Facial Expression Transfer with Machine Learning

20 January, 2019

Supervisor: Dr. Dirk Schnieders
CHOI Wai Yiu (3035240568)
CHENG Tak Kin (3035240116)
Abstract

This project aims to study the usage of deep learning and Facial Action Coding System (FACS) [1] in facial expression transfer. In this project, facial expression transfer will be achieved using the technology of facial expression extraction and facial expression generation under a FACS conditioned scheme which improves the complexity and anatomical accuracy of the output. A software will be developed using a Generative Adversarial Network (GAN) model [2] based on Action Units (AUs) annotations to achieve photorealistic facial expression synthesis. The software will be built on top of the GANimation model proposed by Pumarola et al. [3] and the software will be further combined with the OpenFace 2.0 toolkit [4] to achieve facial expression transfer.

Acknowledgment

We would like to express our deep gratitude to our supervisor, Dr. Dirk Schnieders. We had changed our project and supervisor once in the middle of September. We would like to thank Dr. Dirk Schnieders for his patient and guidance for this project in such a hurry situation.
# TABLE OF CONTENTS

**LIST OF FIGURES** .................................................................................. 2  
**LIST OF TABLES** ............................................................................. 3  
**ABBREVIATIONS** ........................................................................... 3  

1. INTRODUCTION ............................................................................. 4  
  1.1 OBJECTIVE ............................................................................. 4  
  1.2 MOTIVATION ........................................................................... 4  
  1.3 SCOPE OF WORK ...................................................................... 5  

2. RELATED WORK ............................................................................ 5  
  2.1 FACIAL ACTION CODING SYSTEM (FACS) ............................. 5  
  2.2 GENERATIVE ADVERSARIAL NETWORKS (GANs) ................. 5  
  2.3 FACIAL EXPRESSION SYNTHESIS BASED ON FACS ......... 6  
    2.3.1 2D Action Unit Representation .................................. 6  
    2.3.2 3D Action Unit Representation .................................. 6  
  2.4 FACIAL ACTION UNIT INTENSITY ESTIMATION .............. 6  

3. METHODOLOGY ............................................................................ 7  
  3.1 TRAINING OF THE GANIMATION MODAL ......................... 7  
    3.1.1 Datasets ......................................................................... 7  
    3.1.2 Data Preprocessing ..................................................... 8  
    3.1.3 Training ......................................................................... 9  
  3.2 SOFTWARE IMPLEMENTATION ........................................... 10  
    3.2.1 Client Implementation ................................................ 10  
    3.2.2 Server implementation ............................................. 11  

4. RESULTS AND LIMITATIONS ..................................................... 12  
  4.1 GANIMATION MODEL TRAINING RESULTS AND APPLICATION .................................................................................. 12  
  4.2 LIMITATIONS ........................................................................... 15  

5. FUTURE WORK ............................................................................. 16  
  5.1 IMPROVEMENT ON TRAINING DATA .................................. 16  
  5.2 REWRITE THE GANIMATION MODAL ................................ 16  
  5.3 IMPLEMENT OPENFACE 2.0 ................................................. 16  

6. CONCLUSION .............................................................................. 17  

7. SCHEDULE ................................................................................... 17  

8. REFERENCES .................................................................................. 18
List of Figures

Figure 1. haar features for object detection .......................................................... 8
Figure 2. An example of face detection using haar cascade ........................................ 8
Figure 3. Sample response body from the server ...................................................... 10
Figure 4. Flow of the server handling a request ....................................................... 11
Figure 5. Graphs of loss value of the generator ...................................................... 13
Figure 6. Graphs of loss value of the discriminator ................................................ 14
Figure 7. An example of applying GANimation model in facial expression transfer in which the facial expression of Donald Trump is transferred onto the face of Barack Obama .......................................................... 14
Figure 8. Sample results of altering AU17, AU23 and AU45 ..................................... 15
Figure 9(a). The distribution of number of presences of each AU in the training dataset. .............................................................................................................. 15
Figure 9(b). The average AU values in the training dataset ........................................ 16
List of Tables

