Video Face Recognition with OpenCV

Adib Behjat
Cloud Computing Cass

Requirements

- System must have Python 2.7+ installed
- Install Python Dependencies (Step 1 -> Step 9):
  - **http://www.pyimagesearch.com/2015/06/22/install-opencv-3-0-and-python-2-7-on-ubuntu/**
- Accessible camera
- You're familiar with the concept of how faces are detected
- Mac/Linux Machine
  - Windows process isn't much different, but the listed installation steps are for Mac/Linux machines

About OpenCV

- OpenCV stands for Open Computer Vision
- OpenCV is a library of programming functions mainly aimed at real-time computer vision, originally developed by Intel's research center in Nizhny Novgorod, later supported by Willow Garage and now maintained by Itseez.
- Originally written in C++
- The library is cross-platform and free for use under the open-source BSD license.
  - Platforms include: Windows, Android, Linux/Mac
- API is available in the following languages (and more through community contributions):
  - C, C++, Python, Java, F

Step 1: Install OpenCV + Modules

- Open Terminal
- Create a working directory where OpenCV will ‘live’
  - `$ cd ~`
  - `$ mkdir opencv_dir`
  - `$ cd opencv_dir`
- Download OpenCV from Itseez’s GitHub account
  - `$ git clone https://github.com/Itseez/opencv.git`
- Download OpenCV’s Modules (includes face recognition algorithm)
  - `$ git clone https://github.com/Itseez/opencv_contrib.git`

Step 2: Config Build

- Since the source files are written in C++, they should be built by your computer in order for it to be accessible.
- Create a ‘build’ directory which will host the build files, and a module directory
  - `$ mkdir opencv_build`
  - `$ mkdir opencv_mods`
  - `$ cd opencv_build`
- Configure your build using make (basically tells the computer how you want it built)
  - `$ cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_INSTALL_PREFIX=/usr/local .. -DINSTALL_C_EXAMPLES=OFF`
- Specify additional cmake configurations to state where you want your modules to be saved
  - `$ cmake -DOPENCV_EXTRA_MODULES_PATH=opencv_contrib/modules open_cv`

Step 3: Build OpenCV + Modules

- Now that the cmake is configured, let's build this sweet thing
  - `$ make -j7`
  - `The 7 stands for running 7 jobs in parallel, better not use the computer while this is happening`

Step 4: Install Libraries

- After the build is complete, install the libraries from within the build directory
  - `$ sudo make install`

Step 5A: Take photos of yourself

- Before we dive into coding, let's dedicate some time to take pictures of ourselves

- **REQUIREMENTS:**

  - Use white background

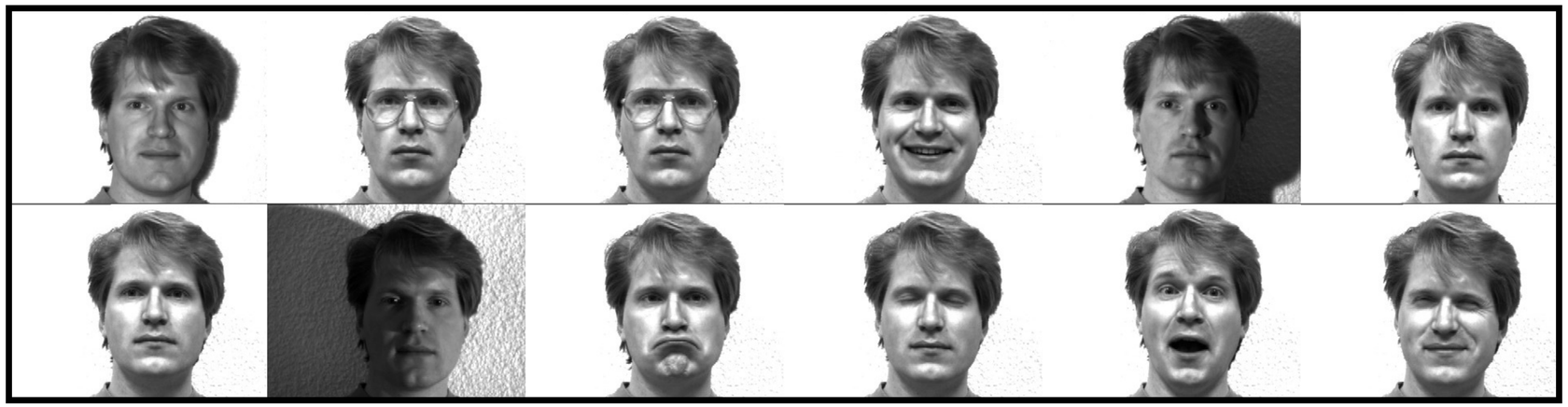
  - Be in a well lit room where the computer can identify your face easily

