Face matching

Hugo Bertiche [*1] and Xavier Serra [†2]

[1]Master in Artificial Intelligence, UPC-UB-URV
[2]Master in Artificial Intelligence, UPC-UB-URV

January 2016

1 Exercise 1

In this first exercise we were asked only two questions:

1. *Explain which is the result showed in the Matlab Figures in Figure 1 below.*

   The Figure 1 mentioned is provided here:

Figure 1: Reconstruction of the face

In this figure we have the reconstruction of the face done by the AAM (Active Appearance Model). Basically, given an approximation of the appearance model, it fits it in the actual image. In our case, from the 26 images in the train set, we select, each time we type a key in the keyboard, a different "shape" of an image. Using this shape, we try to adapt it to the current image, and "modify" the image corresponding to the shape to fit the current position. The result is the Figure 1. In Figure 2 we have the image chosen to test and, in blue, the shape that we have adapted and, in red, the adaptation performed. By pressing the keyboard many types we can see as, starting

[*]hugo_bertiche@hotmail.com
[†]xserralza@gmail.com

from different shapes, we end up with similar ones. And, if we compare the original image with the obtained one, we can see that the resemblance is very high.

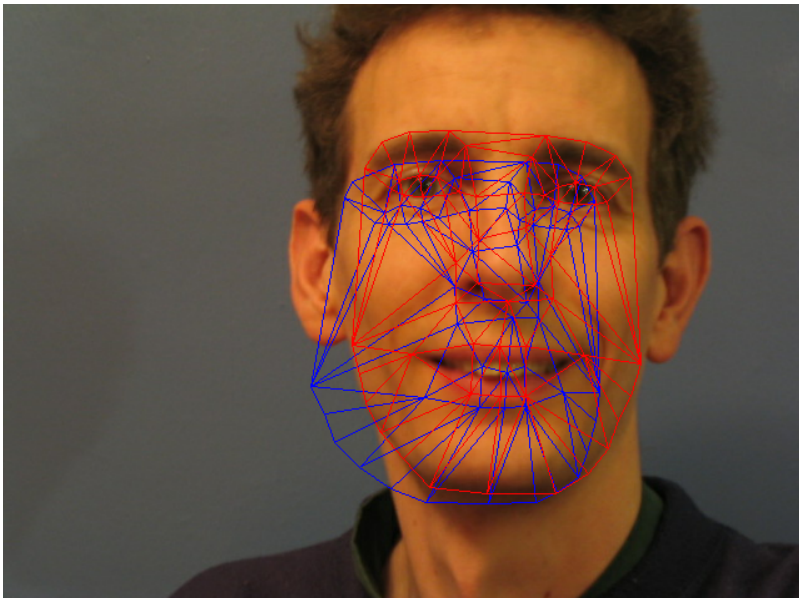

Figure 2: Adaption of the shapes

2. *Explain how the 'Number of Shape Modes' and the 'Number of Appearance Modes' are computed in the function build_ model_ 2d.m.*

   Both values are computed following the same principle:

   > From all the shapes/modes, we obtain the principal components and, from these, we select only the most relevant ones, until we have covered the 98% of the original variance. This way, we get rid of some of the faces (less complexity and faster), while keeping most of the information in the original dataset.

2 Exercise 2

Once again, we were asked to change the test image and to answer to perform the following modification in the code:

- *In the function build_ model_ 2d.m, build two images with all the training shapes before and after the alignment using Procrustes Analysis. As it is done in the Figure 2 below. Use the function triplot.m with the matrix 'shape triangles' as the first argument.*

  And so we have done and, using the first image, the result we have obtained is shown in the following figure:

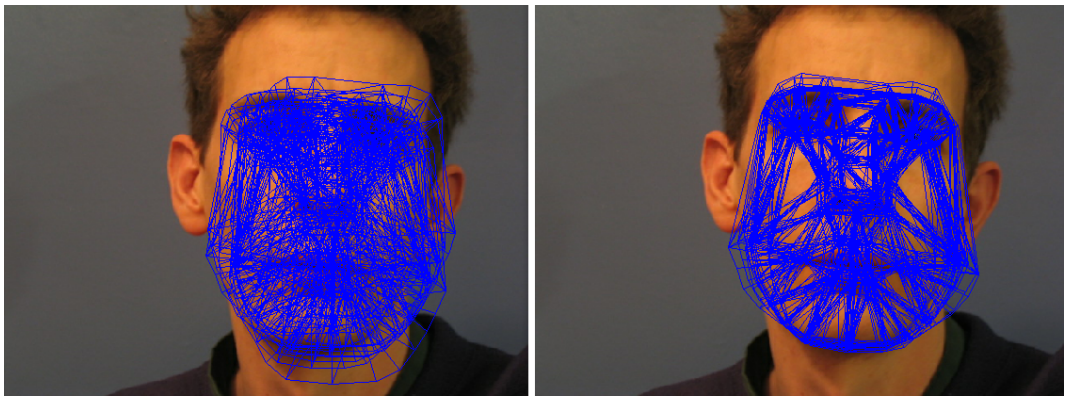

Figure 3: Shapes before and after alignment

As we can see, all of them are, now more centered. The motivation to do so is that the shape vectors in the training set should be in the same coordinate frame. In order to perform this alignment the process followed is to find the set of transformations **T** such that the following formula is minimized:

$$D = \sum |\bar{x} - T_i(x_i)|^2$$

, in which $\bar{x}$ is the mean of all shapes. Therefore, we obtain, in the end, all train shapes in similar coordinates, which will help in the training process.