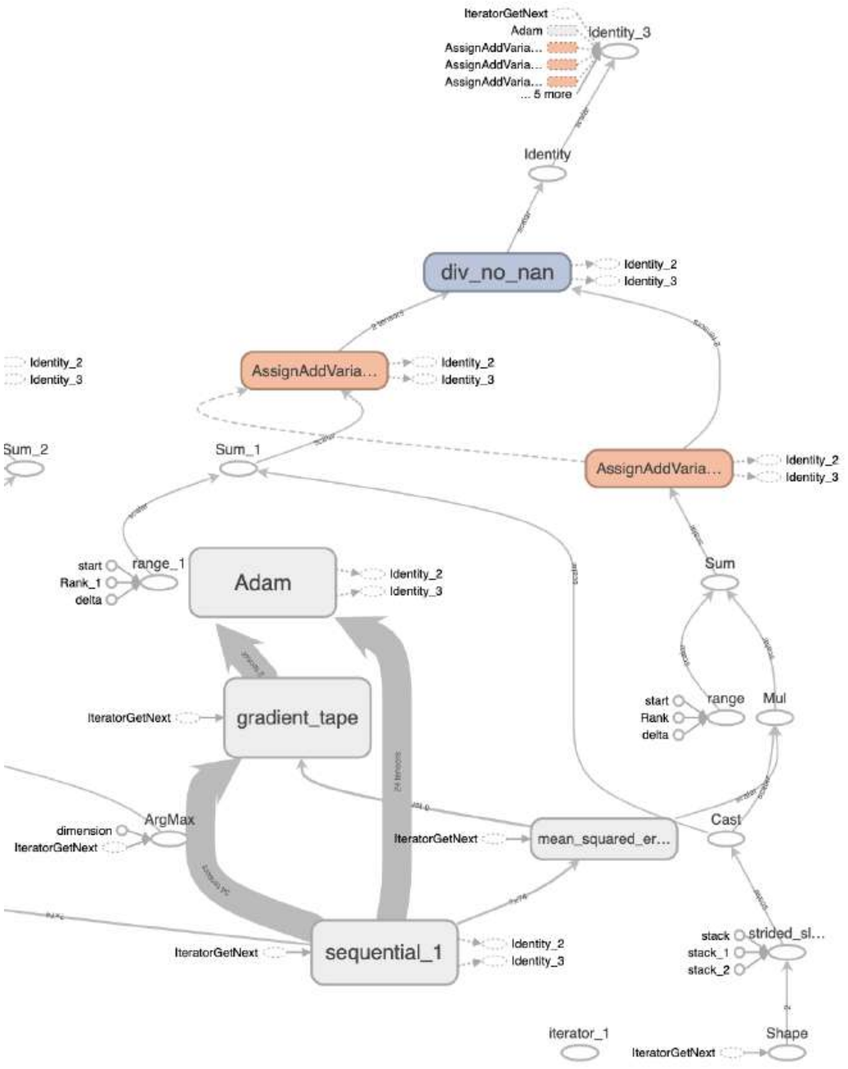


Figure 24: TensorBoard Convolutional layers and Flattens

cast. In [figure 26] we see the layers of our CNN model.

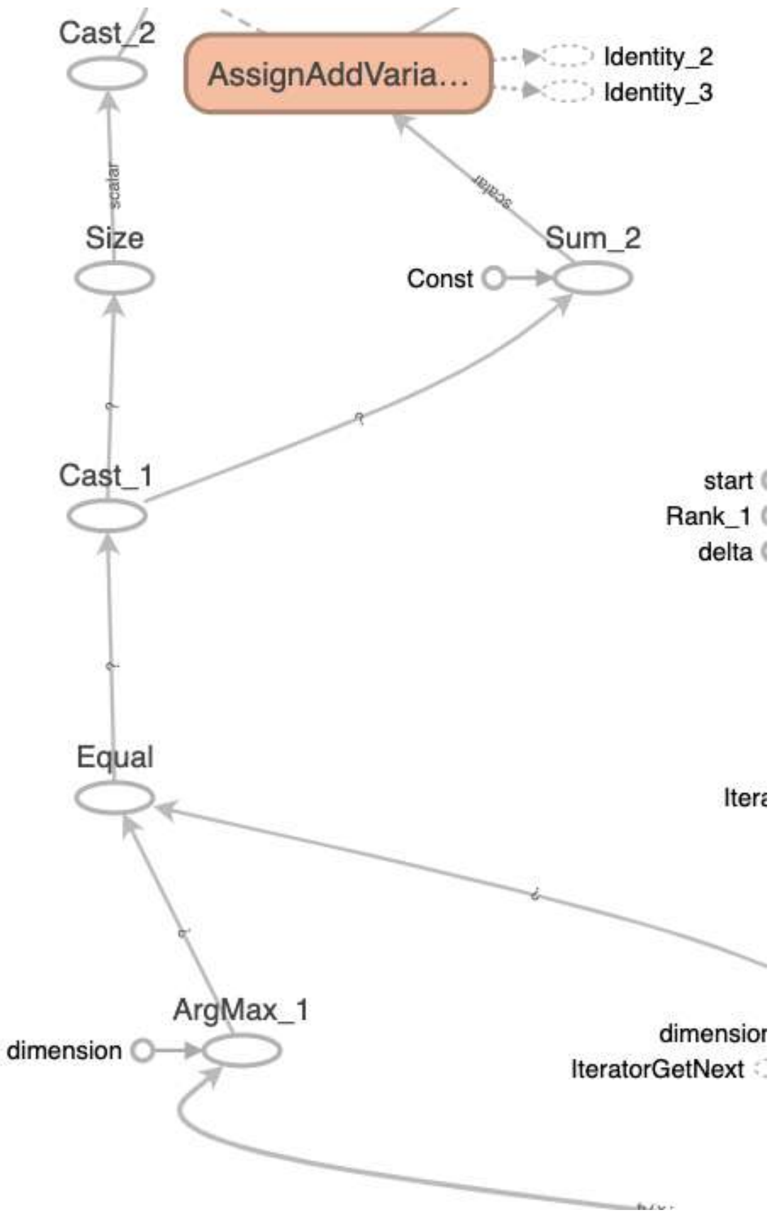


Figure 25: TensorBoard Auxiliary Nodes

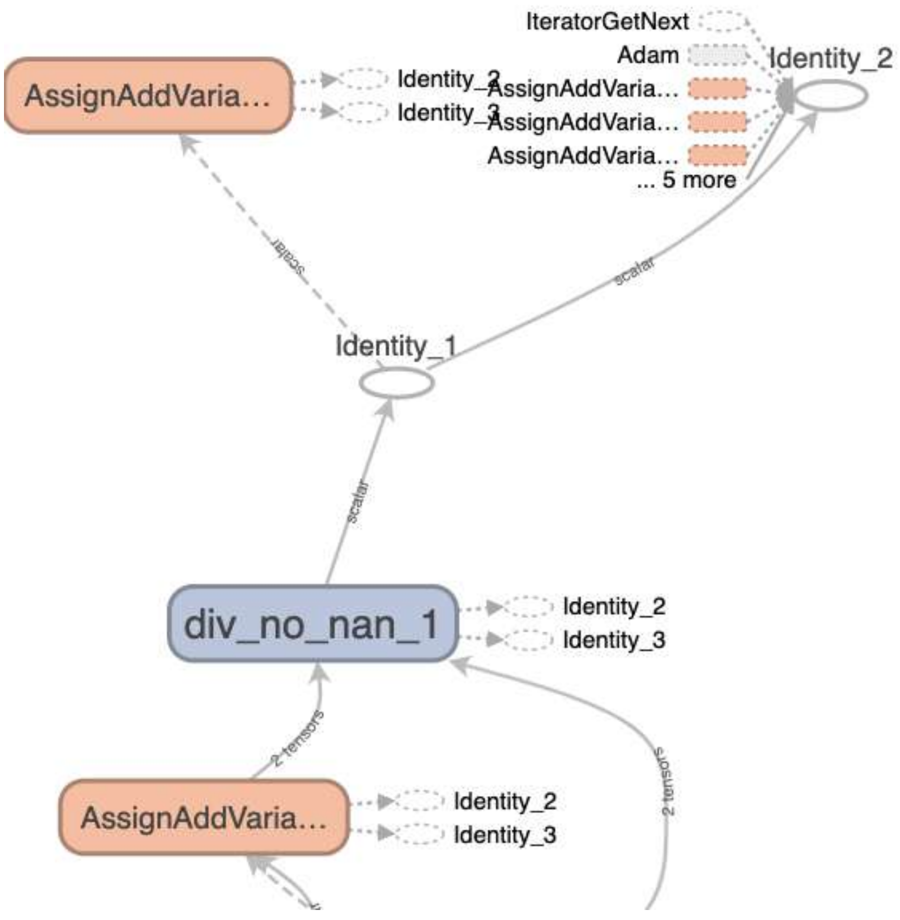


Figure 26: rest of TensorBoard Auxiliary Nodes

4.3 System Running

First we have a function to set and get the values of blend shape of the 3d model. Second, we have a function in our CNN model that takes input a image and the output is 400×400 gray-scale image [figure 28]. Third. we implement our Convolution neural network (CNN) model that have many convolution layer and two max-pool layer and one flatten layer and two dense layer.

