FaceSwap

John Kanu
PhD Student, Computer Science, UMD
Email: jdkanu@cs.umd.edu

Srinidhi Sreenath
Masters Student, Robotics, UMD
Email: ssreenat@terpmail.umd.edu

***Abstract*—In this paper we present and compare several methods for swapping faces between images and videos.**

I. Introduction

In this paper we discuss several approaches to swapping faces between images and videos. We present a triangulation method using Barycentric coordinates, a method using Thin Plate splines, and a deep learning approach. Methodology, experiments, results, and analysis are included below.

II. Traditional Approach

III. Facial Landmarks Detection

A. Model pre-trained on iBUG 300-W

At the base of our pipeline is a model that detects the facial landmarks in an image. The model is pre-trained on the iBUG 300-W face landmark dataset from the 300 Faces-In-The-Wild Challenge [1]. The model available in the `dlib` library. This module is agnostic to the particular domain of the image, whether it is a single standalone image or a frame in a video. Thus, we apply it at our discretion to produce facial landmarks in various locations in our pipeline. Given an image, the model outputs a boundix box rectangle of the face along with the (x, y) locations of each of the 68 fiducials. Below is an example output.

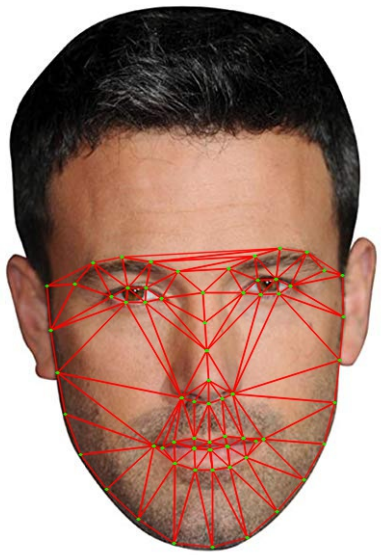


Fig. 1: Example Delaunay triangulation

IV. Warping

A. Triangulation

Given source image I_A and destination image I_B, the task is to map the RGB color values at locations on the face in I_A to the corresponding locations in the face in I_B. Without information on the 3-dimensional information on the geometry of the face, we use the method of triangulation to map pixel values from I_A to I_B. Triangulation subdivides the 2-dimensional image into triangles that cover the face, resembling the projection of a 3-D polygonal approximation of the true surface of the face in 3 dimensions onto the image plane. Our method generates isomorphic triangulations of the faces in I_A and I_B, such that triangles in each image can be paired together and pixels in I_B are mapped to a unique (x, y) position in the corresponding triangle in I_A, so that the color intensity values can be mapped between the images. The triangulation algorithm we employ is Delaunay triangulation.

1) Isomorphic Triangulations: In our implementation, we require that the graphs are isomorphic. Using Delaunay triangulation alone, there is no guarantee that the two graphs produced by the triangulation will be isomorphic. Figure 2 demonstrates this. In order to enforce isomorphism, we compute a weighted average of the fiducials in I_A and I_B and generate a triangulation on this third set of points, which is a graph that can be mapped to the individual sets of fiducials in I_A and I_B.

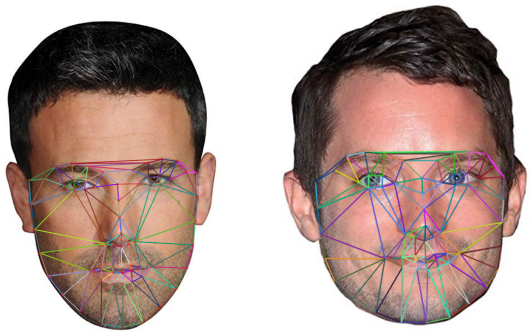


Fig. 2: Demonstration of non-isomorphism of Delaunay triangulation on different sets of fiducials. Notice the edge in the forehead points in two different directions between the two images.

2) Optimal Triangulations: After mapping the graphs from the third set of keypoints to each original set, the triangles may overlap, as the vertices may cross edges in the image plane. In order to minimize this undesirable phenomenon, we search the parameter space of the weighted average to minimize overlap, computed using a geometry library.

