Bidimensional face morph

Marco Benelli
marco.benelli@stud.unifi.it

Luca Bindini
luca.bindini@stud.unifi.it

October 18, 2021

Abstract

In face rehabilitation, patients are asked to perform a face deformation and see how well they can deform their faces. This project aims to give a way to measure this deformation objectively.

Our approach uses frames taken from a colored video to create a 3D model of the person's relaxed face. Subsequently, a second model is created starting from the frames of the video where a certain grimace is performed and through the comparison of the models the extent and intensity with which the grimace was performed is detected.

1 Introduction

The main objective of this project is to create a 3D model from videos, which will be used to measure facial deformations.

The videos, stored in a data set, are from different people making different expressions. They are also already divided into the frames composing them.

From these we detect the facial landmarks, which are recognizable spots in someone's face. After that these landmarks are used to fit a 3D face model. Thanks to this approach we bypassed having to use a 3D scanner.

The 3D model is confronted with another model of the same person with a neutral pose to get an index of deformation and visualize where the facial expression is more pronounced.

The entire project is written in Python 3 and the external libraries used are the following:

- **numpy**[1] is a Python library useful for basic operations on multi-dimensional arrays and matrices. It has been used throughout the entire project.
- **matplotlib**[2] is a Python library for creating static, animated and interactive plots. This library was useful for creating plots and to visualize the 3D models.
- **h5py**[3] is a Pythonic interface to the HDF5 binary data format.
- **face_alignment**[4] is a face alignment network to detect facial landmarks from Python, capable of detecting points in 2D coordinates.
- **deep-3dmm-refinement**[5, 6] contains end-to-end demo code to recover accurate and detailed reconstructions of the facial shape from an unconstrained 2D face image.
- **SLC-3DMM**[7] contains the official Python implementation of SLC and Non-Rigid-Fitting algorithm (NRF).

2 Datasets

The dataset, provided by the *University of Florence*, is a collection of videos showing human faces in a relaxed position and while they are performing a certain grimace.

The videos have already been divided into frames and collected in different directories. The structure of the dataset is as follows: `LR/{person_name}/rgbReg_frames` and inside

Figure 1: This figure represents the frame of a relaxed face acquired by the Microsoft Kinect.

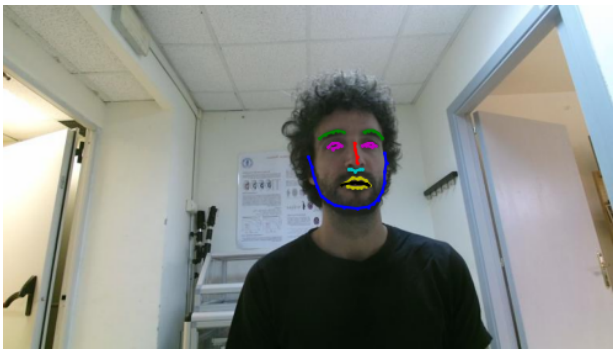

Figure 2: This figure represents a relaxed face with the landmarks superimposed on it.

that there are all the frames in directories called `0001`, `0002`, etc. Within these folders are contained the various frames of the videos in which the grimaces are performed. At the beginning of the video the face is in a relaxed state (first frames) and as you go on the grimace is performed.

The videos, from which all the frames that populate the dataset were then extracted, were captured using a *Microsoft Kinect* at a resolution of 768×432.

3 Landmarks detector

To find points of interest in a face, in order to build a subsequent model, we used the `face_alignment`[4] library.

Given an rgb frame depicting a face, the function `get_landmarks_from_image` returns a set of 68 landmarks divided as follows: 17 blue landmarks delimiting the face contour, 10 green landmarks representing the eyebrows, 4 red landmarks representing the nose profile, 5 cyan landmarks delimiting the nostrils, 12 magenta landmarks delimiting the eyes, 12 yellow landmarks delimiting the lips and 8 black landmarks representing the teeth. In figure 1 we can see a frame with the relaxed face and in figure 2 we can see the same frame with the landmarks detected.

4 3DMM

After we have gotten the 2D landmarks from the image, we need to create a 3D model of the face composed of 6704 points from just these 68 points. To do this we use something called *3D Morphable Model* (3DMM). This is a 3D model (meaning an ordered set of points in the space) that can be easily morphed to adapt to some landmarks.

As we will see in section 5, there are two types of fitting we can subject the landmarks to. The first one uses the average model generated in [6] as a base for the fitting; the second one uses a model obtained with a previous fitting.