4.3.1 Blendshape function

This function takes the value of blendshape as an input. It also insures that the values from range 0 to 100 (if the input is less than 0 we take 0, in case the input is more that 100 we reset it to 100). Finally, our output is the 3d model expression [figure 27].

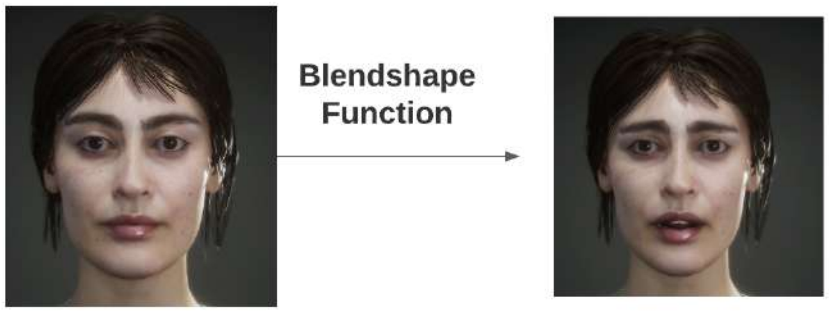


Figure 27: Blendshape function output

4.3.2 Resize and color change function

The input here is the original size of test data and train data all in RGB color and the output is 400×400 gray-scale image [figure 28].

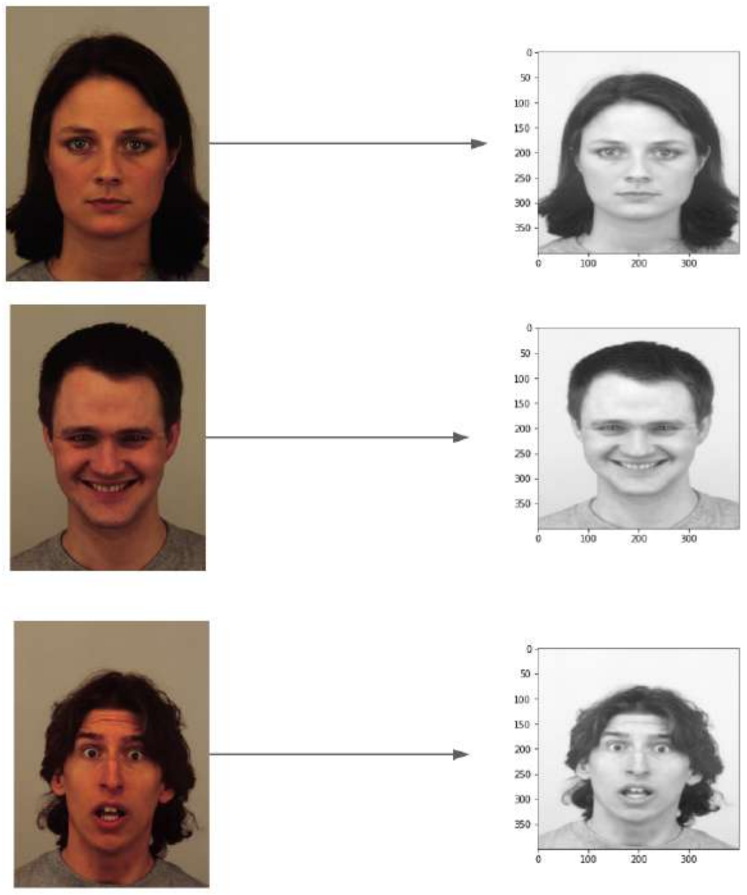
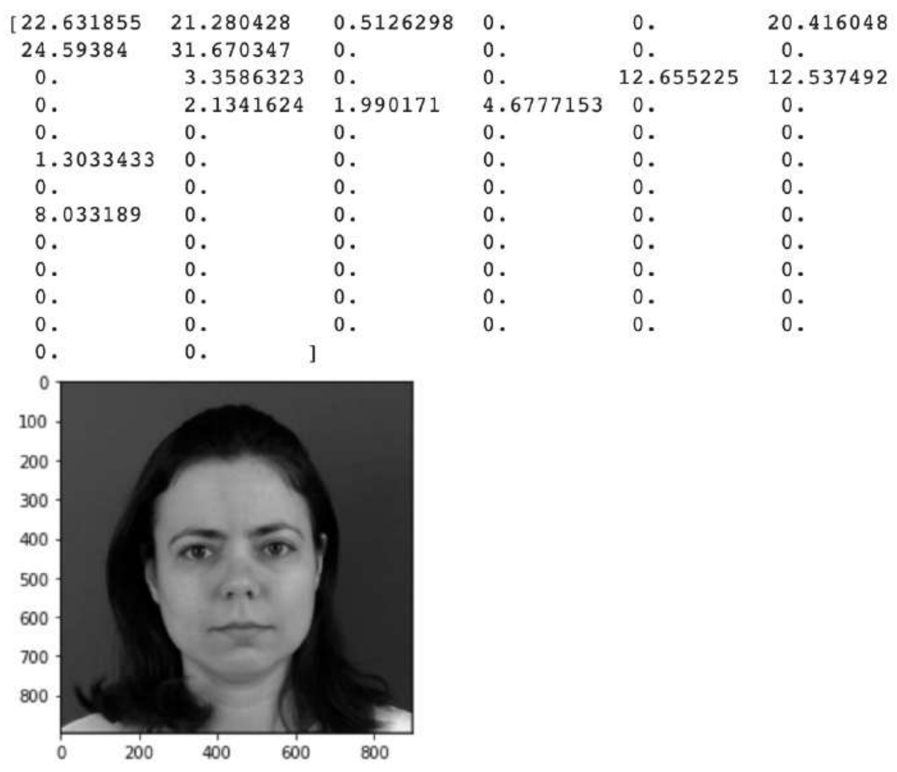


Figure 28: CNN function output

4.3.3 CNN model

Our model contains the input as an image with its original size and color. After we process the image through our function, the model input shape is

(400,400,1) and the output the 74 blendshape [figure 29]. Our model architecture begins with a convolution layer with RELU activation function and 32 filters with kernel size (5,5). It also has another convolution layer with 64 filters, kernel size (3,3) and with activation RELU. along with a Max-Pooling layer with pool size (2,2) also a dropout layer 0.25 to avoid overfitting. And another convolution layer with 128 filters and karnel size (3,3) with RELU activation function and another Max-pooling layer with pool size (2,2) and another convolution layer with 64 filters and karnel size (5,5) with RELU activation function as well as flatten layer to make output 1D and dense layer with 128 and RELU activation function and the final layer is dense layer with 74 with RELU activation function the output of the model in [figure 29]. In [figure 29], we are going



```
[22.631855  21.280428   0.5126298  0.         0.        20.416048
 24.59384   31.670347   0.         0.         0.         0.
  0.         3.3586323  0.         0.        12.655225  12.537492
  0.         2.1341624  1.990171   4.6777153  0.         0.
  0.         0.         0.         0.         0.         0.
  1.3033433  0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  8.033189   0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.        ]
```

Figure 29: CNN output

to illustrate the 74 number in the 3d model in the next list, the 74 blend shape takes number from 0 to 100.

1- it means to move the 3d model brow down in the left by (22.631855).

2- it means to move the 3d model brow down in the right by (21.280428).

3- it means to move the 3d model brow inner up by (0.512698).

4- it means to move the 3d model brow outer up in the left by (0).

5- it means to move the 3d model brow outer up in the right by (0).

6- it means to make the 3d model chick puff by (20.416048).

7- it means to move her chick squint to the left by (24.5938).

8- it means to move her chick squint to the right by (31.670347).

9- it means to move her eye blink in left by (0).

10- it means to move her eye blink in the right by (0).

11- it means to move her eye look down in the left by (0).

12- it means to move her eye look down in the right by (0).

13- it means to move her eye look in left by (3.35873).

14- it means to move her eye look in the right by (0).

15- it means to move her eye lookout in left by (0).

16- it means to move her eye lookout in right by (12.65).

17- it means to move her eye lookup in left by (12.5374).

18- it means to move her eye lookup in right by (0).

19- it means to move her eye squint in the left by (2.134162).

20- it means to move her eye squint in the right by (1.99017).

21- it means to move her eye wide in the left by (4.677153).

22- it means to move her eye wide in the right by (0).

23- it means to move her jaw forward by (0).

24- it means to move her jaw to the left by (0).

25- it means to make her jaw open by (0).

26- it means to move her jaw in the right by (0).

27- it means to close her mouth by (0) the mouth in default is closed.

28- it means to close her mouth subtract by (0).

39- it means to dimple her mouth in the left by (0).

30- it means to dimple her mouth in the right by (1.3033433).