Given landmarks L_A and L_B we compute the third set of landmarks as

$$L_r := r \cdot L_A + (1 - r) \cdot L_B$$

Denote the edges found by Delaunay triangulation on L_r as E_r, and the graph created by mapping edges back to L_A and L_B as E_r^A and E_r^B, respectively. Let the function $\texttt{Overlap}(E)$ compute the total area of overlap between all pairs of triangles created by edges E. The weight for the optimal triangulation for minimizing artifacts is then computed as

$$r^* = \arg\min_r \left(\texttt{Overlap}(E_r^A) + \texttt{Overlap}(E_r^B)\right)$$

This selection results in isomorphic triangulation graphs that are of acceptable quality, as shown in Figure 3.

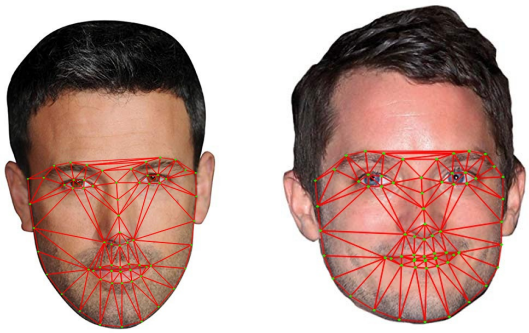


Fig. 3: Optimal triangulations for two sample images mapped to the original images. Notice that the graphs are now isomorphic.

B. Warping using Barycentric coordinates

Mapping the color intensity values between pixels in corresponding triangles T_A and T_B is performed by the following procedure:

1) Compute the Barycentric coordinates α, β, γ for any pixel in T_B
2) Compute the position of the corresponding position in T_A
3) Interpolate the color intensity value at this location in I_A
4) Copy the color value to the location in I_B

The condition for detecting whether a pixel is inside a triangle is given by the following formula:

$$\alpha \in [0, 1], \beta \in [0, 1], \gamma \in [0, 1]$$

$$\alpha + \beta + \gamma = 1$$

Artifacts are minimized by adjusting boundaries of intervals with small epsilon values, given by

$$\alpha \in [-\epsilon, 1 + \epsilon], \beta \in [-\epsilon, 1 + \epsilon], \gamma \in [-\epsilon, 1 + \epsilon]$$

$$1 - \epsilon \leq \alpha + \beta + \gamma \leq 1 + \epsilon$$

Figure 4 shows an example output for this method.

C. Warping using Thin Plate Spline

Triangulation assumes that we are doing affine transformation on each triangle. To better represent the human face, transformation using **Thin Plate Splines (TPS)** is used to model arbitrarily complex shapes.

The TPS in this context maps from the feature points of the target image (B) to the source image (A). 2 splines are fit, one for x direction and another for y direction. (x, y) represent the coordinates of feature points in B, and (u, v) represents the coordinates of feature points in B

- The first spline maps from x coordinates of the feature points in B, to feature points in A. The spline equation is as follows:

$$u = f_x(x, y) = a_1 + (a_x)x + (a_y)y + \sum_{i=1}^{p} w_i U(\|(x_i, y_i) - (x, y)\|_2)$$

- The second spline maps from y coordinates of the feature points in B, to feature points in A. The spline equation is as follows:

$$v = f_y(x, y) = a_1 + (a_x)x + (a_y)y + \sum_{i=1}^{p} w_i U(\|(x_i, y_i) - (x, y)\|_2)$$

where p represents the total number of landmarks in target image B and

$$U(r) = r^2 \log_e r^2$$

The pipeline of TPS is as follows:

- The parameters of the TPS i.e a_1, a_x, a_y and the $w_i's$ for $i \in [0, p]$ are calculated for each of the two splines.
- Once the parameters are obtained, for each (x, y) in image B, the corresponding pixel position in image A i.e (u, v) is obtained using the thin plate spline equations.
- λ is a parameter to be set while estimating the parameters. We kept the vale of $\lambda = e^{-4}$ for the best result of TPS. Other values such as $\lambda = e^{-8}$ and $\lambda = e^{-16}$ were also tried and it gave suboptimal results.
- The pixel information at (u, v) of A is filled into the (x, y) position in B.

