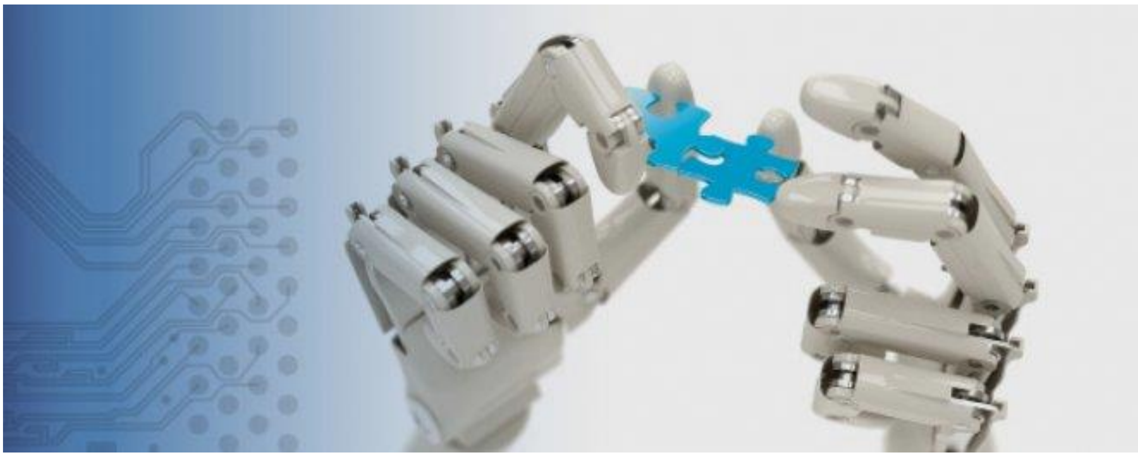
ROBOTICS MASTER

Universitat de Vic



Subject: Perception Systems

Session7: OpenCV

Exercixse 7.1: OpenCV

Author: Toni Guasch Serra

Date: 2015-12-22

Exercise OpenCV

https://github.com/aramisa/RoboticaUVic-PR-week7

RoboticaUVic-PR-week7

UVic Robotics Master. Pattern Recognition Homework 4

Instructions

Fork this repository and write code to complete the assignments. First upload the completed assignment to the course Moodle for grading; then, correct all the issues marked by the teacher and push it to GitHub for the final evaluation.

This is a personal assignment, please complete it **individually**.

OpenCV

- **Q1**) The file opencv-facedetect.ipynb has code to open a video file and detect faces using a pre-trained cascade of classifiers from OpenCV. Your task will be to read and understand the code, and re-write it using C++ to detect faces in a video stream coming from a webcam. You can use the code you developed in earlier homework assignments as a template.

Extra

- **Q2**) Now that you can locate the faces in the video, let's try to decorate them! Load the Charlie Chaplin hat and mustache, and paint them in the image at the appropriate locations based on the detected faces.
- **Q3**) Haar features are not the only type available in OpenCV. There are also cascade detectors trained using Local Binary Patterns (LBP). Use the LBP based cascade detector and compare the performance of both. You can find this (and other cascades) in /usr/share/opencv/, or online at the OpenCV github.
- **Q4**) The example provided in the notebook is a "bare bones" one. For a more complete example see facedetect.py. Read it and understand how it works.

Exercise solution code uploaded to Toni Guasch GitHub profile:

https://github.com/ToniSkan/Face_Detection

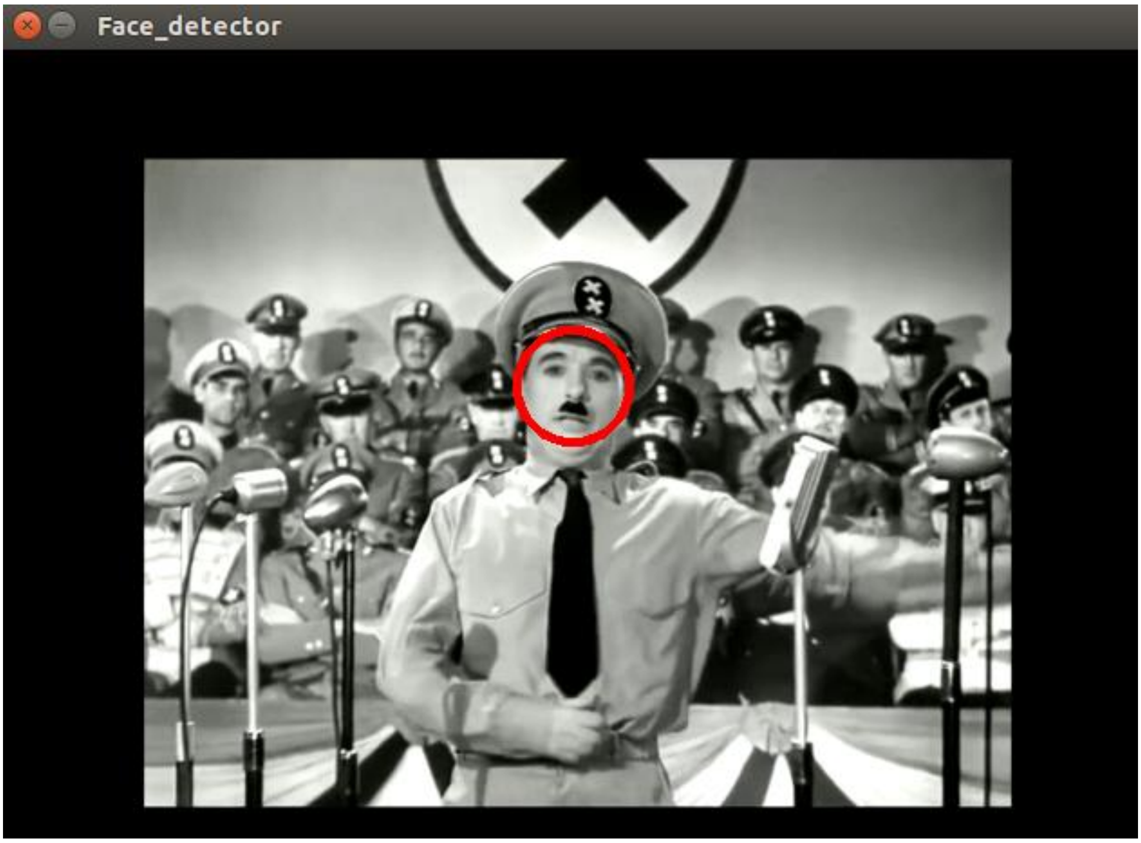
Code in Python:

```
#Get sample video from: https://mega.nz/#!NUtDiTgQ!xFRWP91fdwaoCShTGbVVKXP_sm6PF-pHt5VbNL_s1fE
import cv2
import numpy as np

#Initialize video capture. if an integer is passed instead of a string
#the video source with that index is used
video = cv2.VideoCapture('video.mpeg')
flag = True

#Create window to visualize results
cv2.namedWindow('faces')

#Initialize face detector object
face_detect = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')



#Main loop
while flag:
  #Load next frame
  flag, frame = video.read()
  #Convert frame to grayscale
  gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
  #Detect faces as rectangles
  rects = face_detect.detectMultiScale(gray, scaleFactor=1.3,

                        minNeighbors=4, minSize=(30, 30),
                        flags = cv2.CASCADE_SCALE_IMAGE)

  #Copy frame for visualization
  img = frame.copy()
  green = (0, 255, 0)

  #For each detected face...
  for x1, y1, x2, y2 in rects:
    #Draw rectangle in duplicated image
    cv2.rectangle(img, (x1, y1), (x2+x1, y2+y1), green, 2)
  #Show in window
  cv2.imshow('faces', img)

  #Wait for keypress
  key = cv2.waitKey(5)
  if key == 'q':
    flag = False
cv2.destroyAllWindows()
---------------------------------------------------------------------------
error                            Traceback (most recent call last)
<ipython-input-3-33851846d07b> in <module>()
    13     flag, frame = video.read()
    14     #Convert frame to grayscale
---> 15     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    16     #Detect faces as rectangles
    17     rects = face_detect.detectMultiScale(gray, scaleFactor=1.3,

error: /build/buildd/opencv-2.4.8+dfsg1/modules/imgproc/src/color.cpp:3737: error: (-215) scn == 3 || scn == 4 in function
cvtColor
```

Solution to Q1 in C++: (FaceDetector.cpp)

```
//FACE DETECTION CODE USING OPENCV    |
//TONI GUASCH SERRA                   |
//------------------------------------|
//FOLLOWED DOC:
// http://docs.opencv.org/2.4/doc/tutorials/objdetect/cascade_classifier/cascade_classifier.html
// http://docs.opencv.org/2.4/modules/gpu/doc/object_detection.html
// http://docs.opencv.org/2.4/modules/highgui/doc/reading_and_writing_images_and_video.html#videocapture
//--------------------------------------------------------------------------

//Libraries
#include "cv.h"
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "highgui.h"
#include <iostream>
#include <cstdlib>

//Variables
using namespace cv;
using namespace std;
void detectAndDisplay(Mat frame);
//Set source of video:
        const string filename = "/home/toni/Documents/GIT/Face_Detection/Face_Detection/src/video.mpeg";
//Set Classifier XML source:
        String                                  face_cascade_name                                  =
"/home/toni/Documents/GIT/Face_Detection/Face_Detection/src/haarcascade_frontalface_default.xml";
        CascadeClassifier face_cascade;

//-----------------------------------------------------------------------------------//
//MAIN FUNCTION                                                                   //
//-----------------------------------------------------------------------------------//
int main(int argc, char *argv[])
{
        Mat frame;
//Load the cascade
        if( !face_cascade.load(face_cascade_name)){ printf("--(!)Error loading\n"); return -1; };

//Read the video file
        VideoCapture capture(filename);
  if( !capture.isOpened())        throw "Error reading video file";
        while(true) {
                //Get a frame
                capture >> frame;

                //Apply the classifier to the frame
                if( !frame.empty())detectAndDisplay(frame);

                else {
                        cout << "No captured frame -- Break!" << endl;
                        break;
                }

                int c = waitKey(10);
                if( (char)c == 'c' ) break;
        }
        return 0;
}


//-----------------------------------------------------------------------------------//
//Function DETECT and DISPLAY                                                     //
//-----------------------------------------------------------------------------------//
void detectAndDisplay(Mat frame)
{
        vector<Rect> faces;
        Mat frame_gray;

        //Convert frame to grayscale
```

```
            cvtColor( frame, frame_gray, CV_BGR2GRAY );
            equalizeHist( frame_gray, frame_gray );

            //Detect Faces
            face_cascade.detectMultiScale(frame_gray, faces, 1.3, 4, 0|CV_HAAR_SCALE_IMAGE, Size(30, 30));

            for( int i = 0; i < faces.size(); i++)
                        {
                        //center
                        Point center( faces[i].x + faces[i].width*0.5, faces[i].y + faces[i].height*0.5 );
                        //Draw circle in detected phase with center
                         ellipse( frame, center, Size( faces[i].width*0.5, faces[i].height*0.5), 0, 0, 360, Scalar( 0, 0, 255 ), 4, 8, 0 );

                        }
            imshow("Face_detector", frame);
}
```


Face_detector