31- it means to frown her mouth in the left by (0).

32- it means to frown her mouth in the right by (0).

33- it means to funnel her mouth by (0).

34- it means to move her mouth to the left by (0).

35- it means to lower her mouth down in left by (0).

36- it means to lower her mouth down in the right by (0).

37- it means to press her mouth in left by (0).

38- it means to press her mouth in the right by (0).

39- it means to pucker her mouth by (0).

40- it means to move her mouth to the right by (0).

41- it means to roll her mouth lower by (0).

42- it means to roll her mouth upper by (8.033189).

43- it means to shrug her mouth lower by (0).

44- it means to shrug her mouth upper by (0).

45- it means to make her mouth smile in the left by (0).

46- it means to make her mouth smile in the right by (0).

47- it means to stretch her mouth to the left by (0).

48- it means to stretch her mouth in the right by (0).

49- it means to move her mouth upper up in the left by (0).

50- it means to move her mouth upper up in the right by (0).

51- it means to make her nose sneer in the left by (0).

52- it means to make her nose sneer in the right by (0).

53- it means to make her tongue out in by (0).

54- it means to make her mouth move like saying the word 'ah' by (0).

55- it means to make her mouth move like saying the word 'aa' by (0).

56- it means to make her mouth move like saying the word 'ao' by (0).

57- it means to make her mouth move like saying the word 'ey' by (0).

58- it means to make her mouth move like saying the word 'er' by (0).

59- it means to make her mouth move like saying the word 'iy' by (0).

60- it means to make her mouth move like saying the word 'uw' by (0).

61- it means to make her mouth move like saying the word 'ow' by (0).

62- it means to make her mouth move like saying the word 'aw' by (0).

63- it means to make her mouth move like saying the word 'oy' by (0).

64- it means to make her mouth move like saying the word ay' by (0).

65- it means to make her mouth move like saying the word 'h' by (0).

66- it means to make her mouth move like saying the word 'r' by (0).

67- it means to make her mouth move like saying the word 'i' by (0).

68- it means to make her mouth move like saying the word 'z' by (0).

69- it means to make her mouth move like saying the word 'ch' by (0).

70- it means to make her mouth move like saying the word 'dh' by (0).

71- it means to make her mouth move like saying the word 'v' by (0).

72- it means to make her mouth move like saying the word 'tn' by (0).

73- it means to make her mouth move like saying the word 'ng' by (0).

74- it means to make her mouth move like saying the word 'pbm' by (0).

5 Chapter 5: Results and Evaluation

5.1 Testing Methodology

There are different methodologies to test our Convolutional Neural Network (CNN) model. We test our CNN by trying the output values in 3d model to see if the facial expression prediction is similar to the facial expression from the input image. First, we had to split our dataset into two subsets, one for to train the data and two to validate the data.We had to feed the data in CNN by assigning batch size value and assigning epochs value. Our evaluation method depends on the best output predictions value that is similar to the desired output and the image. For the sake of our regression CNN model, we do not depend on accuracy percentage, but care for loss values instead.

5.2 Results

5.2.1 Best Results Cases

The best result was generated by the following architecture model.it contains convolution layer with RELU activation function and 32 filters with kernel size (5,5) and it contain another convolution layer with 64 filters,kernel size (3,3) and with activation RELU along with i Max-Pooling layer with pool size (2,2) then it has dropout layer 0.25 to avoid overfitting and it has another convolution layer with 128 filters and karnel size (3,3) with RELU activation function and another Max-pooling layer with pool size (2,2) and another convolution layer with 64 filters and karnel size (5,5) with RELU activation function as well as flatten layer to make output 1D and one dense layer with 128 and RELU activation function and the final layer is dense layer with number of blendshape a 74 with RELU activation function. The model was trained in 300 epochs and 32 batch sizes. The following figure shows the image with predicted model values in 3d model for each expression,First neutral expression [figure 30],second the sad expression[figure 31],third the happy expression[figure 32],fourth the surprised expression[figure 33],fifth the angry expression[figure 34],sixth the afraid expression [figure 35],finaly the disgusted expression[figure 36]. As we see here in [figure 30], the input is in the left and the desired output in the middle, and our CNN model output is on the left. We consider this output (on the right) the best case because the desired output of neutral expression is like our CNN expression output. As we see here in [figure 31], the input is in the left and the desired output in the middle, and our CNN model output is on the left. We consider this output (on the right) the best case because the desired output of a sad expression is like our CNN expression output. As we see here in [figure 32], the input is in the left and the desired output in the middle, and our CNN model output is on the left. We consider this output (on the right) the best case because the desired output of a happy expression is like our CNN expression

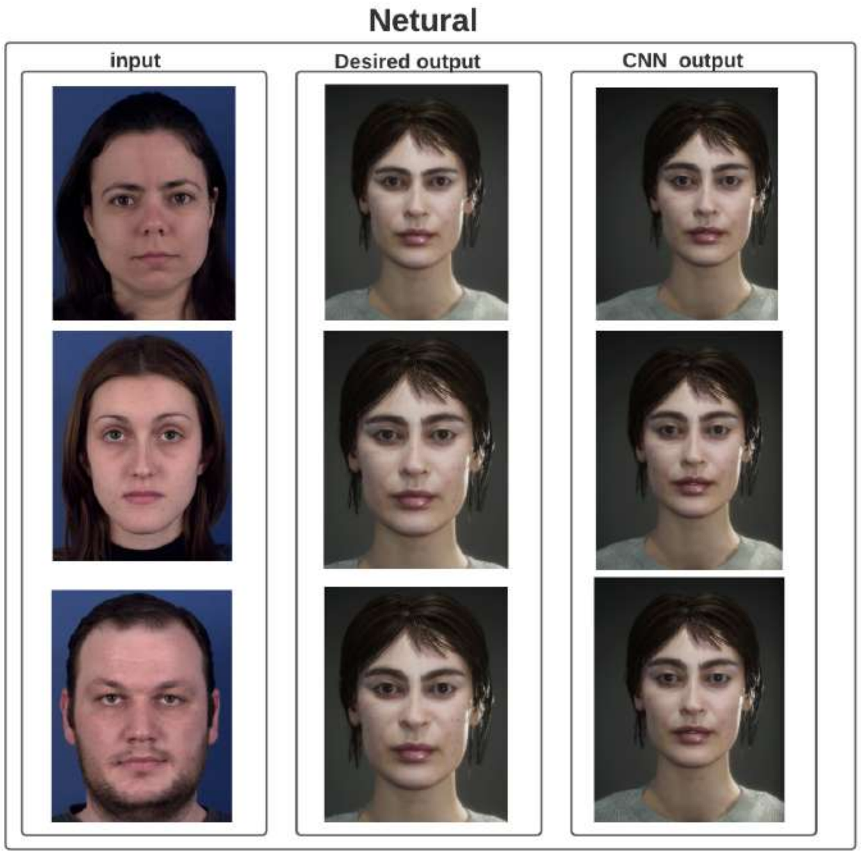


Figure 30: CNN neutral expression output

output. As we see here in [figure 33] the input in the left and the desired output in the middle and our CNN model output on the left. We consider this output (on the right) the best case because the desired output of the surprised expression is like our CNN surprised expression output. As we see here in [figure 34] the input in the left and the desired output in the middle and our CNN model output on the left. We consider this output (on the right) the best case because the desired output of an angry expression is like our CNN expression output. As we see here in [figure 35] the input in the left and the desired output in the middle and our CNN model output on the left. We consider this output (on the

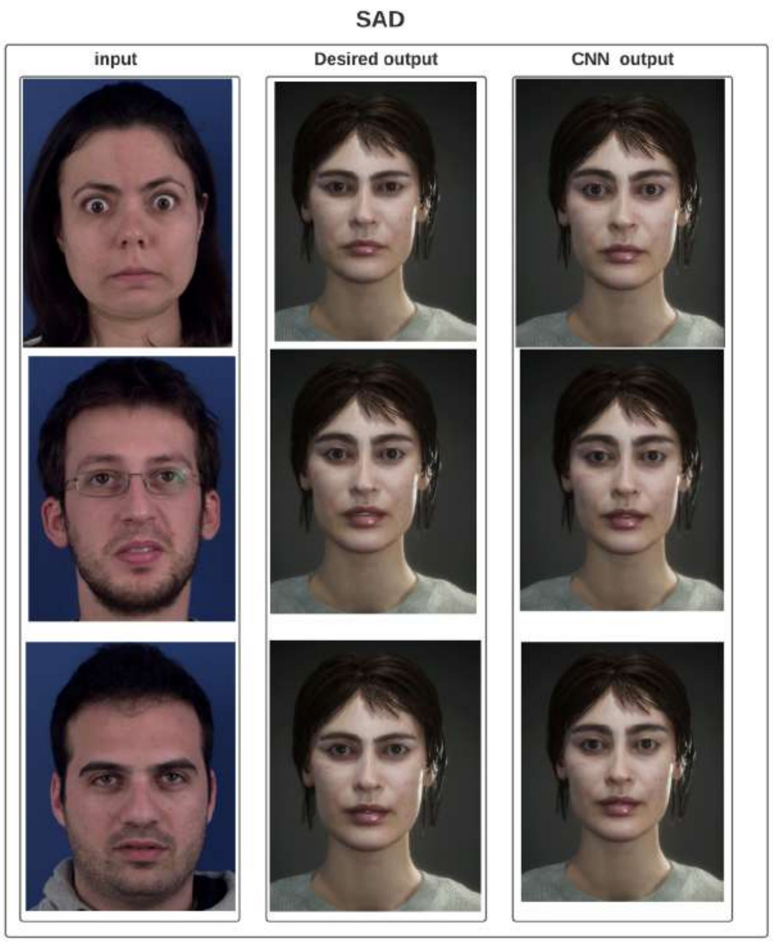


Figure 31: CNN sad expression output

right) the best case because the desired output of the afraid expression is like our CNN expression output. As we see here in [figure 36] the input in the left and the desired output in the middle and our CNN model output on the left. We consider this output (on the right) the best case because the desired output of the disgusted expression is like our CNN expression output.