Table 1. Built in haar cascade files of OpenCV ................................................................. 9
Table 2. Specification of the virtual machine we used .......................................................... 9

Abbreviations

AU Action Unit
CDAAE Difference Adversarial Autoencoder
CelebA CelebFaces Attributes Dataset
CK database Cohn-Kanade (CK and CK+) database
FACS Facial Action Coding System
FERET Facial Recognition Technology program
GAN Generative adversarial network
GUFD Glasgow Unfamiliar Face Database
LFW Labeled Faces in the Wild database
1. Introduction

Synthesizing realistic facial expressions from 2D images of human face could be beneficial to a wide range of industries including the film industry and the social media industry. Automatic facial expression generation technique could also help the development of other computer vision research areas such as face recognition and facial expression prediction.

Before the success of deep learning, researchers tried to produce new facial expressions by manipulating the facial landmarks with a geometric approach such as triangular geometric deformation [5] and frequency analysis [6], but the results were not satisfactory. However, facial expression generation technology has been substantially improved along with the maturity of deep learning.

In this Introduction section, the objective of this project will first be introduced. After that, motivation of this project will be discussed. Finally, the scope of this project will be described to state the focus of the project.

1.1 Objective

The main objective of this project is to achieve accurate and complex facial expression transfer anatomically by facial expression extraction and facial expression generation under an AU conditioned deep learning scheme. As the technology of facial expression extraction is mature, facial expression generation is the focus of this project.

1.2 Motivation

In 2014, Goodfellow et al. introduced generative adversarial networks (GANs) which are powerful for generative tasks [2]. Since then, many research studies in using various types of GANs for facial expression generation have been published such as CycleGAN [7], IcGAN [8] and StarGAN [9]. StarGAN made a huge breakthrough in facial expression generation. The synthesized facial expression results have become more realistic and natural. However, StarGAN is trained on the Radboud Faces Database (RaFD) [10] which consists of facial images with binary facial expression labels such as happy, sad, angry, fearful, etc. Due to the property of StarGAN, the generation results are limited to the corresponding set of discrete facial expression labels and interpolation between different facial expression is not application [2]. Therefore, StarGAN is not suitable for complex facial expression generation. Although ExprGAN [11] enables the intensity adjustment of a given set of facial expressions, the synthesized results are still limited to a number of discrete facial expressions.

Because of the above limitation mentioned, the Facial Action Coding System (FACS) [1] has started to gain attention in the area of facial expression synthesis. FACS is an anatomical facial expression measurement system which describes facial expressions using Action Units (AUs) which are independent actions of different sets of facial muscle [1]. The combination of deep learning and FACS makes complex facial expression generation possible. Research studies using GAN, autoencoder with FACS has successfully synthesized complex and realistic facial expressions [3] [12] [13]. Deep learning models conditioned by AU intensity can be applied to achieve facial expression transfer by combining with AU intensity estimation.

By combining the facial expression generation program GANimation [3] and the existing solution of facial expressions detection, the OpenFace 2.0 toolkit [4], facial expression transfer can be performed. Although the source code of the GANimation program and the OpenFace 2.0 toolkit is published on the internet, there is no open source software of facial expression
transfer available. Therefore, this project aims to build an open source facial expression transfer software based on the GANimation program and the OpenFace 2.0 toolkit.

1.3 Scope of work

There are three stages for the deliverables. For each of the stages, different types of service deployment are used. For the internal testing phase, a simplest deployment method is used in which the software just pipes the output of OpenFace to GANimation. The computer running the program will need to install the dependencies, which are other programs or libraries required to execute the program.