These fittings are done using the functions found in `deep-3dmm-refinement`[5, 6] which allow for the fitting of a preexisting 3DMM to the given 2D landmarks. This preexisting 3DMM is usually either the average model, as we have already said, in the case we are trying to get the model of a face with a neutral expression, or a model specific to our person if we want to fit a face that is in a non neutral pose.

The reason for not using the average model for all of the fittings is that we want more precision on the results: if we were to adapt that 3DMM to a face in an extravagant expression we would be more likely to get an inaccurate, while if we use the 3DMM specific to that person (meaning the model obtained from that person's neutral pose) we can hope to get a better result.

Let's dive into more details about how the fitting is done. Suppose that $\mathbf{l} \in \mathbb{R}^{2\times n}$ is the matrix representing the n landmarks in the 2D plane. This would mean that those same landmarks in the 3D space can be represented by $\mathbf{L} \in \mathbb{R}^{3\times n}$, with these two matrices connected by the following affine transformation:

$$\mathbf{l} = \mathbf{A} \cdot \mathbf{L} + \mathbf{T}$$

where $\mathbf{A} \in \mathbb{R}^{2\times 3}$ and $\mathbf{T} \in \mathbb{R}^{2\times n}$.

The vertices in $\mathbf{L}$ are annotated in the 3D model, and so the unknowns are $\mathbf{A}$ and $\mathbf{T}$. $\mathbf{A}$ is found by solving this least square problem:

$$\arg\min_{\mathbf{A}} \|\mathbf{l} - \mathbf{A} \cdot \mathbf{L}\|_2$$

After that, we can easily get the remaining parameter: $\mathbf{T} = \mathbf{l} - \mathbf{A} \cdot \mathbf{L}$.

Once we have the affine transformation, we can look for a nonrigid transformation that would slightly change the position of the landmarks in the model in order to align them to the ones from the picture. This will then also move the other neighbouring points so as to have a smooth continuous transformation.

The nonrigid transformation will require some parameters to manipulate how the morphing is performed. The most important one of these is λ, a scalar that controls how aggressive the morphing will be: a lower λ means a more aggressive morphing. Through experimentation we have seen that we should use a lower λ for the initial fitting against the average model (we went for $\lambda = 0.15$) and a higher one for the following ones against the specific model ($\lambda = 64$).

What all of this does is create a way to get a 3DMM from a list of 2D landmarks which is really accurate considering the small number of landmarks. An example of a 3DMM can be seen in figure 3.

5 Pipeline

In this section we will see in detail the execution pipeline of the entire procedure focusing on a first naive implementation in which the 3D model is built with average components (useful for the coarse structure of the face).

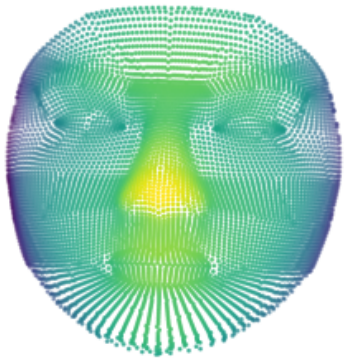

Figure 3: 3D model obtained by fitting the landmarks of a relaxed face with an average face model.

Later we will see the subsequent implementation where the first average model is used for a second fitting with local components (useful for appreciating small deformations of the face typical of grimaces).

Finally we will see how the deformation index is calculated between the model of the relaxed face and the model of the face making a grimace.

5.1 Naive 3D model construction

In the first instance we build a 3D model for the relaxed face by fitting the 68 landmarks of the frame using an average model that tries to capture the "global" information of the face (shape and gross deformations) and what we get is the model in the figure 3.

We do the same procedure fitting the landmarks taken from a grimace expression (in figure 4) with the same average model as before obtaining the model in figure 5.

As we can see from figure 5, the deformation in the model is also not very noticeable, despite the fa-

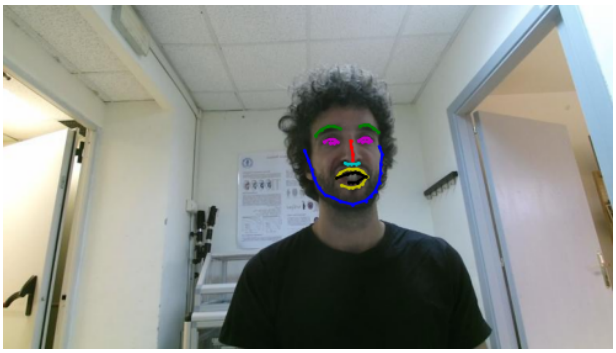

Figure 4: This figure represents a face that opens its mouth with the landmarks superimposed on it.