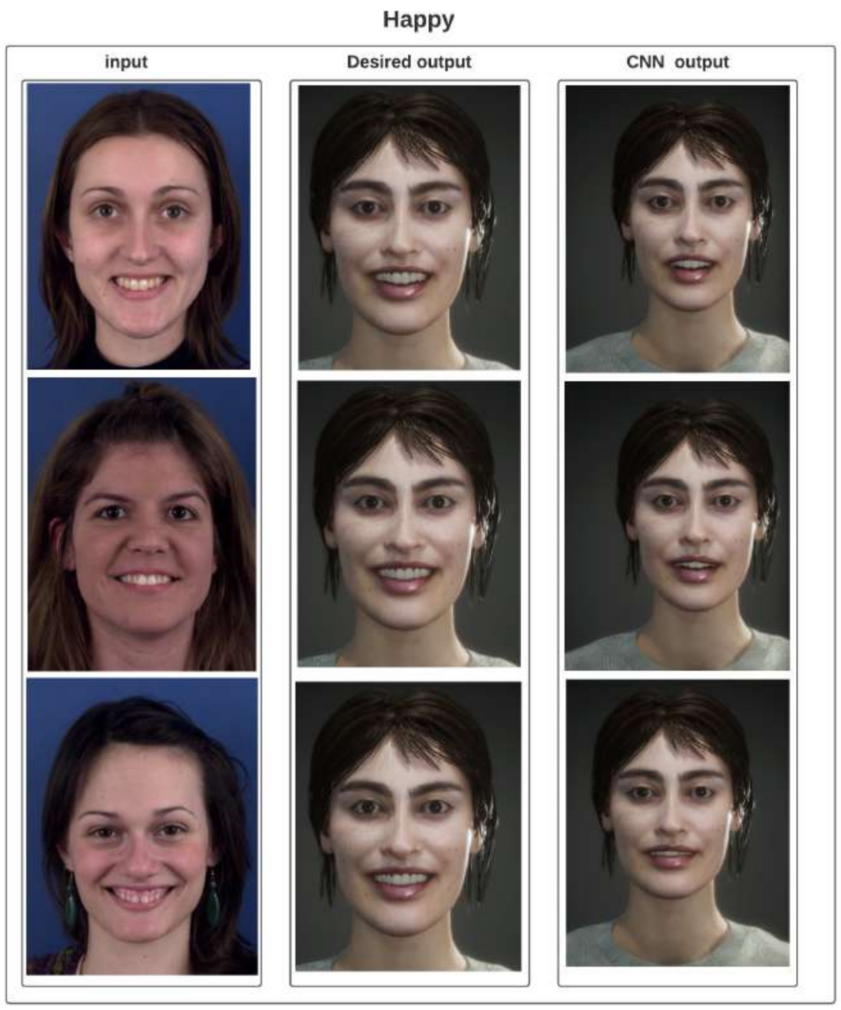


Figure 32: CNN happy expression output

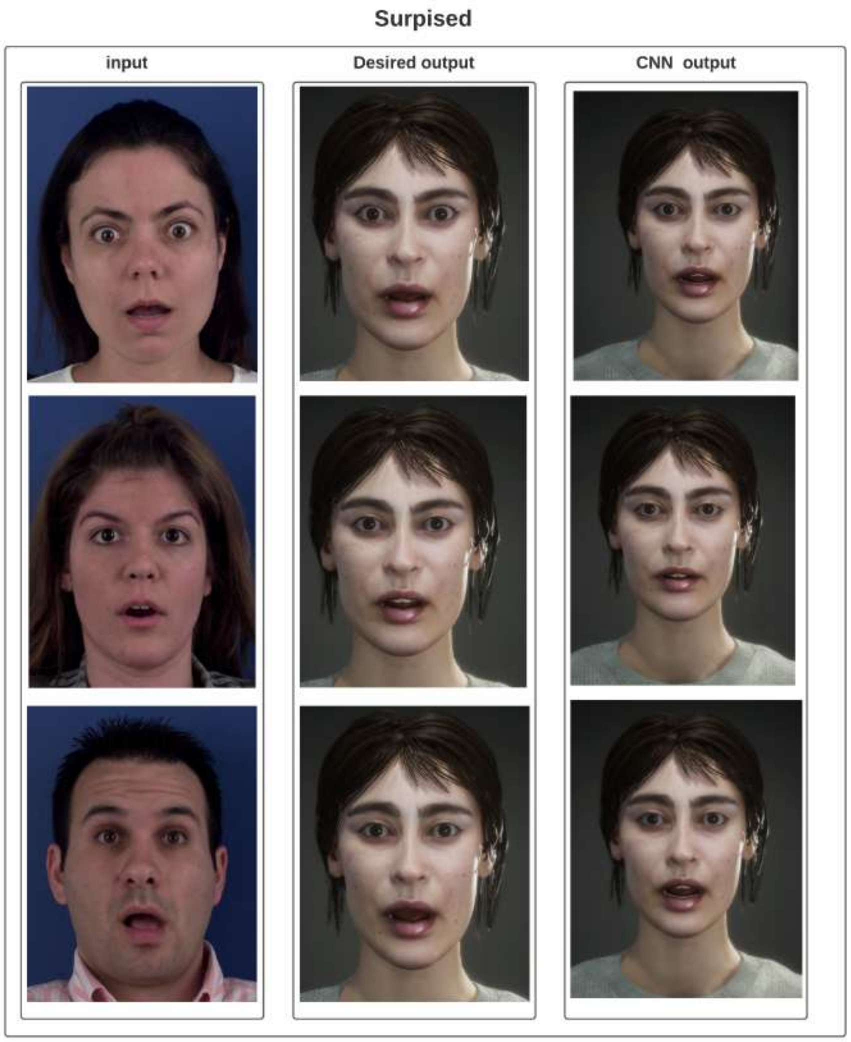


Figure 33: best case CNN surprised expression output

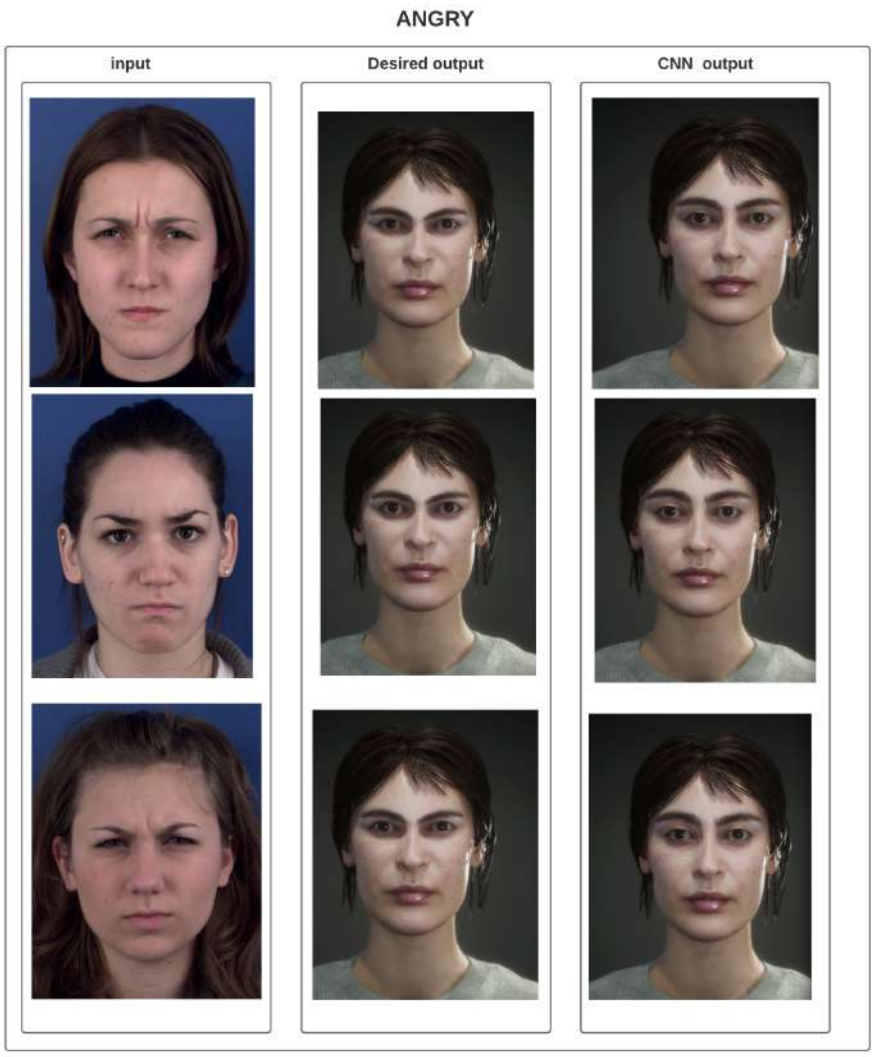


Figure 34: best case CNN angry expression output

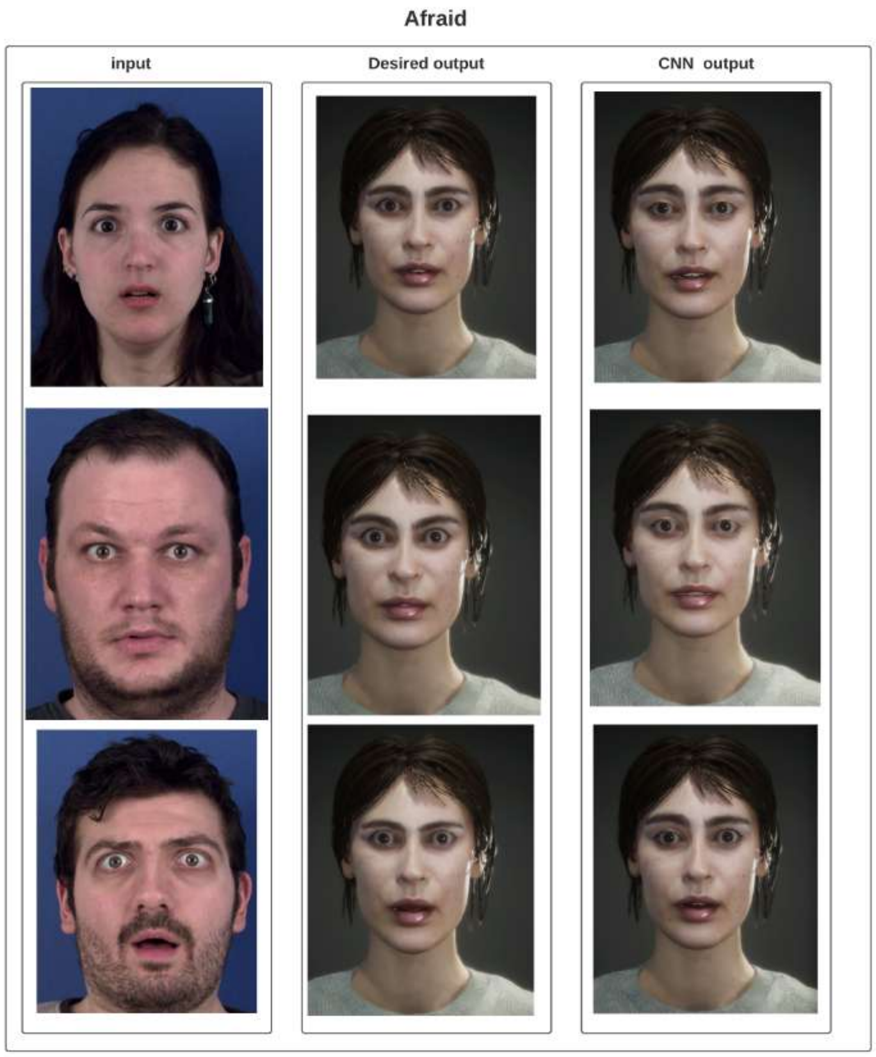


Figure 35: best case CNN afraid expression output

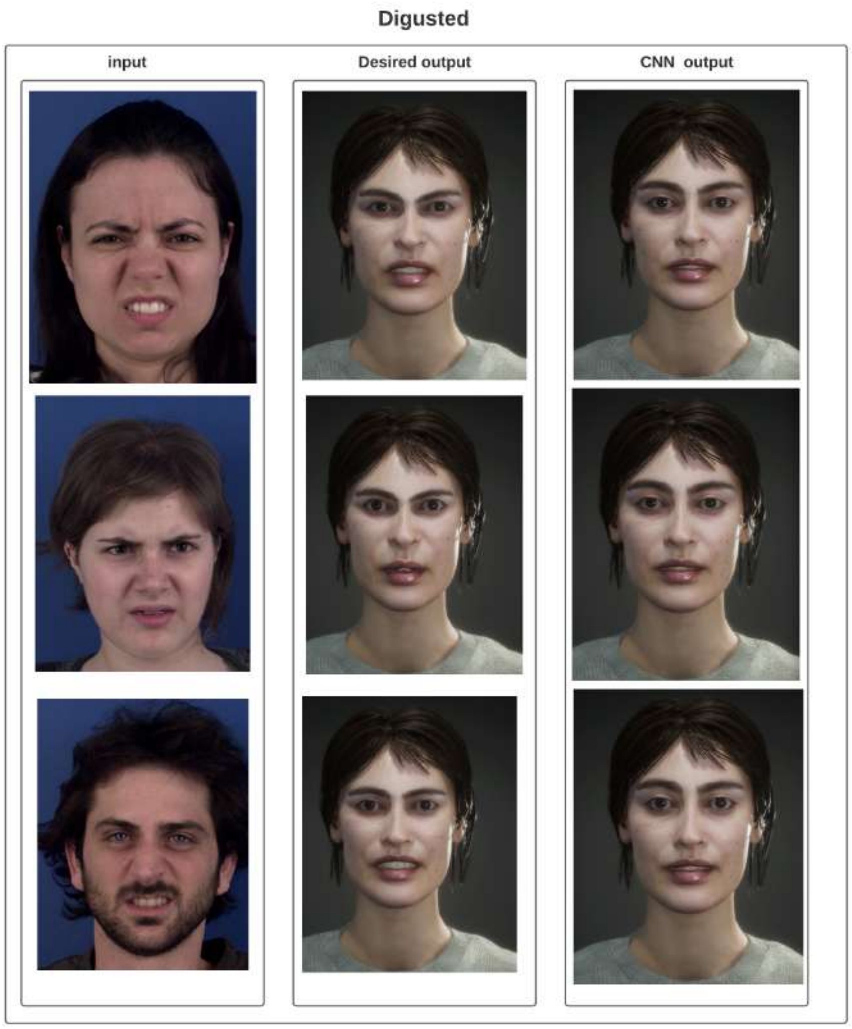


Figure 36: best case CNN disgusted expression output

5.2.2 Acceptable Results Cases

the acceptable result was generated by the following architecture model.it contains convolution layer with RELU activation function and 32 filters with strides (5,5) and it contain another convolution layer with 64 filters,strides (3,3) and with activation RELU and it contains Max-Pooling layer with size (2,2) then it has dropout layer 0.25 to avoid overfitting and it had another convolution layer with 128 filters and karnel size (3,3) with RELU activation function and another Max-pooling layer with pool size (2,2) and another convolution layer with 64 filters and karnel size (5,5) with RELU activation function and flatten layer to make output 1D and one dense layer with 128 and RELU activation function and the final layer is dense layer with number of blendshapes a 74 with RELU activation function. The model was trained in 300 epochs and 32 batch sizes[figure 37][figure 38][figure 39]. As we see in [figure 37] the input in