However, this project aims to build an easy to use application for general public, which means requiring installation of a large amount of libraries in the client computer is not preferable. Thus, the second type of service deployment is a network application. We will host a server running the program of the software developed in the internal testing phase, such that client can send the data through internet requests to the server and get the processed image from the server response. This approach perverts the dependencies problem of the first service deployment method, users are not required to install any libraries. However, it is limited by network communication. If the real-time facial expression transfer is implemented, the software can consume a huge amount of network traffic as it needs to upload and download videos at the same time.

The last type of service deployment is a local interpreter. The existence of the large amount of dependencies is caused by some functionalities not used in this project. For example, GANimation needs libraries to train the model, only the trained model is necessary for providing service. Therefore, the training result can be extracted and be provided to the users independently to minimize the cost of the client machine. We will implement a mobile version if possible.

Although GANimation is powerful, it takes time to run due to its large model size. Thus, a real time function of facial expression transfer for video is not achieved yet. It will be our extra goal to build the application in real time approach.

2. Related Work

2.1 Facial Action Coding System (FACS)

FACS is an anatomical system which is developed by Ekman and Friesen in for facial expression measurement [1]. In this system, facial expressions are divided into several independent sets of muscle movement. The concept of Action Units (AUs) is introduced to represent the correlation between each independent region of a facial expression and the involved facial muscle. For example, AU1 represent the movement of “Inner Brow Raiser” which involves the frontalis and pars medialis muscle.

2.2 Generative Adversarial Networks (GANs)

In 2014, Goodfellow et al. proposed GANs which are a type of unsupervised deep neural network model structured based on a zero-sum game framework [2]. A classic GAN is composed of a generative network and a discriminative network which are trained simultaneously. The generative network is trained to create realistic sample while the discriminative network is trained to determine whether a sample is produced by the generative
network or from a real dataset. GANs are widely used in generative tasks including facial expression synthesis.

2.3 Facial Expression Synthesis based on FACS

2.3.1 2D Action Unit Representation

Zhou and Shi proposed a conditional difference adversarial autoencoder (CDAAE) for photorealistic facial expression synthesis based on FACS [12]. The CDAAE is able to generate a new facial expression based on an unseen input facial image which does not exist in the training dataset while preserving the facial identity. This is achieved by adding a low-level feedforward connection between the encoder and decoder to disambiguate identity changes and facial expression changes [12].

Pumarola et al. introduced a novel GAN scheme conditioned by AU annotation for facial expression generation [3]. The model produces a mapping from a single facial image and an AU intensity vector to a new image of the same facial identity under the desired facial expression. Different from the approach suggested by Zhou and Shi [12], this model achieved an unpaired image to image translation. Instead of pairs of images of the same person under different facial expressions, only images with AU annotations are required for model training, which makes the model more general and flexible. Moreover, Pumarola et al. made the network more robust to background and lighting condition changes by adding an attention layer which limits the network to only manipulate regions of images that are related to producing the new facial expressions [3].

2.3.2 3D Action Unit Representation

Instead of directly generating 2D images, Liu et al. suggested an approach combining 3D Morphable Model (3DMM) [14] with GANs [13]. 3DMM facial expression parameters are first extracted from the input images. The parameters are then used to generate AUs-conditioned 3DMMs. Finally, the output images are rendered from the 3DMMs. Under this approach, the model is able to generate high-resolution facial images.

2.4 Facial Action Unit Intensity Estimation

Baltrušaitis et al. described a method to detect AU occurrence and intensity in real time based on facial geometric features [4]. The problem of individual difference was addressed by using Support Vector Machines (SVM) and Support Vector Regression (SVR) under a person-specific normalization approach based on cross-dataset learning. OpenFace is a popular open source toolkit for facial behavior analysis, including facial action unit detection based on the research of Baltrušaitis et al. It is able to extract 18 kinds of facial action units in 5 discrete levels of intensity.
3. Methodology

This section will explain the methodology used to produce the open source facial expression transfer software of this project.