Figures 4 and 5 show the outputs for both methods.

Fig. 4: A comparison of warps using Barycentric coordinates (left) and Thin Plate Spline (right).

Fig. 5: A comparison of final blended outputs using Barycentric coordinates (left) and Thin Plate Spline (right).

D. Replace Face

Replacing the face was as simple as writing the values as defined in the procedures above. No further processing was performed in order to correctly insert the values of the pixels. Once the required pixels in target image B are filled with information from image A, the faces are essentially warped. A sample of the warped face is shown below. But, since the direct pixel replacement doesn't look natural due to differences in skin tones, the warped image has to blended to fit the tone of the original target image.

Fig. 6: Cristiano Ronaldo onto Toni Kroos using Thin Plate Spline method

Fig. 7: Jimmy Fallon onto Paul Rudd

E. Blending

In order to accomplish this, poisson blending techniques can be used. The original image's texture is applied to the warped image to clone the swapped face. The built-in OpenCV function *seamlessClone* is used with NORMAL_ CLONE. A mask of the face region of the target region that is replaced is to be given as an input as well as the center of the blended result.

- For the mask, a convex hull of the feature landmarks for target image is obtained and the binary image of the convex hull (1 inside hull and 0 otherwise) is the mask needed.
- The centroid of the convex hull (also the center of the bounding rectangle of the convex hull) is the center where the blended face is to be placed on the original image.

Here is a demonstration of the performance of the method on very different images.

F. Motion Filtering

Motion filtering is achieved by first generating a sequence of predictions for the locations of the 68 keypoints for both faces in all frames of a video. Due to inherent error in the landmark models, after generating the predictions, there is no guarantee that the positions will resemble continuous transformations of the head in 3-D space, so invoking the warping and blending on only this sequence will appear unstable and shaky when viewed on videos of common frame rates (around 30 fps). Thus, in order to maximize the aesthetic appeal of the video after generating landmarks, warping, and blending, we first perform a smoothing step on the sequence of landmarks, so that flickering is minimized.

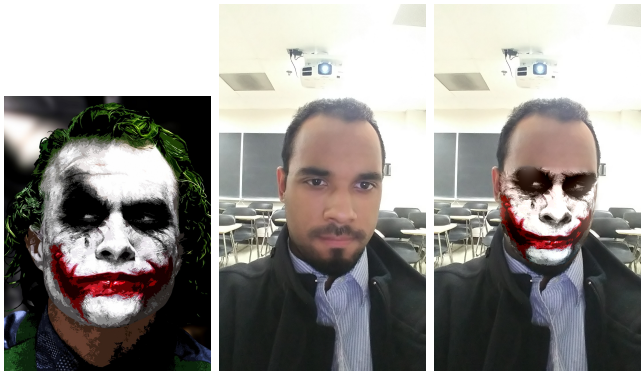


Fig. 8: Playful demonstration of warping using Barycentric coordinates. Left: target image, Center: input image, Right: output

We created our own smoothing method to minimize the amount of flickering. Given N frames, a sequence of landmark positions

$$\{L_i\}_{i=1,...,N}$$

over the N frames, and a weight parameter w, we compute the smoothed value as

$$S_i := \begin{cases} L_1, & \text{for } i = 1 \\ w \cdot L_i + (1-w) \cdot L_{i-1}, & \text{for } i = 2, ..., N \end{cases}$$

We found the value $w = 0.5$ achieves appealing results.

V. Deep Learning Approach

Joint 3D Face Reconstruction and Dense Alignment with Position Map Regression Network (PRN). PRN is a method to jointly regress dense alignment and 3D face shape in an end-to-end manner. This method can be used to swap 2 faces or obtain fiducial features when faces are detected in an image.

The PRNet trained model was downloaded and was used in the detection of fiducial features. The codebase was modified for 3 functionalities, each one described below.

