OBJECT DETECTION : FACE DETECTION USING HAAR CASCADE CLASSFIERS

K Hong
google.com/+KHongSanFFrancisc...
Follow
4,629 followers

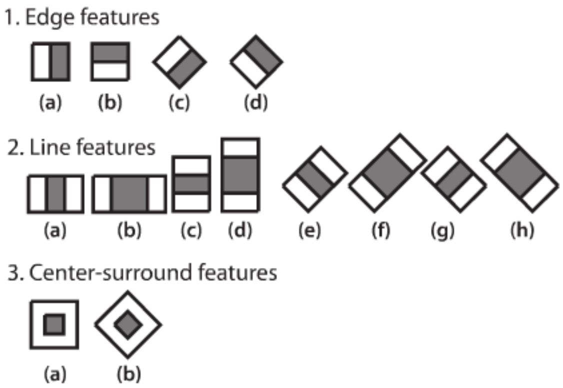
Ph.D. / Golden Gate Ave, San Francisco / Seoul National Univ / Carnegie Mellon / UC Berkeley / DevOps / Deep Learning / Visualization

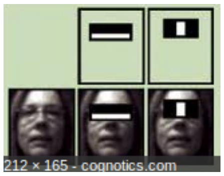
Face Detection

"Face detection is a computer technology that determines the locations and sizes of human faces in arbitrary (digital) images. It detects facial features and ignores anything else, such as buildings, trees and bodies. Face detection can be regarded as a more general case of face localization. In face localization, the task is to find the locations and sizes of a known number of faces (usually one)." - wiki - Face detection (http://en.wikipedia.org/wiki/Face_detection)

Haar features

OpenCV's algorithm is currently using the following Haar-like features which are the input to the basic classifiers:





Picture source: How Face Detection Works (http://www.cognotics.com/opencv/servo_2007_series/part_2/sidebar.html)

Cascade of Classifiers

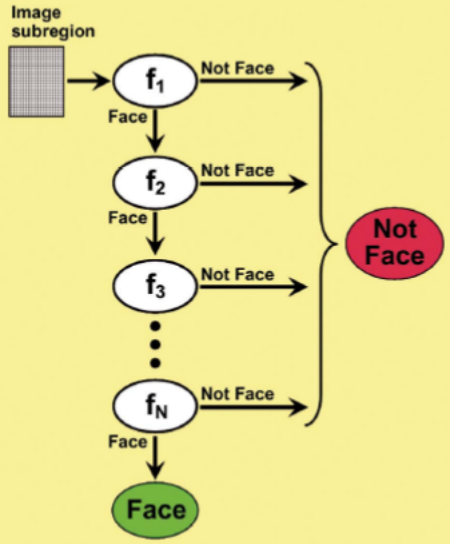
OpenCV 3 image and video processing with Python

OpenCV 3 with Python (/python/OpenCV_Python/pythor

Image - OpenCV BGR : Matplotlib RGB (/python/OpenCV_Python/pythor

Basic image operations - pixel access (/python/OpenCV_Python/pythor

"Instead of applying all the 6000 features on a window, group the features into different stages of classifiers and apply one-by-one. (Normally first few stages will contain very less number of features). If a window fails the first stage, discard it. We don't consider remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region." - Face Detection using Haar Cascades (http://docs.opencv.org/trunk/doc/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html#face-detection).