3.1 Training of the GANimation modal

3.1.1 Datasets

A large amount of data is required for training the GANimation model. There are several datasets available on the Internet. Initially, it was decided that the EmotioNet database and CK+ database will be used in the current stage. Nevertheless, the EmotioNet database is only available for research laboratories and permission of using the database was not granted for this project. Therefore, some alternative databases are used. The following is a list of alternative databases used in this project and description of the databases.

3.1.1.1 CK+ database

Cohn-Kanade AU-Coded Expression Database is a facial expression database prepared by Kanade, Cohn, & Tian [15] and Lucey et al. [16]. It includes 486 sequences of actions from 97 actors and AUs are marked on peak expression. However, due to historical reasons, most of the images are in monochrome, and cannot be used in the training of GANimation. After filtering, it turns out that number of color images in CK database is under 1000.

3.1.1.2 GUFD

Glasgow Unfamiliar Face Database was a database prepared for Glasgow Face Matching Test (GFMT) [17]. It contains about 6000 images in total, from 303 identities and 20 images for each.

3.1.1.3 Color FERET Database

Color FERET Database is part of the Facial Recognition Technology (FERET) program [18], which aims to develop new technology, for the automatic recognition of human faces. It contains 14,126 facial images of 1199 individuals.

3.1.1.4 LFW

Labeled Faces in the Wild [19] [20] contains more than 13,000 images of faces collected from the Internet by detecting with the traditional Viola-Jones algorithm.

3.1.1.5 CelebA

CelebFaces Attributes Dataset [21] is the largest dataset that we can access. It contains 202,599 face images from 10,177 identities obtained from the Internet.
3.1.2 Data Preprocessing

To be used to train GANimation, the images should be in size 128*128 pixels, and their AU indensities must be labeled by OpenFace.

3.1.2.1 Crop and resize the images and face detection

To crop the training images properly, face detection is applied during the process. The default Haar feature-based cascade classifiers included in OpenCV is used. It is an algorithm using haar features to detect objects in a monochrome image. As shown in Figure 1, there are 3 kinds of haar features, edge features, line features and four-rectangle features. They represent different pattern of white and black area of the image.

![Haar Features Diagram](image1.jpg)

**Figure 1. haar features for object detection**

Here is an example of how a cascade classifier detect faces. In the image in Figure 2, there is a edge feature between the eyes and the cheeks, and a line feature within the area of the eyes and the bridge of the nose. If these kinds of features located in a proper position and with a proper size, there is most likely a face in that area.

![Face Detection Example](image2.jpg)

**Figure 2. An example of face detection using haar cascade**

However, the algorithm is not perfect and missed detection may occurs. Therefore, OpenCV provided different kind of cascade classifiers for us to choose. Table 1 lists out all the haar cascade files provided by OpenCV and the highlighted items are for human face detection.
Table 1. Built in haar cascade files of OpenCV

<table>
<thead>
<tr>
<th>Built in haar cascades file</th>
<th>Target to detect</th>
</tr>
</thead>
<tbody>
<tr>
<td>haarcascade_eye.xml</td>
<td>eye</td>
</tr>
<tr>
<td>haarcascade_eye_tree_eyeglasses.xml</td>
<td>glasses</td>
</tr>
<tr>
<td>haarcascade_frontalcatface.xml</td>
<td>cat face</td>
</tr>
<tr>
<td>haarcascade_frontalcatface_extended.xml</td>
<td>cat face</td>
</tr>
<tr>
<td>haarcascade_frontalface_alt.xml</td>
<td>human face</td>
</tr>
<tr>
<td>haarcascade_frontalface_alt2.xml</td>
<td>human face</td>
</tr>
<tr>
<td>haarcascade_frontalface_alt_tree.xml</td>
<td>human face</td>
</tr>
<tr>
<td>haarcascade_frontalface_default.xml</td>
<td>human face</td>
</tr>
<tr>
<td>haarcascade_fullbody.xml</td>
<td>full human body</td>
</tr>
<tr>
<td>haarcascade_lefteye_2splits.xml</td>
<td>left eye</td>
</tr>
<tr>
<td>haarcascade_licence_plate_rus_16stages.xml</td>
<td>Russian number plate</td>
</tr>
<tr>
<td>haarcascade_lowerbody.xml</td>
<td>lower human body</td>
</tr>
<tr>
<td>haarcascade_profileface.xml</td>
<td>profile face</td>
</tr>
<tr>
<td>haarcascade_righteye_2splits.xml</td>
<td>right eye</td>
</tr>
<tr>
<td>haarcascade_russian_plate_number.xml</td>
<td>Russian number</td>
</tr>
<tr>
<td>haarcascade_smile.xml</td>
<td>smile</td>
</tr>
<tr>
<td>haarcascade_upperbody.xml</td>
<td>upper human body</td>
</tr>
</tbody>
</table>