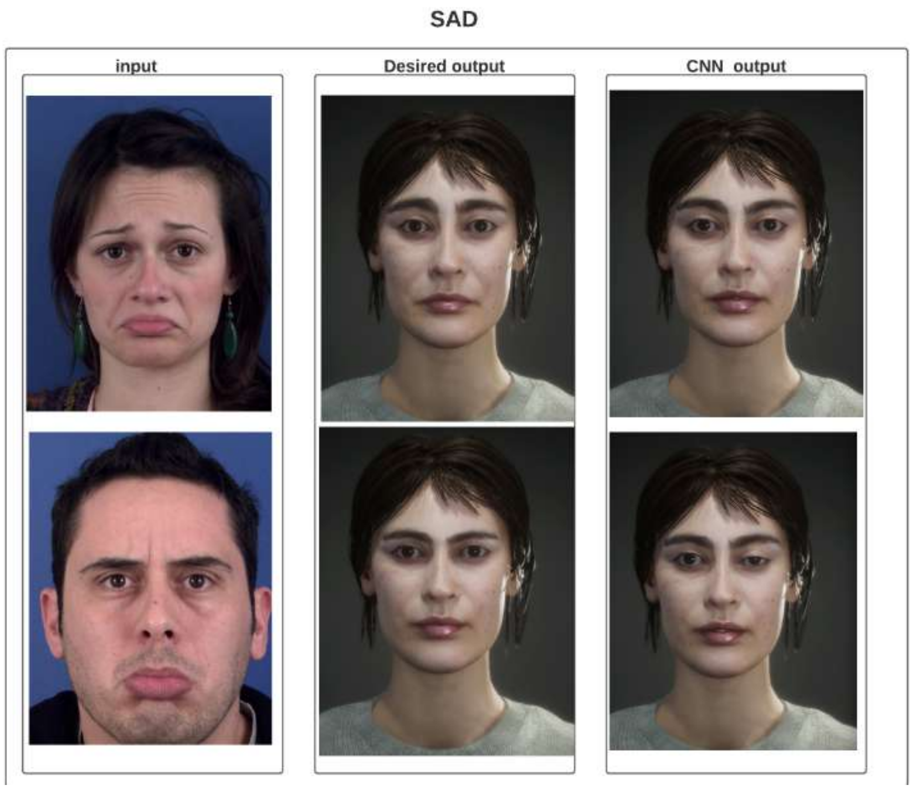


Figure 37: output of acceptable CNN model

the left and the desired output in the middle and our CNN model output on the left. We consider this output an acceptable case because our model had mimic the sad expression with a slight difference. As we see in [figure 38] the input

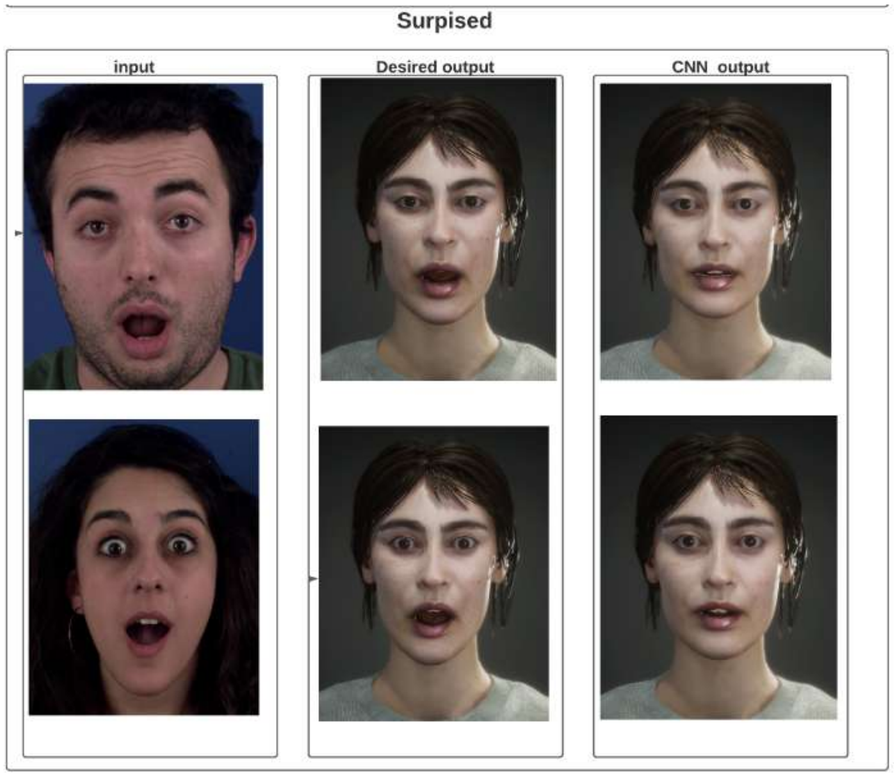


Figure 38: output of acceptable CNN model rest of expression

in the left and the desired output in the middle and our CNN model output on the left. We consider this an acceptable case because our model had mimic the surprised expression with the jaw open but not opened wide as the desired output. . As we see in [figure 39] the input in the left and the desired output in the middle and our CNN model output on the left. We consider this output an acceptable case because our model has mimic the disgusted expression with the mouth open wide, but our CNN output had mimic the expression but without opening the mouth wide. .

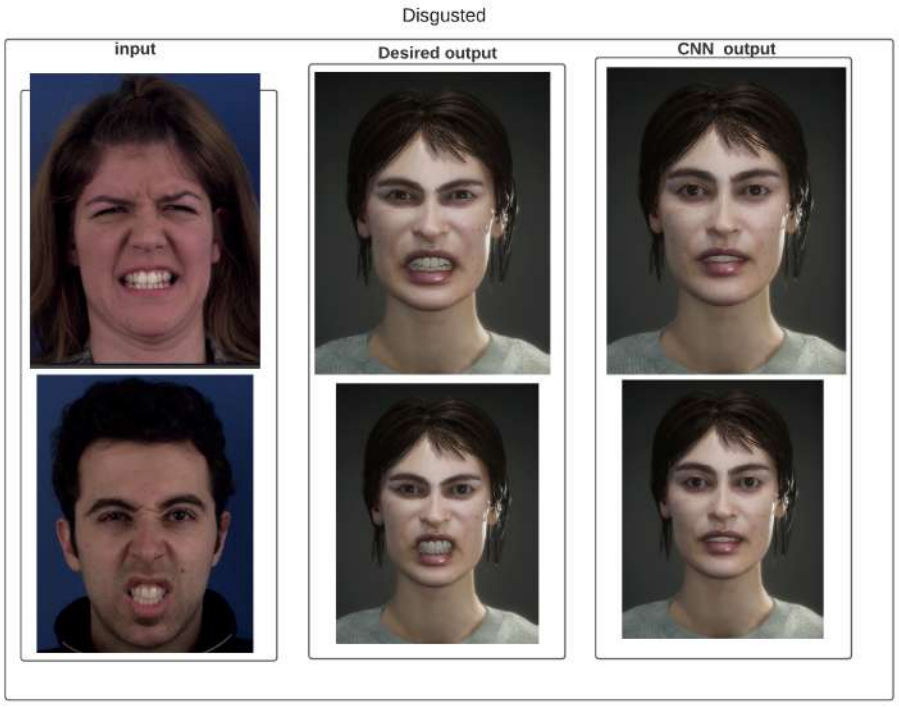


Figure 39: output of acceptable CNN model rest of expression

5.2.3 Worst Results Cases

the worst result was generated by the following architecture model.it begins with a convolution layer with RELU activation function and 32 filters with kernel size (5,5). It also has another convolution layer with 64 filters, kernel size (3,3) and with activation RELU. along with a Max-Pooling layer with pool size (2,2) also a dropout layer 0.25 to avoid overfitting. And another convolution layer with 128 filters and karnel size (3,3) with RELU activation function and another Max-pooling layer with pool size (2,2) and another convolution layer with 64 filters and karnel size (5,5) with RELU activation function as well as flatten layer to make output 1D and dense layer with 128 and RELU activation function and the final layer is dense layer with 74 with RELU activation. The model was trained in 300 epochs and 32 batch sizes .The following figures show the output of this CNN model.[figure 40][figure 41]. As we see here in [figure 40]

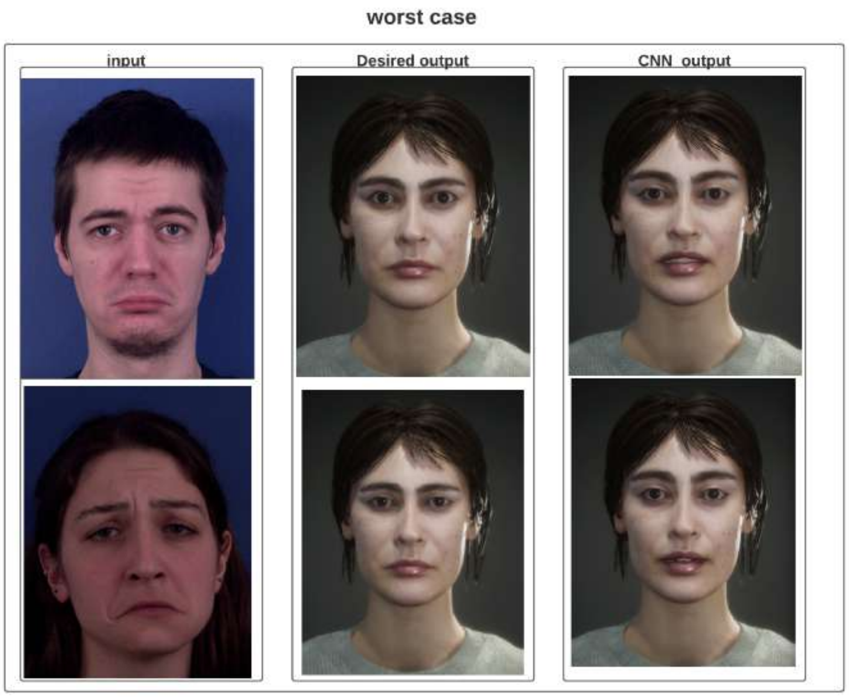


Figure 40: output of worst case CNN model

the input on the left and the desired output in the middle and our CNN model output on the right. We consider this output (on the right) as the worst-case because the input image has a mouth closed and the desired output has a mouth closed, but unfortunately, our CNN model output has a mouth open. As we see here in [figure 41] the input in the left and the desired output in the middle and our CNN model output on the left. We consider this output (on the right) as the worst-case because the input image has a wide smile, the mouth is open and it has a chick squint on the right and on the left, and the desired output has a wide smile, the mouth is open and it has chick squint in the right and in the left but unfortunately our CNN model it had the has only smile but without

