2016

CIS581 Final Project Report

Sai Krishnan Chandrasekar

Sakthivel Sivaraman

12/23/2016

OVERVIEW

The goal of this project is automatic detection and replacement of faces in videos. Given a test video, our code will automatically replace a target face.

We have decided to go with “Option 2” for this project.

- We decided to code using Python.
- For image processing functions, we’ve used OpenCV. OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.
- For facial detection algorithms and face landmarks, we’ve used “DLib”. Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. It has a Python API that enables us to use it in our python code.

METHODOLOGY

The overall methodology of the project has been outlined below:

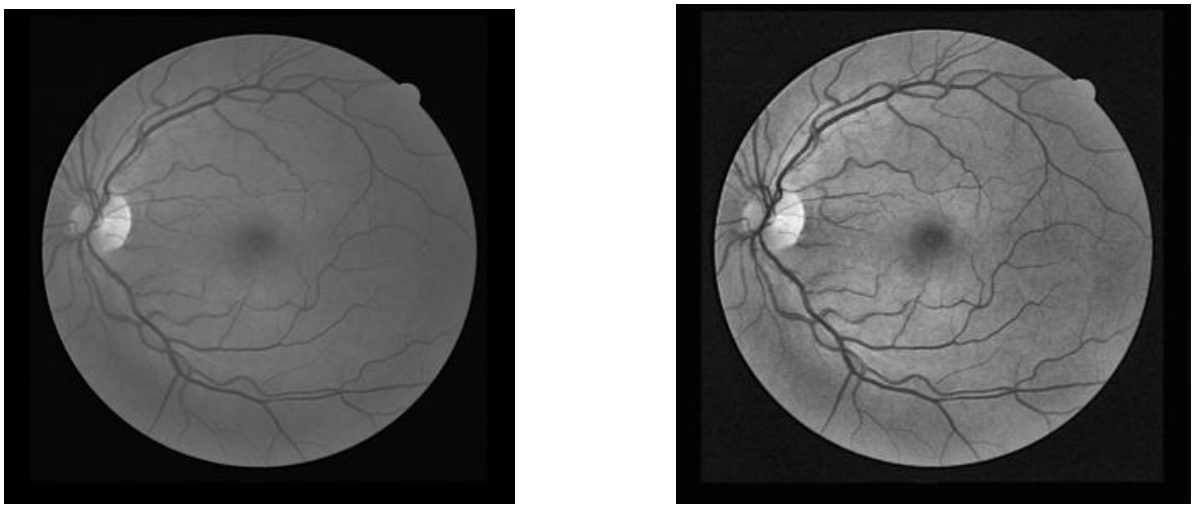
- Contrast Limited Adaptive Histogram Equalization (CLAHE).
- Automatic multiple face detection in videos.
- Detecting facial landmarks on target face(s) (Faces in the videos).
- Detecting facial landmarks on replacement face (Faces in photos).
- Find convex hull for all the target face(s) and the replacement face.
- Warp the replacement face onto the target face(s) using Delaunay triangulation.
- Blend seamlessly.

In the following pages, we've described the methods we used to implement the steps described above.

CONTRAST LIMITED ADAPTIVE HISTOGRAM EQUILIZATION

We used CLAHE to equalize contrast in the images. Regular histogram equalization considers the global contrast of the image. In many cases, it is not a good idea. Instead, we used CLAHE. In this, image is divided into small blocks called "tiles" (tileSize is 8x8 by default in OpenCV). Then each of these blocks are histogram equalized as usual. So in a small area, histogram would confine to a small region (unless there is noise). If noise is there, it will be amplified. To avoid this, **contrast limiting** is applied. If any histogram bin is above the specified contrast limit (by default 40 in OpenCV), those pixels are clipped and distributed uniformly to other bins before applying histogram equalization. After equalization, to remove artifacts in tile borders, bilinear interpolation is applied.

We did this to compensate for change in illumination. This improved the results on videos “Hard 3” and “Hard 1” significantly.