3.1.2.2 AU intensities detection

As suggested by GANimation, OpenFace 2.0 is used to detect the AU intensities of face images. Additionally, OpenFace 2.0 won’t give any output if it cannot recognize it is a face image. In this case, we would try to use another haar cascades file to detect face from the original image, which can minimize the chance of miss detection.

3.1.3 Training

After following the instruction of GANimation to pack the training data into a .pkl file, the training of the GANimation modal can be performed.

GANimation used CUDA support of PyTorch. Thus, a machine with suitable GPU is required to train the model. In order to minimize the cost of this project, virtual machine rental service from cloud computing platform is used. Among Microsoft Azure, Amazon Web Service (AWS) and Google Cloud Platform (GCP), GCP is chosen to host the project server as it is the only platform providing a GPU instance on free trial. Table 2 shows the specification of the virtual machine.

Table 2. Specification of the virtual machine we used

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>8 virtual CPUs</td>
</tr>
<tr>
<td>Memory</td>
<td>30 GB</td>
</tr>
<tr>
<td>GPU</td>
<td>1 x NVIDIA Tesla K80</td>
</tr>
</tbody>
</table>
3.2 Software Implementation

A server-client approach is used in the deliverable. In this way, the setting and installed library on the virtual machine can be reused, and the specification requirement of the client side can be minimized.

3.2.1 Client Implementation

Python program and web application were two implementation method considered during the design of the client. Web application was chosen for the purpose of easy deployment. Plain HTML and Javascript is used and the deliverable is hosted on the FYP account.

At first, it was planned to crop the face image with OpenCV.js on the frontend before sending the request to minimize the request size. However, OpenFace was not installed in the client and was not able to be used to audit the result as in data processing stage. Therefore, the quality of face detection would be poor. Thus, the client was designed to send two raw images to the server.

The response from the server consists of 4 integers representing the position of the face (x, y, width, height), and a base 64 string of the result image data in 128*128 as shown in Figure X. According to this information, only the area of the face will be updated with the new expression.

Figure 3. Sample response body from the server
3.2.2 Server implementation

NodeJS is used to manage internet requests, and Python to provide the actual service. The Python script is modified from the testing script of GANimation, it can detect faces and crop images as mentioned in 3.2.1, and also use the GANimation modal to perform facial expression modification. Since it takes times for the Python program to load the GANimation modal, the Python program is pre-started, and the modal is preloaded such that the response time can be minimized. It communicates with the NodeJS program using socket.

Then, AU indesities detection is required from another image. To achieve it, the program will start a subprocess of OpenFace, and check for its output.

The actual flow of the server handling requests described in Figure 4. The NodeJS server and the Python program will keep running and waiting for requests. The OpenFace program will only be fired whenever it is needed.

![Figure 4. Flow of the server handling a request](image-url)
4. Results and Limitations

In this section, the results of the GANimation training process as well as the application of the model in facial expression transfer will first be described. After that, limitations of the results will be discussed.