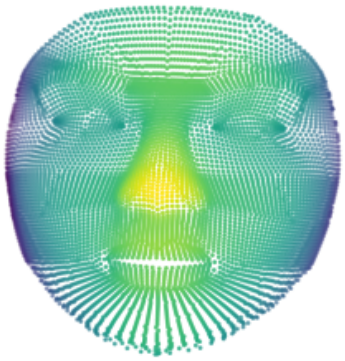

Figure 5: 3D model obtained by fitting the landmarks of figure 4 with an average face model

cial grimace being very pronounced. This suggests us that we should use additional local components in fitting phases that take into account even the smallest deformations of the face.

5.2 Double-fitting model construction

Due to the issues reported in the previous section we have built the starting model (relaxed face) by making a double fitting: the first one between the 2D landmarks of the frame and the average model, as in the naive implementation, then the fitting is performed between the same 2D landmarks of the frame but using the model resulting from the first fitting and using local components that take into account the smallest deformations of the face. This model was then used to compare it with the model created starting from 2D landmarks of the frames where the face is making the grimace, also this obtained with local components, and going to see how much the points of the model with the grimace, which are labeled, has moved from the relaxed position.

As can be seen from figure 6, the deformation of the smiling face is much more pronounced than the model that did not take into account the local components of the face, obviously on the same frame, in the figure 5.

To improve performance and avoid sporadic errors of individual models built from a single frame, the model is constructed in such a way that each point is the median between the same points of the three closest frames, including the frame itself.

5.3 Deformation Index

To evaluate how much and where the grimace model deviates from the relaxed face model, we calculate the distance between all 6704 3D points, which are labeled, between the two models.

Let $\mathbf{A} \in \mathbb{R}^{N\times 3}$ be a matrix where each row of the matrix is a point of the relaxed face model and the columns represent the spatial coordinates in 3D space. Also let $\mathbf{B} \in \mathbb{R}^{N\times 3}$ be a matrix where each row of the matrix is a point of the model of the face that performs the grimace.

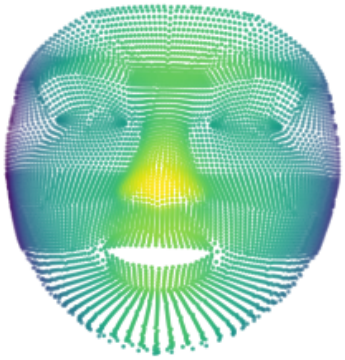

Figure 6: 3D model obtained by fitting the landmarks of figure 4 with the model obtained by double fitting the relaxed face model

The distance for each point i such that $0 < i \leq N$, corresponds to the distance between the i-th rows of the two matrix and is calculated using the Euclidean distance.

At this point we want a value that quantifies the extent of the deformation, called deformation index which is calculated like:

$$\frac{1}{N}\sum_{i=1}^{N}\sqrt{\sum_{j=1}^{3}(\mathbf{A}_{ij}-\mathbf{B}_{ij})^2}$$

6 Experimental results

Now that we've established what the pipeline is, we should look at the experiments to see how our approach fairs.

The first thing to look at is the 3DMM colored based on the face movement of each frame. In figure 7 we show the model of someone raising his eyebrows, the colors indicate how much each point has moved. We can clearly see that the program has correctly identified the eyebrow area as the area with the most deformation.

One nice thing we can do is compare a face that has completed a certain grimace with a face that is in the process of doing it, and therefore will exhibit the same deformations but to a lesser extent. The sequence of frames we have selected shows a man smiling. Two of the selected frames are the ones in the figures 8 and 9. The first picture has been taken while the person was starting to smile, therefore it's midway between a full smile and a neutral pose.

If we put together the models from these two frames, as well as the other ones at one quarter and three quarters of the sequence, we get figure 10. As we can all guess from the colors, the bottom model corresponds to the full smile, while the other ones corresponds to the intermediate frames. The figure also shows the deformation index we have defined in section 5, which is the mean movement of all the points of the model with respect to the neutral frame. For better comparison, the models are using the same scale, meaning that every color corresponds to the same amount of movement in all the models.

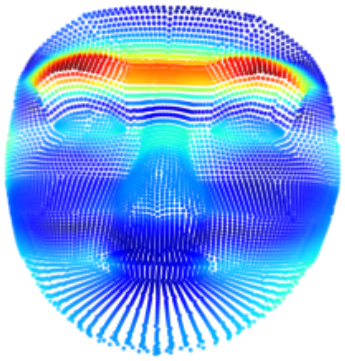

Figure 7: The model of a face with its eyebrows raised. The red areas indicate high deformation and, as expected, these are around the eyebrows.