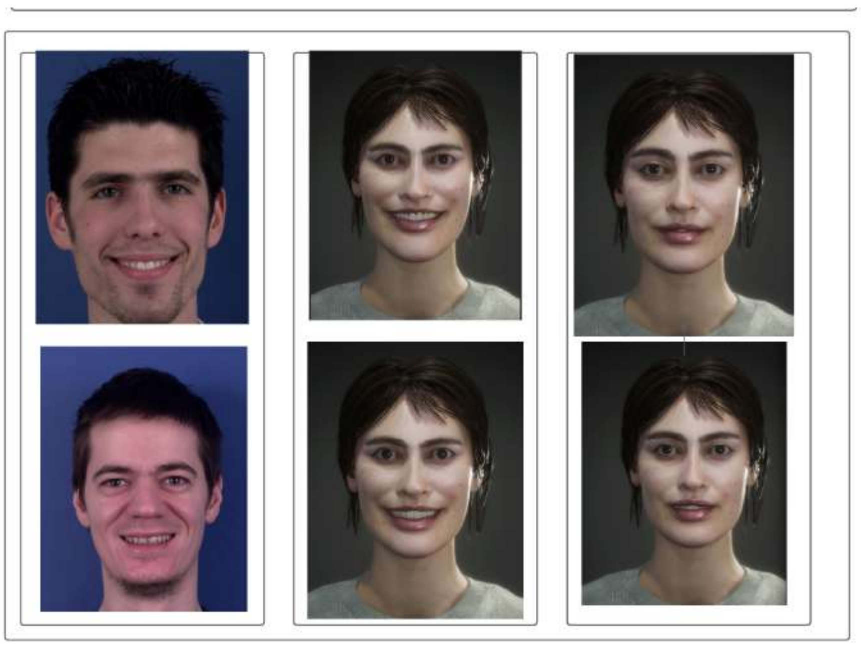


Figure 41: output of worst case CNN output

mouth open that match the desired output.

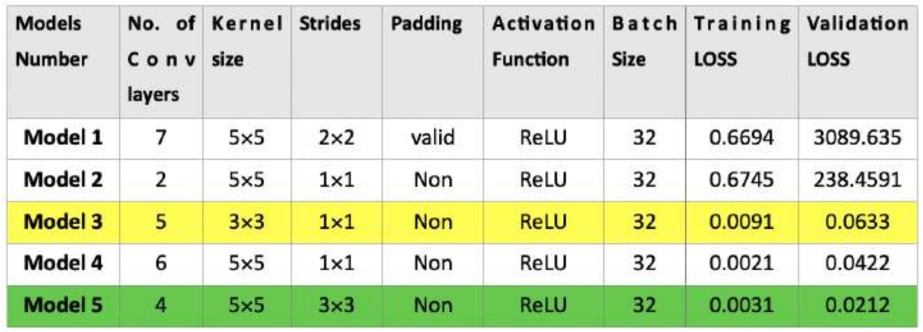
5.2.4 Limitation

In developing our project we faced few limitations. The first limitation was that the prepossessing of the dataset was taking too much time, due the fact that we had to take the image and trying the 74 blendshape from range 0 to 100 in 3d model to get the perfect expression that matches the image expression. As a result of this limitation we could not create a bigger dataset. The second limitation was the human face itself. Humans tend to have different expressions when happy, some would slightly smile. some would open their mouth and close their eyes, and some would smirk. All as an expression of happiness. These many way to express happiness, ended up confusing our machine, which resulted in many worst case output for the happiness expression. Along the fact that the training data for the happy expression were just 70 image. which was not enough.

5.3 Evaluation

5.3.1 Accuracy Evaluation

First of all, we do not depend on accuracy because we have a CNN regression model, so we depend on loss value and validation loss value, and, of course, the output results.[Figure 42] The first model in [figure 42] is Model1 its input



Models Number	No. of Conv layers	Kernel size	Strides	Padding	Activation Function	Batch Size	Training LOSS	Validation LOSS
Model 1	7	5×5	2×2	valid	ReLU	32	0.6694	3089.635
Model 2	2	5×5	1×1	Non	ReLU	32	0.6745	238.4591
Model 3	5	3×3	1×1	Non	ReLU	32	0.0091	0.0633
Model 4	6	5×5	1×1	Non	ReLU	32	0.0021	0.0422
Model 5	4	5×5	3×3	Non	ReLU	32	0.0031	0.0212

Figure 42: summary of regression CNN models Evaluation

shape (400,400,1),The first layer is convolution layer with 64 filter,kernel size (5,5) with strides (2,2) with padding and RELU activation function then it has Max-pooling layer with pool size (2,2) with strides (2.2) also batch normalization layer to normalize the output to the next layer then it has convolution layer with 96 filters , kernel size (1,1) and strides (1,1) with RELU activation function next it has convolution layer with 208 filter and kernel size (3,3) with strides (1,1) with RELU activation function then it has dropout layer with 0.25 as well as max-pooling layer with pool size (3,3) and strides (1,1) then it has convolution layer with 64 filters and kernel size (1,1) and strides (1,1) also dropout layer with 0.25 then it has max-pooling layer with pool size (3,3) and strides (2,2) as well as convolution layer with 96 filter and kernel size (1,1) and strides (1,1) then it has convolution layer with 208 filters and kernel size (3,3) and strides

(1,1) then it has dropout layer with 0.25 then it contain max-pooling layer with pool size (3,3) and strides (1,1) then it has convolution layer with 64 filters and kernel size (1,1) and strides (1,1) then it contains dropout layer with 0.25 then it contain max-pooling layer with pool size (3,3) and strides (1,1) then it has flatten layer and the final layer is dense layer with 74 and RELU activation function. The loss function is mean squared error (MSE), and the optimizer is Adam with batch size 32 and 1000 epochs. The second model in [figure 42] is Model2 it's input shape (400,400,1) the first layer is convolution layer with 32 filters and kernel size is (5,5) with RELU activation function and it contains a max-pooling layer with pool size (2,2) and strides (2,2) and padding is valid then it contains convolution layer with 64 filters and kernel size (5,5) with RELU activation function then it contains batch normalization layer as well as flatten layer the final layer is dense with 74 and activation function RELU. The loss function is "MSE" and Adam optimizer with 32 batch sizes and 300 epochs. The third model in [figure 42] is Model3,First layer is convolution layer with 128 filters and kernel size (3,3) and it contain batch normalization layer then max-pooling layer with pool size (2,2) and strides (2,2) and padding is valid then it contains convolution layer with 64 filters and kernel size (3,3) and RELU activation function then batch normalization layer then as well as max-pooling with pool size (2,2) , strides (2,2) and padding is valid next it has convolution layer with 32 filters and kernel size (3,3) and RELU activation function then batch normalization layer then it contain max-pooling with pool size (2,2) and strides (2,2) then it has convolution layer with 16 filters and kernel size (3,3) then batch normalization layer then it contain max-pooling with pool size (2,2) and strides (2,2) also it has convolution layer with 8 filters and kernel size (3,3) then batch normalization layer then it contains max-pooling with pool size (2,2) and strides (2,2) also it contains flatten layer and the final layer is dense layer with 74 and RELU activation function model was compiled with "MSE" loss function and Adam optimizer with 300 epochs and batch size 32, as we see in [figure 42] the train loss was 0.0091 and the validation loss 0.0633 so we tired the output of this model but Unfortunately the results was a disappointment because it did not achieve what we looking for. The fourth model in [figure 42] is Model4 the first layer is convolution layer with 64 filters and kernel size (5,5) with RELU activation function then it has convolution layer with 64 filters and kernel size (5,5) with RELU activation function also contains batch normalization layer as well as a max-pooling layer with pool size (2,2) then it has convolution layer with 128 filters and kernel size (5,5) and RELU activation function then it has another convolution layer with 128 filters and kernel size (5,5) and RELU activation function then it contains batch normalization layer then max-pooling layer with pool size (2,2) next it contains convolution layer with 256 filters and pool size (3,3) with RELU activation function then it has another convolution layer with 256 filters and pool size (3,3) with RELU activation then it contain batch normalization layer then max-pooling layer with pool size (2,2) then flatten layer and it contains Dense layer with 128 and RELU activation function then it has batch normalization layer as well as dropout layer with 0.2 the final layer is Dense layer with 74 and RELU activation function,the model was compiled with "MSE" loss function and Adam optimizer and it was trained with 1000 epochs and 32 batch size as we see in [figure 42] the train loss was 0.0021 and the validation loss 0.0422 so we tired the output of this model but Unfortunately the results was a disappointment because it did not achieve what we looking for. The fifth model in [Figure 42] is model5 that is our best model that begins with a convolution layer with RELU activation function and 32 filters with kernel size (5,5). It also has another convolution layer with 64 filters, kernel size (3,3), and RELU activation function. Along with a Max-Pooling layer with pool size (2,2), there is also a dropout layer of 0.25 to avoid overfitting. And another convolution layer with 128 filters and kernel size (3,3) with RELU activation function and another Max-pooling layer with pool size (2,2) and another convolution layer with 64 filters and kernel size (5,5) with RELU activation function as well as flatten layer to make output 1D and dense layer with 128 and RELU activation function and the final layer is a dense layer with 74 with RELU activation function, the model was compiled with Mean squared error (MSE) loss function and Adam optimizer and the model was trained with