CLAHE Before and After

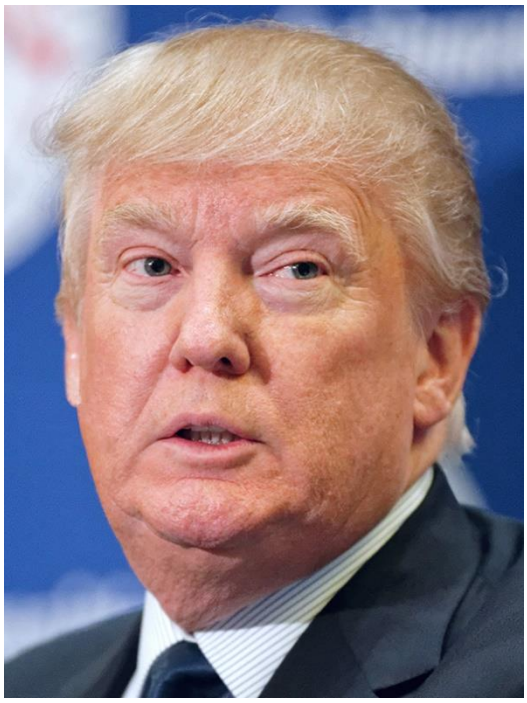
FACE AND FACIAL LANDMARK DETECTION

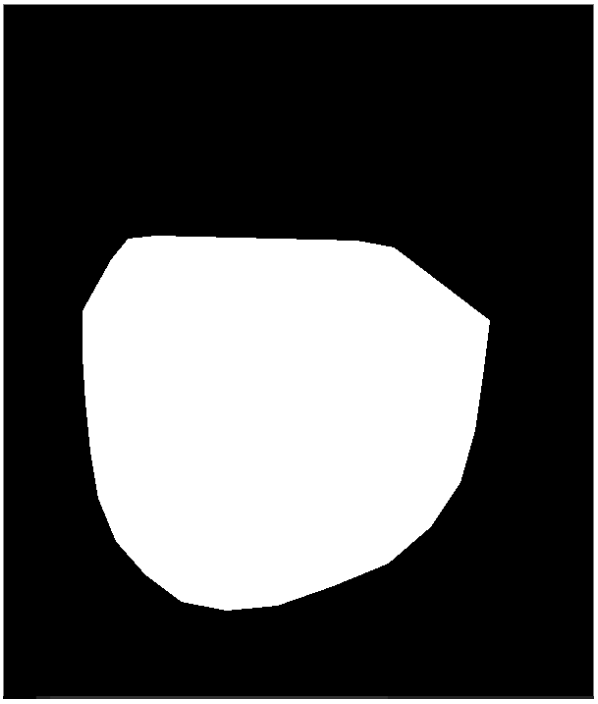
We used Dlib to detect faces and landmarks on the faces.

- Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. It has a Python API that enables us to use it in our python code.
- Dlib implemented the paper “One Millisecond Face Alignment with an Ensemble of Regression Trees by Vahid Kazemi and Josephine Sullivan, CVPR 2014”.
- Dlib returns 68 facial landmarks such as corners of eyes, nose tip, lips, chin etc.

CONVEX HULL

We calculated the convex hull of the facial landmark points on the target face. We created a mask using the convex hull by making the points inside the hull white. A sample mask of Donald Trump's face is shown below. Not to scale.



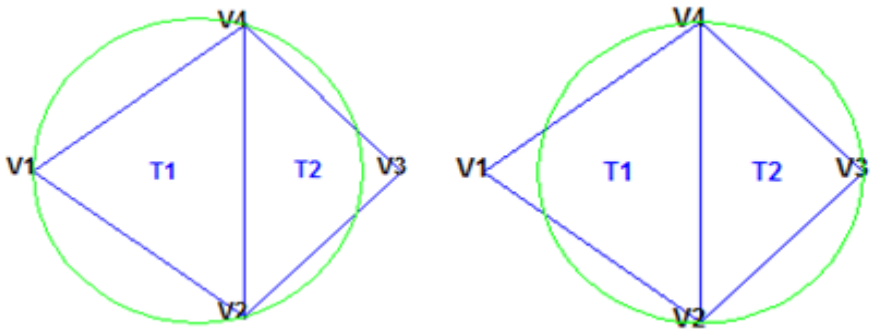


We used OpenCV's cv2.ConvexHull() function to achieve this.

REPLACEMENT

Delaunay Triangulation.

Delaunay triangle computes triangles such that no point in the set of control points is within the circumcircle of any triangle.



Delaunay Triangulation was implemented using opensource code from learnopencv.com.

Affine transformation

OpenCV's cv2.getAffineTransformation() function finds the affine transformation given a pair of triangles. cv2.warpAffine applies the affine transformation to the source image.

Seamless blending

Blending was done to blend the edges of the replacement face onto the target face.