    - You may also benefit from taking pictures of your face under different light positions

  - Your head should be centered in the image

Step 5B: Take photos of yourself

- Take at least 20 pictures of yourself:
  - Each image should represent an emotion to enable the computer to identify the commonalities of your face under different circumstances.
  - Crop the images to a reasonable, square size (400x400px), and save them all as JPEG
  - Example of faces from Yale:



Step 5C: Take photos of yourself

- Name each file in the following manner
  - 01_<emotion>.jpg
  - example:
    - 01_happy.jpg
    - 01_sad.jpg
    - 01_surprised.jpg
  - Place all the images in a folder called “faces”

Summary - Folder Structure

```
opencv_dir/
├── faces/
│   ├── 01_sad.jpg
│   ├── 01_happy.jpg
│   ├── ...
│   └── 01_angry.jpg
├── opencv/
│   └── data/
│       └── haarcascade/
└── opencv/opencv_build
└── opencv/opencv_mods
```

Step 6: Download the Code

- Place the following code in the opencv_dir directory

https://github.com/adibbehjat/python_video_recognition/blob/master/video_detection.py

- Download the SVG image used for recognition

https://github.com/adibbehjat/python_video_recognition/blob/master/Admin_Square.svg

Step 7: Read the comments within the code

- The code provides comprehensive information of each step of the process and includes links to tutorials and sources I found for additional assistance.

```
# Provide the HaarCascades, or face template. Usually found in the OpenCV directory
# The frontal face HaarCascades template captures the face 'skin' region.
cascPath = "opencv-3.1.0/data/haarcascades/haarcascade_frontalface_default.xml"

# Hair and other elements are excluded. However, you're welcome to add more templates,
# such as for detecting eyes, ears, nose, objects, etc. More detection templated
# can be found in the Haarcascades directory

""" Request from the user two inputs """
# Tolerance is the padding for the box around the detected face. Otherwise, if padding is not provided,
# the box will 'stick' to close to the skin of the face.
tolerance = int(raw_input("Please specify the padding, ideally between 20 - 50 (int): "))

# Confidence level outputted by OpenCV is a value between 100 - 0. 100 means that there is absolutely
# no similarity/equality between the detected face and recognized face, 0 means that there is an
# exact match between the recognized face and detected face.
confidence_level = int(raw_input("Please specify the required confidence, ideally <40 (int): "))

# Facial Mapper
faceCascade = cv2.CascadeClassifier(cascPath)

```

Summary - Folder Structure

```
opencv_dir/
├── faces/
│   ├── 01_sad.jpg
│   ├── 01_happy.jpg
│   ├── ...
│   └── 01_angry.jpg
├── opencv/
│   └── data/
│       └── haarcascade/
└── video_detection.py
└── Admin_Square.svg
└── opencv/opencv_build
└── opencv/opencv_mods
```