Figure 8: The face of someone that is starting to smile.

Figure 9: The face of someone smiling.

Instead of just looking at the final and the intermediate frame, we can look at many more of them, but instead of visualizing the entire model, we decided to show just the index of deformation. In figure 11 we see how the index of deformation varies throughout the video. In particular we took measurements of nine frames equally spaced so that we selected the first frame, the last frames and the frames in between space at eighths of the full distance.

We can plot the same thing for all the grimaces we have on the same graph and get figure 12. From it, we can get an idea of the deformation trends in the videos. All of the curves tend to go upward since the videos start from a relaxed pose and go towards a full expression. Most of the curves tend to plateau at a certain maximum index of deformation, which is specific to the video in question. This is because some grimaces will inevitably lead to more deformation than other ones.

7 Conclusions

In conclusion, we have written a program that can find the zones of face deformation starting from just 2D images with good accuracy, as we have seen from figures 7 and 10. At the end of the article, you can find table 1, a table showing more models and their deformation, which show similar results.

This is great for face rehabilitation since people won't have to sustain high costs for specialized equipment and can just use a simple video camera.

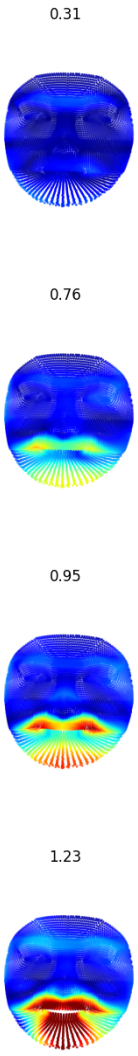

Figure 10: The models taken from the frames of the smiling sequence. The time flows from top to bottom, and the frames are chosen at fourths of the sequence. We can clearly see how the colors change from cold ones to warm ones as we go forward in time. The numbers on the top of each model show the index of deformation of the two faces and, as expected, they increase from top to bottom.

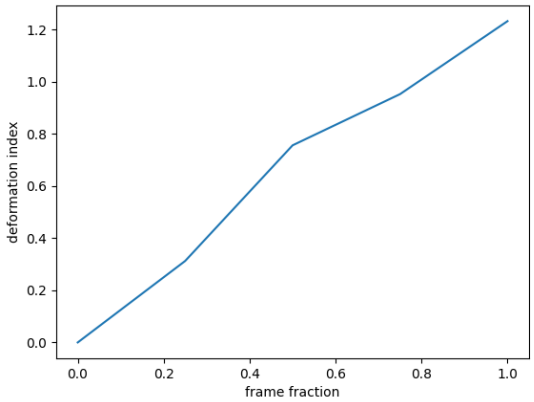

Figure 11: How the deformation with respect to the first frame varies throughout the video. The frames chosen are 9 in total, taken at interval of eighths. The first frame has a deformation of zero since it is the one we are comparing against.

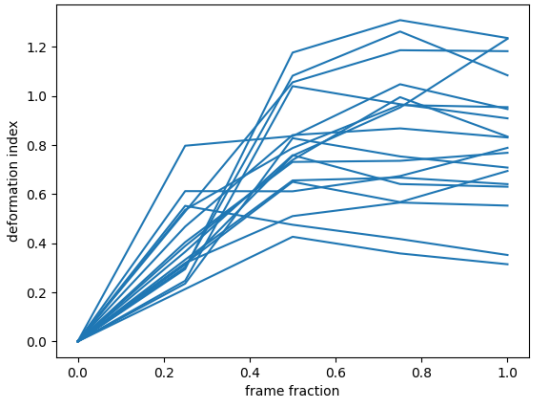

Figure 12: The plots of the deformation variation for all the videos. We can see that they all start from zero and have an ascending trend. The deformation indices for the final frames vary dramatically between each other since some grimaces involve more deformation than others.

Something we want to put stress on is that the parameter λ in the fitting of the local components is really important to get right. This is not that hard, since this parameter is not very sensitive to variations and therefore choosing double of what we chose or half of it, you are still going to be fine, but if you were to choose something too small, we could get horribly wrong results.

If you would enjoy having a look at the source code, you can find it on our GitHub repository at this link: `https://github.com/lucabindini/bidimensional-face-morph`.

References

[1] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[2] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[3] Andrew Collette. *Python and HDF5*. O'Reilly Media, Inc., November 2013.

[4] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks). In *International Conference on Computer Vision*, 2017.