Seamless blending

MOTION COMPENSATION

When we followed the aforementioned steps, we achieved successful face replacement. However, the face wasn't too stable and had a jitter to it. To reduce the jitter, we implemented motion compensation.

We smoothed the images using the formula:

landmark_Smooth = w*landmarks + (1-w)*landmark_Smooth

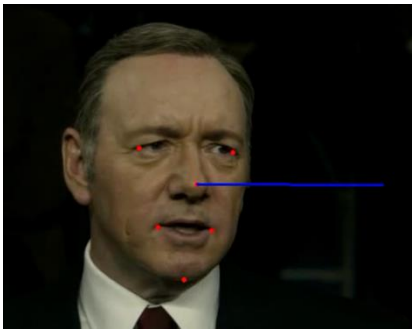
This improved the stability of the faces considerably, as will be evident in the videos.

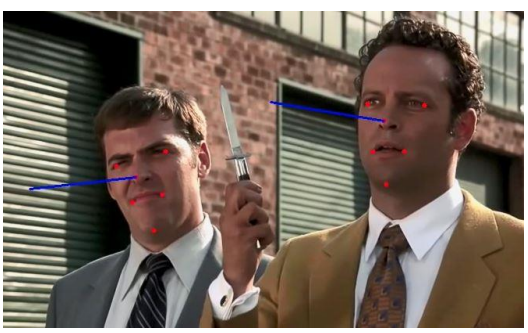
HEAD POSE ESTIMATION

We obtained orientation of the faces by the follow method.

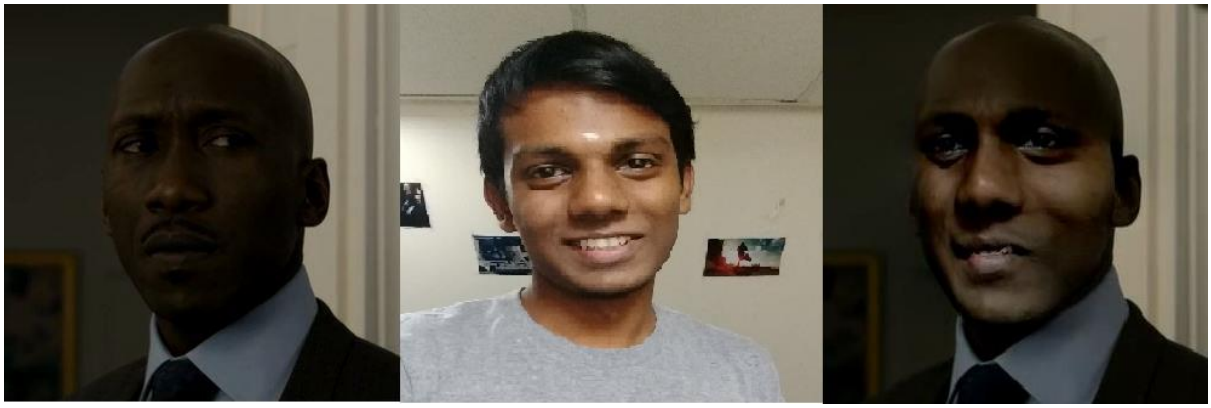
- We found out which landmarks of Dlib output correspond to nose tip, eye corners, lip corners and chin.
- We used a default 3D model points that approximates the 3D model shape of the face.
- We used cv2.solvePnP. This function estimates object "pose" from the 3D-2D point correspondences.

We followed the tutorial on learnopencv.com to achieve this.





SNAPSHOTS!



CHALLENGES FACED

1. Installing DLib and the dependencies.
2. Handling and processing videos in python.
3. Modifying from single face replacement to multiple face replacement.
4. Handling face warping issues.

INTERESTING OBSERVATIONS

1. Warping had issues if the person had glasses on.
2. Detection had issues if the person had a beard.
3. Detection failed if the entire face wasn't in the frame.
4. For best results, we required a face that was looking directly at the camera.
5. Too much disparity in skin tone did not produce great results.

Points 2, 3, 4 and 5 might be rectified by training our own classifier with a wider range of faces.

REFERENCES

1. www.opencv.org – Most of our code has been adapted from their open source library. Their documentation helped us greatly as well.
2. dlib.net – We used their open source package and landmarks file for face detection and facial landmark detection.
3. www.learnopencv.org – We referred to the website learnopencv.org for their helpful tutorials and examples on computer vision using OpenCV. We used their open source code for Delaunay Triangulation, warping and head pose estimation. We did not use head pose anywhere though.