A. Obtain fiducial features from the model

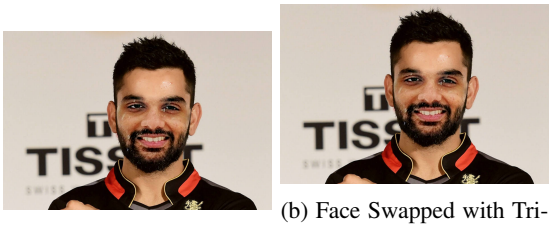
The fiducial features were obtained using the deep learning network. The module uses dlib library for detecting faces and then positions of facial features are obtained from the network. Then these points are used to obtain the 68 facial landmarks. A sample of the fiducial features obtained from the model is shown below:

Fig. 9: Fiducial features from PRNet

Once the 68 landmark points are obtained, then the 68 points are used in the traditional pipeline to swap the faces in the desired images. The traditional pipeline uses Thin Plate Spline (TPS) method to swap the faces in this case.

(a) Source Image. (b) Target Image

(a) Face Swapped with TPS (b) Face Swapped with Triangulation



(a) Face Swapped with TPS (b) Face Swapped with Triangulation

B. Directly obtain swapped faces

The codebase for the PRNet also has an option to directly obtain the swapped faces when 2 images are given as input.

This method used the texture obtained from the source image to perform poisson blending on the target image. A sample output is shown below:

Fig. 13: Directly swapped texture image

This method is used when a video with frames containing a single target face needed to be swapped with a static source image. The Data1OutputPRNet.mp4 result is obtained by this methodology.

C. Obtain fiducial features for multiple faces

The original codebase for the PRNet uses dlib for detecting faces and using those faces to detect fiducial features from the network. However, when multiple faces are detected in a single image, then only the first and largest face detected is considered and the other images are ignored.

The code was modified in such a way that when multiple faces are detected, the network is used to obtain the fiducial features for the 2 largest faces.

Once the facial landmarks are obtained for the 2 largest faces in a single image, then the traditional pipeline is used to swap the faces. The traditional pipeline uses Thin Plate Spline (TPS) method to swap the faces in this case. The Data2OutputPRNet.mp4 result is obtained by this methodology. A screengrab of the video is shown here:

Fig. 14: Fiducial features from PRNet

D. Outputs for Data1

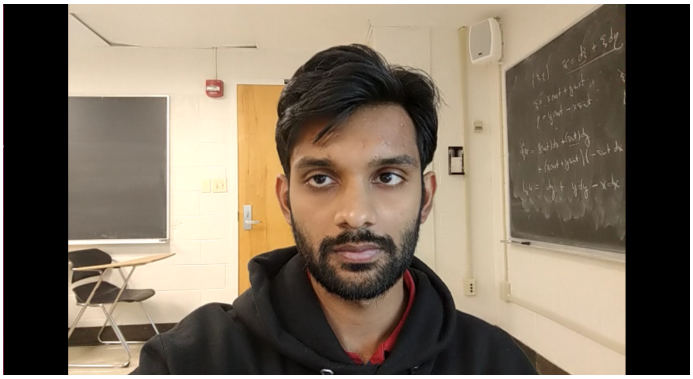


Fig. 15: Input image for one of the frames

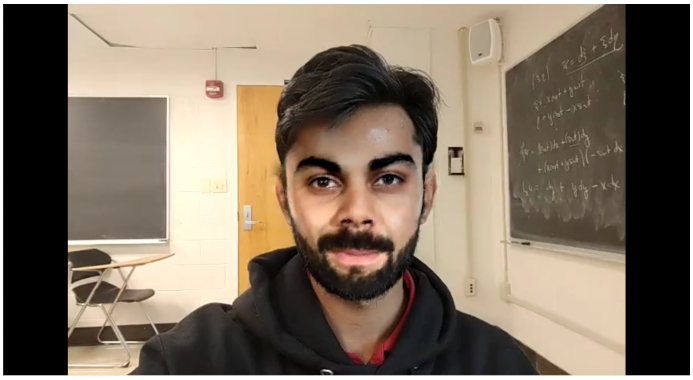


Fig. 16: Output image for one of the frames, using PRNet

Fig. 17: Output image for one of the frames, using TPS

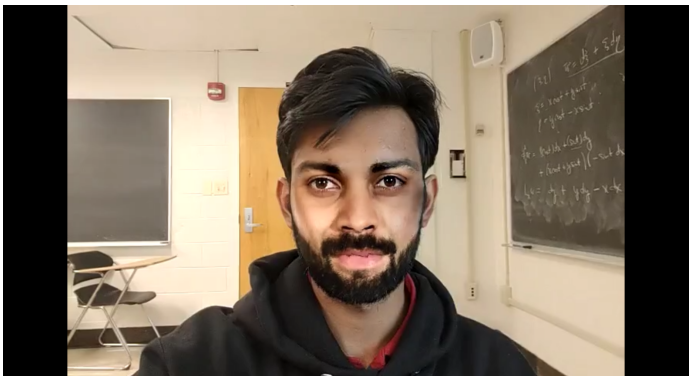


Fig. 18: Output image for one of the frames, using Triangulation

E. Outputs for Data2

Fig. 19: Input image for one of the frames

Fig. 20: Output image for one of the frames, using PRNet

Fig. 21: Output image for one of the frames, using TPS

Fig. 22: Output image for one of the frames, using Triangulation

VI. Error Analysis

A. Error Type 1: Smile artifact

When the person in the source image is not smiling, and the person in the target image is smiling, the colors of the pixels mapped into the mouth region in the destination image do not have the appearance of teeth, which one would expect to see in between the keypoints of an open mouth. Figures 23 and 24 demonstrate this error case.

B. Error Type 2: Light properties mismatch

When the lighting between the two images are widely different, there is a high risk of qualitative error in the blending procedure. In Figure 25, we can observe that the face in the target image has a different surface lighting pattern than the face in the output image. Here, it appears that the blending algorithm could have done a better job at matching the lighting of the surface of the face to the new 3-dimensional scene.

C. Error Type 3: Large difference in perspective

When the perspectives of the two images are very different, the direction of the faces point in very different directions. The two interpolation algorithms we tried are incapable of mapping small surfaces to large surfaces, as there is a paucity of information present at the low level. Whereas a human creating this warping may use high-level information and avoid pixelation of the surface, the Barycentric coordinate and Thine Plate Spline methods fail to generate a visually appealing result.

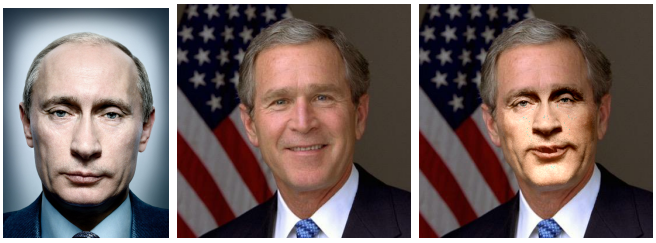


Fig. 23: Vladimir Putin onto George W. Bush

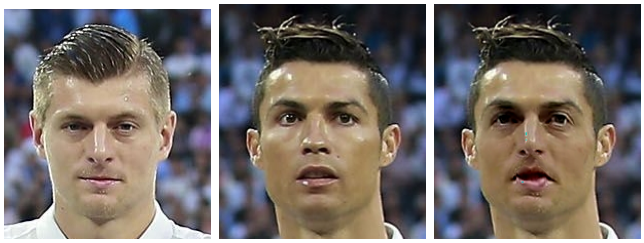


Fig. 24: Toni Kroos onto Cristiano Ronaldo

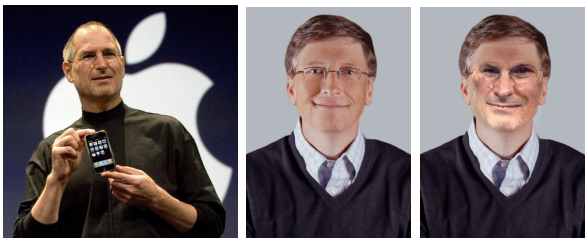


Fig. 25: Steve Jobs onto Bill Gates

Fig. 26: Rowan Atkinson onto Jeremy Clarkson

References

[1] C. Sagonas, E. Antonakos, G, Tzimiropoulos, S. Zafeiriou, M. Pantic. *300 faces In-the-wild challenge: Database and results.*, 3rd ed. Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation "In-The-Wild". 2016.