Computer Vision

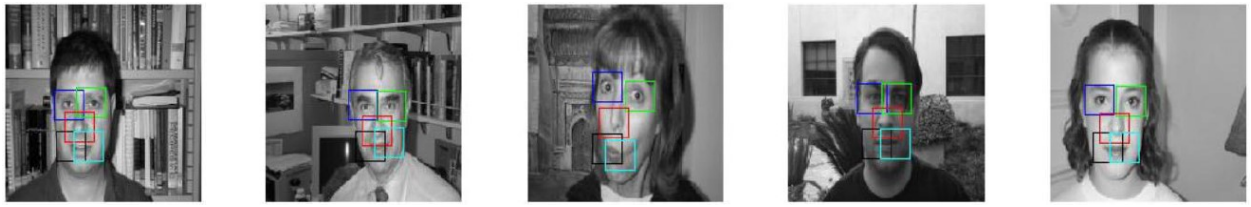
Exercises of Lab 10

Exercise 10.1: Part-based mdel for face detection

The goal of this exercise is to give you some intuition about how part-based models work. A part-based model consists of a number of parts (e.g., eyes, nose, and mouth) and a spatial configuration of the parts. Given an input image the algorithm first searches for candidate occurrences of each part in the image. Whenever the parts are in the correct spatial configuration, we say that we have detected a face.

Open Exercise10.1.m and look through the code.

Task 1: Run the code (it will not work the first time). Five training images are displayed one at a time (see figure below). Mark the spatial position of each of the five parts in each training image using the mouse. The five parts are: Right eye, left eye, nose tip, right mouth, and left mouth. Make sure that you specify the parts in that order! The results are stored in the file my_parts.mat.



Task 2: Calculate the average appearance each part (intensities of the patches averaged over the five training images). Replace the code in line 96:

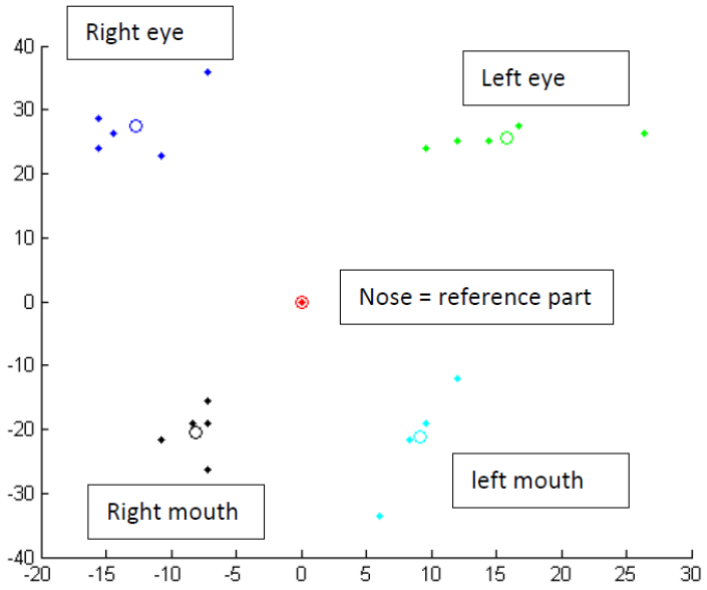
```
% TASK 2:
% Calculate average/mean patch for the current part
part_patch(:,:,part) = rand(patchsz,patchsz); % REPLACE!!!
```

Check that your results look fairly similar to these:

Task 3: Calculate the average position of each part relative to the reference part, which in this example is the nose tip (part 3). Replace the code in lines 114+115

```
% TASK 3:
% Calculate average part position
part_pos(1,part) = randn(1); % x coordinate (REPLACE!!!)
part_pos(2,part) = randn(1); % x coordinate (REPLACE!!!)
```

Make sure that the result looks fairly similar to this (circles correspond to averaged positions, whereas dots correspond to part positions in individual training images).



Task 4: Given a test image, the code performs normalized cross correlation (NCC) with each of the five templates (averaged patches) shown above. The resulting NCC maps are spatially shifted and summed together to form a single NCC map, where peaks potentially correspond to faces in the images. Figure out what happens in lines 143-154 and why.

Exercise 10.2: Active appearance models for pain detection

In this exercise, we will use active appearance models (AAMs) and support vector machines (SVMs) to learn to recognize pain from facial expressions. We will use a small subset of the data available from the UNBC-McMaster Shoulder Pain Expression Archive Database.

AAM Toolbox download:

Start by downloading the Active Shape Model (ASM) and AAM toolbox for Matlab, developed by Dirk-Jan Kroon. The toolbox homepage is here. The Matlab code can be downloaded here:

http://www.mathworks.com/matlabcentral/fileexchange/submissions/26706/v/6/download/zip

Task 1: Open Exercise10.2.m and look through the code. Change the path to the AAM toolbox (variable AAM_DIR). Run the code until it crashes. Figure 1 shows the loading of the training images + landmarks. Figure 2 shows what the first 10 principal components of the spatial model. What does Figure 3 show? What does Figure 4 show?

Task 2: Perform principal component analysis on the aligned textures (variable J). See AAM_MakeShapeModel2D.m for inspiration. The first 5 principal components of the texture model are shown in Figure 5.

Task 3: Figure 6 shows the results of using the spatial model as input to an SVM that is trained to predict pain. Here, we use 180 images for training and the remaining 20 images for testing. Implement the same procedure, but where we instead use the texture model as input to the SVM.