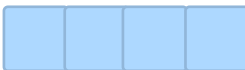


Biometric Systems - 02238

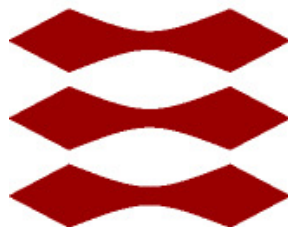
Research Paper:

Face Quality Enhancement (FQE)

Daniel 

June 2018

DTU



Facial Detection

With the advances and spread of both surveillance cameras and face detection technology, this research paper aims to analyze the limitations of combining these technologies for mass surveillance. A preliminary investigation on the overall impact of sample quality on detection ability is tested by applying OpenCV's feature detection to a live camera stream.

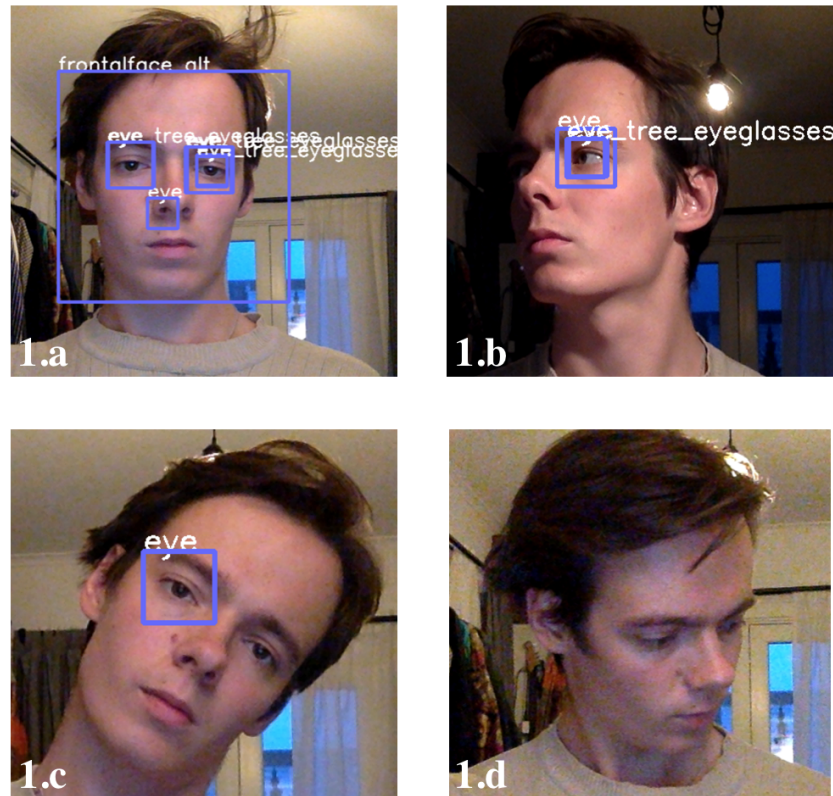


Figure 1: Variations in lighting and head position can prevent facial detection.

The 720p front-facing camera of a laptop revealed that even when the subject is centered and consumes a majority portion of the screen, OpenCV feature detection quality depends primarily on direct or neutral lighting and prefers a frontal-full-face at camera level with no forward/backward and side-to-side tilt.

- **1.a** - number of different classifications including misclassification of an eye for a nostril.
- **1.b** - lack of face detection due to sharp light.
- **1.c** - lack of face detection due head tilt.
- **1.d** - complete lack of detection due to combined forward and side head tilt

In the case of a security camera watching over a stream of pedestrians, a subject may have an easily detectable head posture in good lighting for a single frame or two. With the help of OpenCV's face detection library and MIT-CBCL's synthesized face database, this report investigates to what extent resolution / distance (simulated through re-sampling) affects detection ability in a face database with mixed lighting and head orientation.

The Data

The MIT-CBCL face recognition database consists of 3240 images synthesized from high definition 3D models of 10 individuals heads with varying lighting angles and differing-facing-angles. Both of which are parameters that pose a challenge to OpenCV's facial detection system. Examples are shown in figure 2:



Figure 2: One subject's face synthesized with differing light and head angles.

The due to the parameter range for light and face angle as well as digital aliasing from bad synthesis cause some faces in the database to be unrecognizable by all classifiers even at original resolution. Some of these are shown in figure 3.



Figure 3: Badly synthesized faces from the MIT-CBCL face database.

Detection Method

OpenCV's python library `cv2`, performs face detection. The process involves analyzing differences between neighboring regions of in an image, to find certain features. The features may vary in complexity from diagonal and vertical edges to human faces and number plates.

The features are detected using *Haar feature-based Cascade Classifiers* which are classifier models pre-trained to specific features using machine learning and large sets of images.

The OpenCV library contains a range of Haar cascades ranging for detecting features like Russian license plates, cat faces and sub-features of human faces. To test how the image quality effects detection accuracy, several cascades are chosen - but not too many to keep computational time reasonable.

Choosing OpenCV Cascades

The initial scanning yields the following OpenCV cascades as the top performers:

Cascade	Successes
haarcascade_frontalface_alt	2928
haarcascade_frontalface_alt2	3221
haarcascade_frontalface_default	2887
haarcascade_frontalface_profileface	1813

Table 1: Faces detected out of 3240 synthesized images

Several feature cascades are omitted due to irregular performance:

- `haarcascade_smile` yields high false negative rates occasionally mis-classifying a thin eyebrow as a smile.
- `lefteye_2splits` and `righteye_2splits` are asymmetric features and under-perform on slightly side-facing images or uneven lighting.

Implementation and Results

Emulating distance from a camera and measuring its effect is done by down-sampling images and performing facial detection at different qualities of down-sampling. The down-sampling is done on 20 different levels preserving 5% and 100% of the data in the image. The original images are 200x200 pixels, meaning the lowest resolution tested is 10x10 pixels (i.e 5%).

Face detection is performed on each image with four feature cascades. The accuracy of detection for each image is rated on a level from 0-4 determined by how many of the cascades could detect a face.

The implementation is done in Python and can be described in three grossly simplified stages:

- Loading cascades and images into memory
- Down-sampling images on differing levels
- Performing multi-cascade face detection on each of the levels
- Counting detection and categorizing image into level 0-4

The results can be seen on figure 4 which shows the distribution of detection accuracy across the range down-sampling. Judging by the steep drop-off at 0.25 and peaks in well-detected images at 0.75, 0.5, 0.35 there is significant of misclassification at play.

To prevent misclassification in down-sampled images, the coordinates of the facial region detected in the original image should be compared to the coordinates of facial image detected in the down-sampled version. This, however, is out of scope of this assignment.

Sources

- MIT CBL database
<http://cbcl.mit.edu/software-datasets/heisele/facerecognition-database.html>
- OpenCV
https://docs.opencv.org/3.4.1/d7/d8b/tutorial_py_face_detection.html