Picture source: How Face Detection Works (http://www.cognotics.com/opencv/servo_2007_series/part_2/sidebar.html)

OpenCV's pre-trained classifiers

OpenCV already contains many pre-trained classifiers for face, eyes, smile etc. Those XML files are stored in **opencv/data/haarcascades/** folder:

```
~/OpenCV/opencv/data/haarcascades$ ls

haarcascade_eye_tree_eyeglasses.xml   haarcascade_mcs_leftear.xml
haarcascade_eye.xml                   haarcascade_mcs_lefteye.xml
haarcascade_frontalface_alt2.xml      haarcascade_mcs_mouth.xml
haarcascade_frontalface_alt_tree.xml  haarcascade_mcs_nose.xml
haarcascade_frontalface_alt.xml       haarcascade_mcs_rightear.xml
haarcascade_frontalface_default.xml   haarcascade_mcs_righteye.xml
haarcascade_fullbody.xml              haarcascade_mcs_upperbody.xml
haarcascade_lefteye_2splits.xml       haarcascade_profileface.xml
haarcascade_lowerbody.xml             haarcascade_righteye_2splits.xml
haarcascade_mcs_eyepair_big.xml       haarcascade_smile.xml
haarcascade_mcs_eyepair_small.xml     haarcascade_upperbody.xml
```

iPython - Signal Processing with NumPy (/python/OpenCV_Python/pythor

Signal Processing with NumPy I - FFT and DFT for sine, square waves, unitpulse, and random signal (/python/OpenCV_Python/pythor

Signal Processing with NumPy II - Image Fourier Transform : FFT & DFT (/python/OpenCV_Python/pythor

Inverse Fourier Transform of an Image with low pass filter: cv2.idft() (/python/OpenCV_Python/pythor

Image Histogram (/python/OpenCV_Python/pythor

Video Capture and Switching colorspaces - RGB / HSV (/python/OpenCV_Python/pythor

Adaptive Thresholding - Otsu's clustering-based image thresholding (/python/OpenCV_Python/pythor

Edge Detection - Sobel and Laplacian Kernels (/python/OpenCV_Python/pythor

Canny Edge Detection (/python/OpenCV_Python/pythor

Hough Transform - Circles (/python/OpenCV_Python/pythor

Watershed Algorithm : Marker-based Segmentation I (/python/OpenCV_Python/pythor

Watershed Algorithm : Marker-based Segmentation II (/python/OpenCV_Python/pythor

Image noise reduction : Non-local Means denoising algorithm (/python/OpenCV_Python/pythor

OpenCV's face detection

Let's load the required XML classifiers.

```
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
```

Then, we need to load input image in grayscale mode:

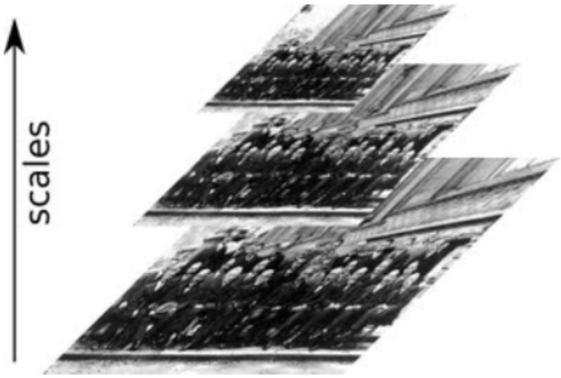
```
img = cv2.imread('xfiles4.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

We use **v2.CascadeClassifier.detectMultiScale()** to find faces or eyes, and it is defined like this:

```
cv2.CascadeClassifier.detectMultiScale(image[, scaleFactor[, minNeighbors[, flags[, minSize[
```

Where the parameters are:

1. **image** : Matrix of the type CV_8U containing an image where objects are detected.
2. **scaleFactor** : Parameter specifying how much the image size is reduced at each image scale.



Picture source: Viola-Jones Face Detection ()
This scale factor is used to create scale pyramid as shown in the picture. Suppose, the scale factor is 1.03, it means we're using a small step for resizing, i.e. reduce size by 3 %, we increase the chance of a matching size with the model for detection is found, while it's expensive.
3. **minNeighbors** : Parameter specifying how many neighbors each candidate rectangle should have to retain it. This parameter will affect the quality of the detected faces: higher value results in less detections but with higher quality. We're using 5 in the code.
4. **flags** : Parameter with the same meaning for an old cascade as in the function cvHaarDetectObjects. It is not used for a new cascade.
5. **minSize** : Minimum possible object size. Objects smaller than that are ignored.

local_Means_Denoising_Algorithr

Image object detection : Face detection using Haar Cascade Classifiers (/python/OpenCV_Python/pythor

Image segmentation - Foreground extraction Grabcut algorithm based on graph cuts (/python/OpenCV_Python/pythor

Image Reconstruction - Inpainting (Interpolation) - Fast Marching Methods (/python/OpenCV_Python/pythor

Video : Mean shift object tracking (/python/OpenCV_Python/pythor

Machine Learning : Clustering - K-Means clustering I (/python/OpenCV_Python/pythor Means_Clustering_Vector_Quanti

Machine Learning : Clustering - K-Means clustering II (/python/OpenCV_Python/pythor Means_Clustering_Vector_Quanti

Machine Learning : Classification - k-nearest neighbors (k-NN) algorithm (/python/OpenCV_Python/pythor nearest_neighbors_k-NN.php)

6. **maxSize** : Maximum possible object size. Objects larger than that are ignored.

If faces are found, it returns the positions of detected faces as Rect(x,y,w,h).

```
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
```

Once we get these locations, we can create a ROI for the face and apply eye detection on this ROI.

The Code

```
import numpy as np
import cv2
from matplotlib import pyplot as plt

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')

img = cv2.imread('xfiles4.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

faces = face_cascade.detectMultiScale(gray, 1.3, 5)


for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output 1

Python tutorial

Python Home (/python/pytut.php)

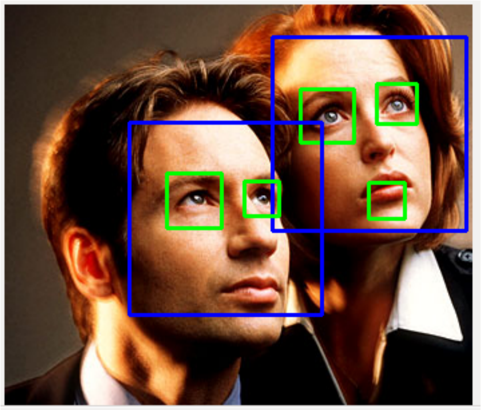
Introduction (/python/python_introduction.ph

Running Python Programs (os, sys, import) (/python/python_running.php)

Modules and IDLE (Import, Reload, exec) (/python/python_modules_idle.p

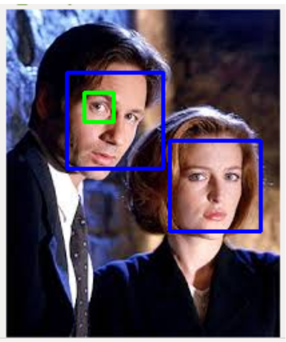
Object Types - Numbers, Strings, and None (/python/python_numbers_string

Strings - Escape Sequence, Raw



We're almost there except we got an additional eye.

Output 2



I got the image using:

```
faces = face_cascade.detectMultiScale(gray, 1.03, 3)
```

But if with scaleFactor = 3 and minNeighbors = 5, I got this:

String, and Slicing (/python/python_strings.php)

Strings - Methods (/python/python_strings_method

Formatting Strings - expressions and method calls (/python/python_string_formattir

Files and os.path (/python/python_files.php)

Traversing directories recursively (/python/python_traversing_direc

Subprocess Module (/python/python_subprocess_mc

Regular Expressions with Python (/python/python_regularExpressi

Object Types - Lists (/python/python_lists.php)

