Face Swapping in Photographs

SCC-0251 - Image Processing

Prof. Moacir Antonelli Ponti

Universidade de São Paulo

Instituto de Ciências Matemáticas e Computação

Members

Group 6
Hugo Moraes Dzin - 8532186
Matheus Gomes da Silva Horta - 8532321
Raul Zaninetti Rosa - 8517310

1. Main Objetive

The main objective of this project is to find faces in a picture and replace them. Each recognized face will be compared against a database of professors of ICMC, and the one with the closest matching facial traits will replace the face found in the picture. It will also be possible to swap around multiple detected faces in the same picture.

2. Input Images

Our project will deal with frontal, standing-up pictures of human faces. It may be able to process slightly rotated faces, but anything too close to lying down, or even upside-down, probably won't be recognized correctly. The faces also need to be facing the camera, which means that profile pictures are not expected to work.

Furthermore, the images can't be too small, otherwise there wouldn't be enough detail for analysis. We estimate that a face needs to be at least 50 pixels tall and wide to be usable. Two other factors are also important: we won't be dealing with low or weird lighting conditions, neither partially obstructed faces, as shown in the examples below.

Fig. 1: Valid Picture
Source: Led Zeppelin Wikia

Fig. 2: Obstructed Face
Source: Can Stock Photo

Fig. 3: Weird lighting conditions
Source: Pinterest

As described before, the input images will be compared against a dataset of photos of ICMC (Instituto de Ciências Matemática e Computação) professors, downloaded from the institute's official website (http://icmc.usp.br). Most of these pictures are well lit, are facing the camera properly and have an acceptable resolution, and therefore have a very high chance of being recognized correctly.

3. Methods

3.1 - Recognizing Faces and Analyzing Facial Traits

To identify faces and their separate parts, we will use Haar Feature-based Cascade Classifiers. Haar Feature-based Cascade Classifiers is an object detection method proposed by Paul Viola and Michael Jones in 2001. It is a method that uses learned machine where

the cascade function is trained from a lot of positive and negative images. In fact, we will not implement this method. OpenCV already offers us some trained XML-files that are sufficient for our purpose. In addition to XML files, opencv also offers some methods for manipulating these XML-files and also methods to apply the knowledge of these XML-files in our images.

The example below shows the output of a code that recognizes faces and eyes and gives us as output the input image with rectangles in the detected parts:

Fig. 5: Input Picture
Source: ICMC Website

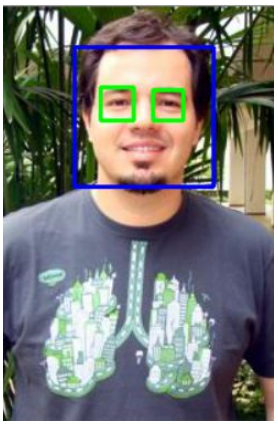


Fig. 6: Output with face and eye recognition
Source: ICMC Website

3.2 - Face Comparison

Three methods for comparing faces are used: distance between the eyes, nose width and SIFT method. Each method has an assigned score and at the end this score is added and, based on it, the best possible combination of faces is decided. The distance between eyes and nose width methods have an equal potency of 30 and the SIFT method has a score of 40. Each method returns the 5 best combinations of teachers given a given input. From this the sum of the scores is done and returned to the face of the teacher who has the best ranking. If there is a tie, an arbitrary choice is made between the faces.

3.2.1 - Distance between the eyes

Fig. 7: distance d between the eyes
Source: ICMC Website

The distance between the eyes is calculated by the distance between the points of the center of the eyes. If the recognition method does not find exactly two eyes on the face, the method is not calculated for that specific teacher and the function returns null.

3.2.2 - Nose width

Fig. 8: width w of the nose
Source: ICMC Website

The same is true in comparing the width of the nose. If the recognition method does not find a nose in the image, the method is not calculated for that specific teacher.

3.3.3 - SIFT

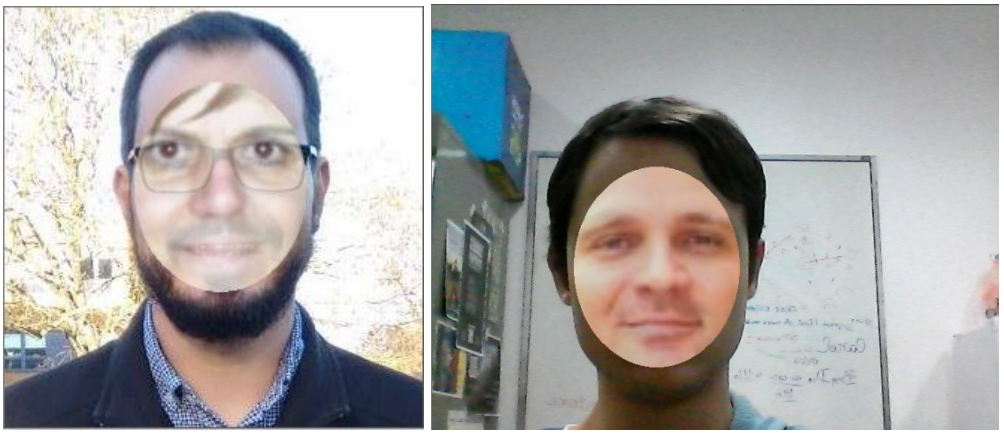
SIFT is a computer vision algorithm published by David Lowe in 1999 (Lowe, 1999) and patented in the USA by the University of British Columbia. SIFT is composed of two distinct parts: the detector and the descriptor. The SIFT detector is based on Gaussian difference calculations and the SIFT descriptor uses histograms of oriented gradients to describe the local neighborhood of the points of interest. The following description of both parties is based (Lowe, 2004).

3.3 - Face Swapping

The following procedure will be performed:

1. The face recognizer will return the coordinate of an axis-aligned rectangle for each face.
2. The replacement face will be resized to the recognized face's dimensions
3. An elliptic region will be used to crop the replacement face and paste into the recognized face, using an alpha blending algorithm of factor 0.75.

4. Results and Conclusion



The results were a little insatisfactory. The recognition would frequently return the same results. This may be caused by the fact that faces have an overall similar shape and SIFT is not a good method to use in this situation, and the position and dimensions of features found by Haar cascade are not precise enough for direct comparison.

5. Running the Programs

5.1 Necessary packages

This project needs the OpenCV contrib package, due to the use of the SIFT algorithm. The Windows package can be downloaded from:

http://www.lfd.uci.edu/~gohlke/pythonlibs/

The version used was opencv_python-3.2.0+contrib-cp36-cp36m-win32.whl, meant for Python 3.5 32-bit.

5.3 Running the Program

The command-line arguments are as follows:

```
py main.py [-f FILENAME] [-d DIRECTORY]
```

FILENAME: Path of the input image. If not specified, real-time input from the computer's recording device is used instead. When using the webcam, press q to quit.
DIRECTORY: Directory containing the replacement faces. Defaults to **`./professors`** if not specified. The included directory **`./professors_all`** can also be used, but startup times will become significantly longer.