300 epochs and 32 batch size and we tried the output result it had achieved the desired output the training loss is 0.0031 and validation loss is 0.0212[Figure 43]. In [figure 43] the graph has an blue color that represents the training loss and

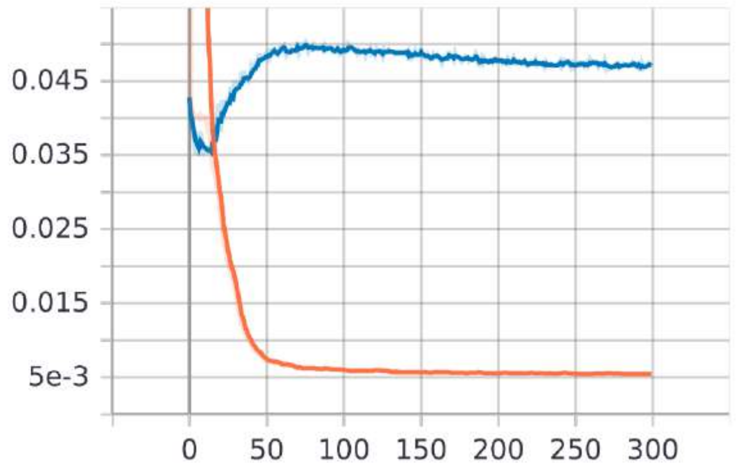


Figure 43: best model train and validation loss

the orange color represents the validation loss through the 300 epochs. In [figure 44] there are different optimizes we had tried in our CNN best model. As we see in [figure 44], the first graph represents the Adam optimizer, the training loss is 0.0031 and the validation loss 0.0212 in 300 epochs, so we tried the output in a 3d model and it did achieve the best result. Then the second graph represents the Adamax optimizer the training loss is 0.0039 and validation loss is 0.0714 in 300 epochs so we tried the output result, unfortunately, the best result was in one expression so we ignore it, the Third graph represents the Nadam optimizer the training loss is 0.0458 and validation loss is 0.0455 so we tried the output result but Unfortunately, the output result did not achieve what we looking for.

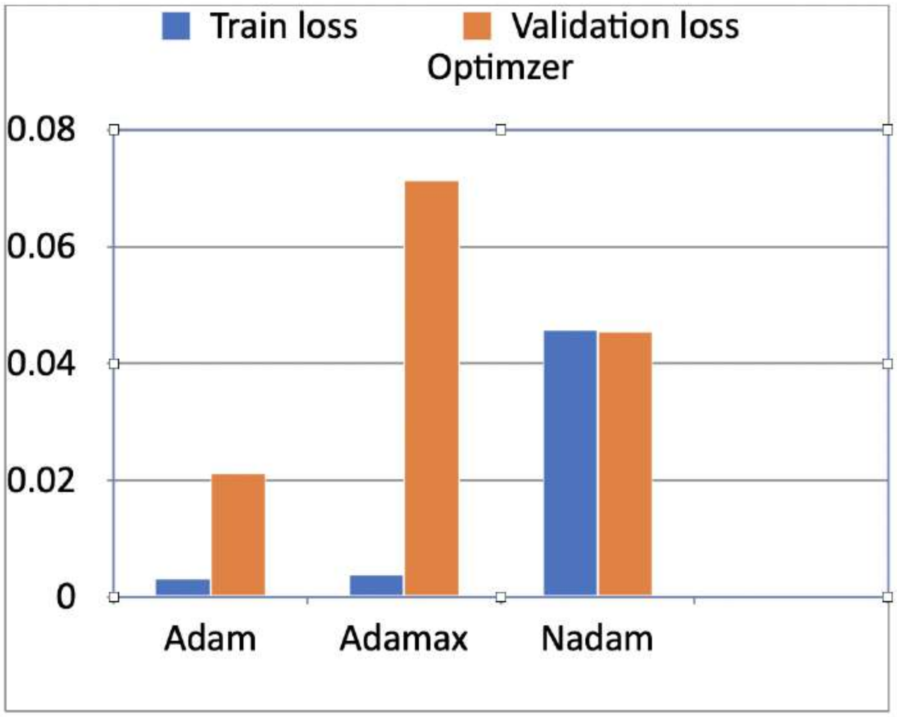


Figure 44: the different optimizes we had tried in our CNN best model

5.3.2 Time Performance

Using google colab with Keras APIs to run our model, gave us a huge advantage as in saving the computer resources and saving time rather than using TensorFlow and Spyder environments on the local PC CPU. When comparing google colab with our local personal computer, we find that google colab takes a 125ms per step and in local pc it takes 5 minutes per step because we do not have a GPU.

6 Chapter 6: Conclusion and Future Work

6.1 Conclusion

In conclusion, we developed a system for human facial expression augmentation using projection mapping. After searching many approaches and algorithms that were widely used in projection mapping we decided on using deep learning approach to mimic the human face expression. We estimate the blend shape values of the input expression by using a Convolution Neural Network (CNN) algorithm. We take a human facial expression image as an input and the model output the blend shapes of human facial expression (the input image) then we take these blend shape values and run it into the 3d model to projects the mask with the expression of human face. First, we had to find a way that make us change the blend shapes values of the 3d model by using C# code in unity. The way we settled on using was a blend shape class that we used as a function to make us read and write the values of 74 blend shape of the 3d model. Second, we had to build a dataset for the CNN model by taking a image of human facial expression and process it to find the 74 blend shape that very similar to this expression. Third, it was the implementation stage, we had to implementing the CNN model that can predict the 74 blend shape weights of 3d model from an image. We tested the system with the test data with different model architecture. The best results came from the model with four convolution layer with two max-pooling layer and two dense layer with RELU activation function and adam optimzer. The results from this model were promising, we think that to increase the accuracy of the model it will need more dataset size to get enough training. Our system has many advantages and could be used in theaters,movies industry. Another advantage is that our system can project the mask faster than deforming it from scratch.

6.2 Problem Issues

6.2.1 Technical issues

The system consumes a huge amount of time and a costing amount of computational power in the local computer CPU processor. One of our limitations was the fact that we did not have a GPU so we had to run the system in a cloud services like google colab, and we could not run the 3d model in best performance because we did not have enough local RAM

6.2.2 Scientific issues

we could not run the system with large batch size for example 100 because we have a large CNN model architecture. Also we did not use a RNN model because the CNN can recognize human expression image, when we used a Linear activation function it did not give us a promising results so we had to use RELU activation function.

6.3 Future Work

As a future work, we plan on making a deeper pre-trained models with larger and better architecture using a data-set size that is bigger, which will result in better accuracy. In the future as well, we might implement facial recognition through video so we could read the facial expression directly from our end user instead of an image.

7 References

1-Aliaga, D., Law, A. and Yeung, Y., 2008. A virtual stage for real-world objects. ACM Transactions on Graphics , 27(5), pp.1-10.
2-Bermano, A., Billeter, M., Iwai, D. and Grundhöfer, A., 2017. Makeup Lamps: Live Augmentation of Human Faces via Projection. Computer Graphics Forum , 36(2), pp.311-323.
3-N. Hashimoto, R. Koizumi and D. Kobayashi, "Dynamic Projection Mapping with a Single IR Camera", International Journal of Computer Games Technology, vol. 2017, pp. 1-10, 2017. Available: 10.1155/2017/4936285.
4—Jones, B., Sodhi, R., Murdock, M., Mehra, R., Benko, H., Wilson, A., ... Shapira, L. (2014, October). RoomAlive: magical experiences enabled by scalable, adaptive projector-camera units. In Proceedings of the 27th annual ACM symposium on User interface software and technology (pp. 637-644).
5- Siegl, C., Lange, V., Stamminger, M., Bauer, F. and Thies, J., 2017. Face-Forge: Markerless Non-Rigid Face Multi-Projection Mapping. IEEE Transactions on Visualization and Computer Graphics , 23(11), pp.2440-2446.