4.1 GANimation Model Training Results and Application

Figure 5 and 6 are scalar visualization produced by tensorboard. Figure 5 and 6 show training results of the generator and discriminator of the GANimation model respectively. There were 6,000,000 training steps taken during the process. The whole training process took more than 224 hours in total.

From Figure 5(a), a decrement in loss value of AU values for the generator can be observed which indicates an improvement of the performance of the generator during the training process. Figure 5(b) and 5(c) show the quality of the attention masks produced by the generator during the training, for both of the real image existing in the training dataset and fake image generated during the training. Although subtle, a stable decrement of loss value can be observed for both of the real image and fake image attention masks. All of the above observation suggests that during the training process, the performance of the generator improves.

From Figure 6(a), similar to the situation of the generator, a decrement in loss value of AU value discrimination can be observed which is an evidence of performance improvement of the discriminator. Figure 6(b) and 6(c) show the performance of the discriminator in fake image discrimination and real image discrimination respectively. Figure 6(b) and 6(c) are highly related to Figure 5(d). While the loss value of real image discrimination conducted by the discriminator and the loss value of fake image generation conducted by the generator are directly proportional with each other, both of them are in inverse proportion with the loss value of fake image discrimination of the discriminator.

From Figure 5(d), it can be observed that there was a decrement of loss value followed by an gradual increment, and an decrement at the end of the training. This fluctuation of loss value during training can be explained by the game theory applied in GAN. As the generator and the discriminator was completing with each other in a zero-sum manner during training, the increment of loss value of fake image generation was caused by the improvement of the discriminator. Despite there was a fluctuation of the performance of fake image generation of the generator, both of the performance of the generator and the discriminator was gradually improving, and most importantly, the performance of the generator surpassed the performance of the discriminator at the end of the training. Therefore, it is reasonable to conclude that this training process was successful. Figure 7 shows an example of the application of the GANimation in facial expression transfer in which the facial expression of Donald Trump was transferred onto the face of Barack Obama.
Figure 5. Graphs of loss value of the generator

- Figure 5(a). Loss value of output AU values for the generator during training
- Figure 5(b). Loss value of the attention mask output of new generated image for the generator during training.
- Figure 5(c). Loss value of the attention mask output of real image existing in the training dataset for the generator during training.
- Figure 5(d). Loss value of the quality of new generated image by the generator during training.
Figure 6. Graphs of loss value of the discriminator

Figure 6(a). Loss value of AU value discrimination for the discriminator

Figure 6(b). Loss value of discriminating new generated image for the discriminator during training.

Figure 6(c). Loss value of discriminating real image existing in training dataset for the discriminator during training.

Figure 7. An example of applying GANimation model in facial expression transfer in which the facial expression of Donald Trump is transferred onto the face of Barack Obama.
4.2 Limitations

Although the overall facial expression generation result is satisfactory, the performance for certain AUs could be further improved. In particular, the performance regarding to AU17 (Chin Raiser), AU23 (Lip Tightener) and AU45 (Blink) could be improved. Figure 8 shows some sample result regarding to AU17, AU23 and AU45 in which from left to right, the images represent the case when: all AU values are 0, only the value of AU17 is 5, only the value of AU23 is 5, and only the value of AU45 is 5. As shown in Figure 8, there is observable noise produced when altering the value of AU17. For AU23 and AU45, although the results of generating new facial expression regarding to these two AUs are observable, the results are not significant.

Figure 8. Sample results of altering AU17, AU23 and AU45.

The above mentioned problem regarding to AU17, AU23 and AU45 is likely to be related to the distribution of presence of AUs and the intensity of AUs in the training dataset. Figure 9(a) shows the distribution of presence of AUs in the training dataset. Figure 9(b) shows the average values for each of the AUs in the training dataset. From Figure 9(a) and 9(b), AU17, AU23 and AU45 have the smaller number of presence and the lower AU values comparing to other AUs in the training dataset. In particular, the insignificance of facial expression generation results of AU23 and AU45 is related to the low AU values, while the noise occurs when altering AU17 is likely to be related to the lack of training data and possibly due to the quality of training data.