Object Types - Dictionaries and Tuples (/python/python_dictionaries_tup

Functions def, *args, **kargs (/python/python_functions_def.p

Functions lambda (/python/python_functions_lamb

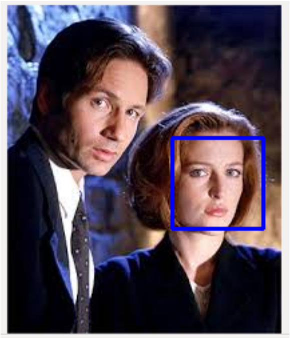
Built-in Functions (/python/python_functions_built_

map, filter, and reduce (/python/python_fncs_map_filter_

Decorators (/python/python_decorators.php

List Comprehension (/python/python_list_comprehen

Sets (union/intersection) and itertools - Jaccard coefficient and shingling to check plagiarism (/python/python_sets_union_inte



Hashing (Hash tables and hashlib) (/python/python_hash_tables_ha

Dictionary Comprehension with zip (/python/python_dictionary_com

The yield keyword (/python/python_function_with_y

Generator Functions and Expressions (/python/python_generators.php

generator.send() method (/python/python_function_with_g

Iterators (/python/python_iterators.php)

Classes and Instances (__init__, __call__, etc.) (/python/python_classes_instanc

if__name__ == '__main__' (/python/python_if__name__equa

argparse (/python/python_argparse.php)

Exceptions (/python/python_try_except_final

@static method vs class method (/python/python_differences_bet

Private attributes and private methods (/python/python_private_attribut

bits, bytes, bitstring, and constBitStream (/python/python_bits_bytes_bitst

json.dump(s) and json.load(s) (/python/python-json-dumps-loads-file-read-write.php)

Python Object Serialization - pickle and json (/python/python_serialization_pic

Python Object Serialization - yaml and json (/python/python_yaml_json_conv

Priority queue and heap queue data structure (/python/python_PriorityQueue_

Graph data structure (/python/python_graph_data_stru

Dijkstra's shortest path algorithm (/python/python_Dijkstras_Short

Prim's spanning tree algorithm (/python/python_Prims_Spanning

Closure (/python/python_closure.php)

Functional programming in Python (/python/python_functional_prog

Remote running a local file using ssh (/python/python_ssh_remote_ru

SQLite 3 - A. Connecting to DB, create/drop table, and insert data into a table (/python/python_sqlite_connect_

SQLite 3 - B. Selecting, updating and deleting data (/python/python_sqlite_select_up

MongoDB with PyMongo I - Installing MongoDB ... (/python/MongoDB_PyMongo/py

Python HTTP Web Services - urllib, httplib2 (/python/python_http_web_servi

Web scraping with Selenium for checking domain availability (/python/python_Web_scraping_

REST API : Http Requests for Humans with Flask (/python/python-REST-API-Http-

Requests-for-Humans-with-Flask.php)

Blog app with Tornado
(/python/Tornado/Python_Torna

Multithreading ...
(/python/Multithread/python_mu

Python Network Programming I - Basic Server / Client : A Basics
(/python/python_network_progra

Python Network Programming I - Basic Server / Client : B File Transfer
(/python/python_network_progra

Python Network Programming II - Chat Server / Client
(/python/python_network_progra

Python Network Programming III - Echo Server using socketserver network framework
(/python/python_network_progra

Python Network Programming IV - Asynchronous Request Handling : ThreadingMixIn and ForkingMixIn
(/python/python_network_progra

Python Interview Questions I
(/python/python_interview_quest

Python Interview Questions II
(/python/python_interview_quest

Python Interview Questions III
(/python/python_interview_quest

Python Interview Questions IV
(/python/python_interview_quest

Python Interview Questions V
(/python/python_interview_quest

Image processing with Python image library Pillow
(/python/python_image_processi

Python and C++ with SIP

(/python/python_cpp_sip.php)

PyDev with Eclipse
(/python/pydev_eclipse_plugin_ir

Matplotlib
(/python/python_matplotlib.php)