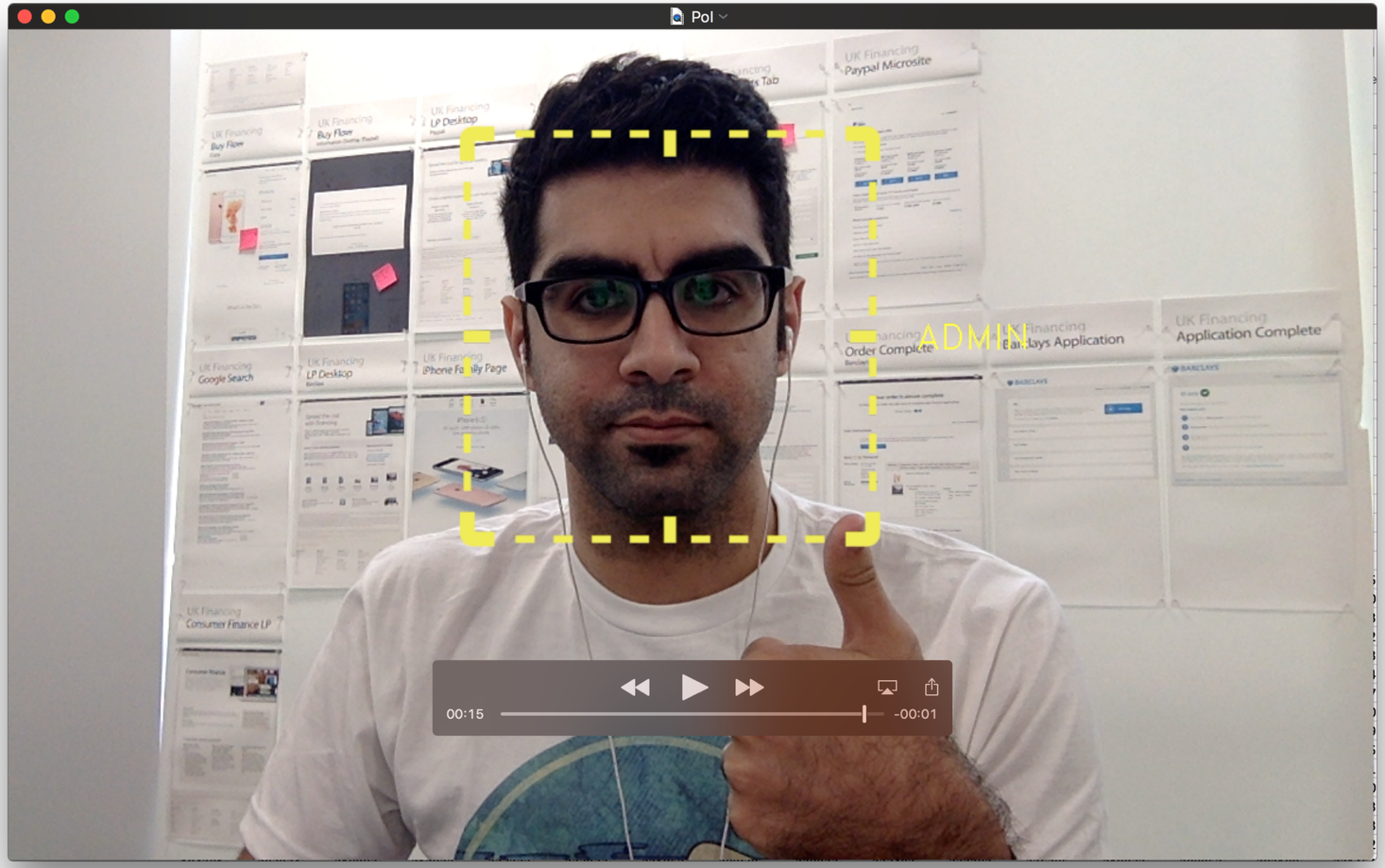
Step 8: Run the video recognition command

- `open_cv$ python video_detection.py`
- You will be prompted with two information:

```
                                                $ python video_detection.py
Please specify the padding, ideally between 20 - 50 (int): 50
Please specify the required confidence, ideally <40 (int): 35
```

- Padding provides padding around your face and the box
- Confidence is based on accuracy of recognition
    - **0 = The face is exactly alike the saved, recognized face**
    - **100 = The detected face is absolutely not the saved, recognized face**

Step 9: Enjoy!


PoI
UK Financing
Paypal Microsite
UK Financing
Buy Flow
UK Financing
LP Desktop
UK Financing
Google Search
UK Financing
LP Desktop
UK Financing
Consumer Finance LP
ADMIN
UK Financing
Application Complete
00:15
-00:01