Figure 9(a). The distribution of number of presences of each AU in the training dataset.
5. Future Work

In the next stage of the project, the goal will be improvement of the current work in terms of quality and profitability. To achieve it, training data, the GANimation modal and OpenFace are three key points.

5.1 Improvement on training data

As mentioned in section 4.2, the dataset used to train is lack of some kinds of AUs (17, 23 and 45). We will try to find more training data and adjust the presence of each AUs of the dataset to see if it will bring any improvement in terms of quality.

5.2 Rewrite the GANimation modal

The code of the GANimation model is heavily relied on CUDA. An adaptation of the model without CUDA is required in order to make the trained model available on machines without GPU. Also, GANimation is using Pytorch as the main framework, which is not considered to be run on a mobile device. Therefore, for the extra stage of this project, in order to provide services on mobile devices, the model will be rewritten using Tensorflow to provide better mobile device support.

Moreover, the current size of the GANimation model is too large for real time processing. It needs to be resized for the extra stage of this project. It can be foreseen that there will be a trade-off of quality.

5.3 Implement OpenFace 2.0

Currently, the OpenFace toolkit is used in a standalone manner. Although the response time is short, it has a large file size of over 500MB. The main reason is that it is a complete toolkit and contains numerous of functions that are not used in this project. Therefore, to reduce the size, we can study the paper and train a model that is only for AU detection. The original OpenFace toolkit can be used to generate training data.
6. Conclusion

This project aims to explore the application of facial expression extraction and generation in facial expression transfer under a FACS conditioned scheme, as well as to implement an open source facial expression transfer software based on the GANimation program and OpenFace toolkit. The GANimation has been successfully trained and applied. A website has been developed to demonstrate the functionality of the GANimation model, and to provide service of facial expression transfer between photorealistic facial images. Currently, the software was implemented as a website in a server-client approach due to the CUDA requirement of the trained GANimation model. For remaining time of this project, the goal will be eliminating the CUDA requirement of the GANimation model and optimizing the model to provide better support for mobile device in order to complete the extra stage of this project.

7. Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>January 20, 2019</td>
<td>Deliverable of Phase 2</td>
</tr>
<tr>
<td></td>
<td>Detailed interim report</td>
</tr>
<tr>
<td></td>
<td>Preliminary implementation: an application performing facial</td>
</tr>
<tr>
<td></td>
<td>expression transfer using GANimation model and OpenFace toolkit</td>
</tr>
<tr>
<td>January 21 – February 14, 2019</td>
<td>If time allows, enter the extra stage, otherwise continuous the development and testing of the desktop application.</td>
</tr>
<tr>
<td></td>
<td><strong>Extra Stage:</strong></td>
</tr>
<tr>
<td></td>
<td>API for the mobile application demo</td>
</tr>
<tr>
<td></td>
<td>A mobile application performing remote computation by calling API</td>
</tr>
<tr>
<td>February 15 – April 13, 2019</td>
<td>A mobile application performing local computation</td>
</tr>
<tr>
<td>April 14, 2019</td>
<td>Deliverable of Phase 3</td>
</tr>
<tr>
<td></td>
<td>Finalized tested implementation: finalized desktop application</td>
</tr>
<tr>
<td></td>
<td>and/or a mobile version of the application</td>
</tr>
<tr>
<td></td>
<td>Final report</td>
</tr>
<tr>
<td>April 15 – 19, 2019</td>
<td>Final presentation</td>
</tr>
<tr>
<td>April 29, 2019</td>
<td>Project exhibition</td>
</tr>
</tbody>
</table>
8. References