Redis with Python
(/python/python_redis_with_pyth

NumPy array basics A
(/python/python_numpy_array_tu

NumPy Matrix and Linear Algebra
(/python/python_numpy_matrix_

Pandas with NumPy and Matplotlib
(/python/python_Pandas_NumPy

Celluar Automata
(/python/python_cellular_automa

Batch gradient descent algorithm
(/python/python_numpy_batch_g

Longest Common Substring Algorithm
(/python/python_longest_commo

Python Unit Test - TDD using unittest.TestCase class
(/python/python_unit_testing.php

Simple tool - Google page ranking by keywords
(/python/python_site_page_ranki

Google App Hello World
(/python/GoogleApp/python_Goo

Google App webapp2 and WSGI
(/python/GoogleApp/python_Goo

Uploading Google App Hello World
(/python/GoogleApp/python_Goo

Python 2 vs Python 3
(/python/python_differences_Pyt

virtualenv and virtualenvwrapper (/python/python_virtualenv_virtu

Uploading a big file to AWS S3 using boto module (/DevOps/AWS/aws_S3_uploadin

Scheduled stopping and starting an AWS instance (/DevOps/AWS/aws_stopping_sta

Cloudera CDH5 - Scheduled stopping and starting services (/Hadoop/BigData_hadoop_CDH5

Removing Cloud Files - Rackspace API with curl and subprocess (/python/python_Rackspace_API_

Checking if a process is running/hanging and stop/run a scheduled task on Windows (/python/python-Windows-Check-if-a-Process-is-Running-Hanging-Schtasks-Run-Stop.php)

Apache Spark 1.3 with PySpark (Spark Python API) Shell (/Hadoop/BigData_hadoop_Apac

Apache Spark 1.2 Streaming (/Hadoop/BigData_hadoop_Apac

bottle 0.12.7 - Fast and simple WSGI-micro framework for small web-applications ... (/python/Bottle/Python_Bottle_Fr

Flask app with Apache WSGI on Ubuntu14/CentOS7 ... (/python/Flask/Python_Flask_Blog

Selenium WebDriver (/python/python_Selenium_WebD

Fabric - streamlining the use of SSH for application deployment (/python/Fabric/python_Fabric.ph

Ansible Quick Preview - Setting up web servers with Nginx,

configure enviroments, and deploy an App (/DevOps/Ansible/Ansible_Settin

Neural Networks with backpropagation for XOR using one hidden layer (/python/python_Neural_Networ

NLP - NLTK (Natural Language Toolkit) ... (/python/NLTK/NLTK_install.php)

RabbitMQ(Message broker server) and Celery(Task queue) ... (/python/RabbitMQ_Celery/pytho

OpenCV3 and Matplotlib ... (/python/OpenCV_Python/pytho

Simple tool - Concatenating slides using FFmpeg ... (/FFMpeg/ffmpeg_fade_in_fade_c

iPython - Signal Processing with NumPy (/python/OpenCV_Python/pytho

iPython and Jupyter - Install Jupyter, iPython Notebook, drawing with Matplotlib, and publishing it to Github (/python/IPython/IPython_Jupyte

iPython and Jupyter Notebook with Embedded D3.js (/python/IPython/iPython_Jupyte

Downloading YouTube videos using youtube-dl embedded with Python (/VideoStreaming/YouTube/yout dl-embedding.php)

Machine Learning : scikit-learn ... (/python/scikit-learn/scikit_machine_learning_Su

Django 1.6/1.8 Web Framework ... (/python/Django/Python_Django_

CONTACT

BogoToBogo
contactus@bogotobogo.com (mailto:#)

FOLLOW BOGOTOBOGO

(https://www.facebook.com/KHongSanFrancisco) (https://twitter.com/KHongTwit) (https://plus.google.com/u/0/+KHongSanFrancisco/posts)

ABOUT US (/ABOUT_US.PHP)

contactus@bogotobogo.com (mailto:#)

Golden Gate Ave, San Francisco, CA 94115

Golden Gate Ave, San Francisco, CA 94115

Design: Web Master (http://www.bogotobogo.com)