[5] Leonardo Galteri, Claudio Ferrari, Giuseppe Lisanti, Stefano Berretti, and Alberto Del Bimbo. Deep 3d morphable model refinement via progressive growing of conditional generative adversarial networks. *Computer Vision and Image Understanding*, 185:31–42, 2019.

[6] Claudio Ferrari, Giuseppe Lisanti, Stefano Berretti, and Alberto Del Bimbo. A dictionary learning-based 3d morphable shape model. *IEEE Transactions on Multimedia*, 19(12):2666–2679, 2017.

[7] Claudio Ferrari, Stefano Berretti, Pietro Pala, and Alberto Del Bimbo. A sparse and locally coherent morphable face model for dense semantic correspondence across heterogeneous 3d faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

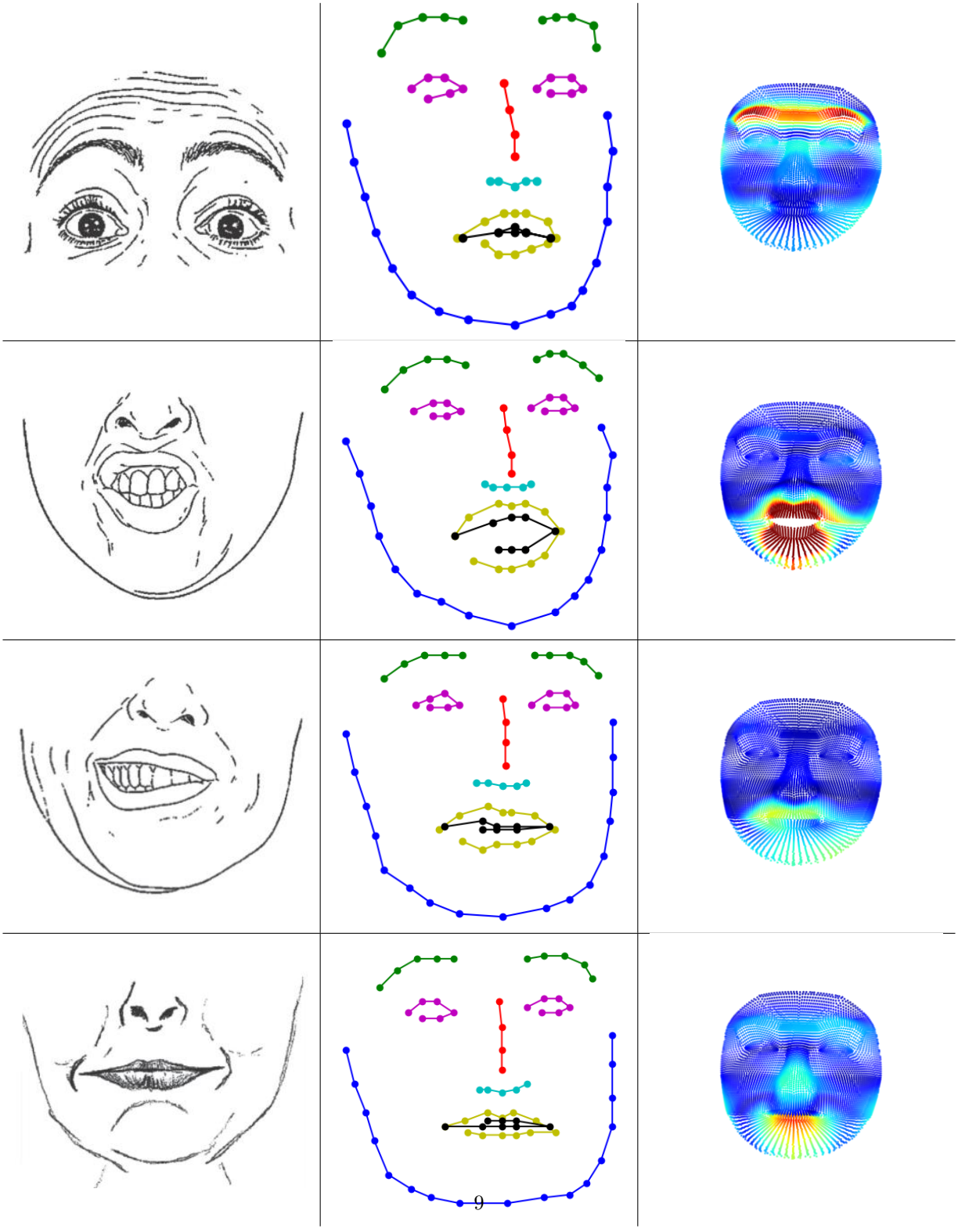

Table 1: The results of four selected expressions.