Project 2 : Face Swap
CMSC733

Abhishek Nalawade
Master of Engineering in Robotics
University of Maryland, College Park, 20740
abhi1793@umd.edu

Aditya Jadhav
Master of Engineering in Robotics
University of Maryland, College Park, 20740
amjadhav@umd.edu

***Abstract*—The project aims to swap two given images or frames using Traditional and Deep Learning Approach. The report contains a detailed solution to Phase 1 and implementation details of Phase 2 of the Faceswap task. Phase - 1 pipeline consists of facial landmark detection, face warping using Delaunay Triagulation and Thin Plate Spline, and blending and the Phase - 2 makes use of a Deep-Learning approach for faceswap named PRNet. The PRNet method used is illustrated in [1].**

I. Phase - 1 : Traditional Approach

In this section we present the Traditional Approach to swap faces in two given images or frames which is implemented in five steps listed below,

- Facial Landmark Detection
- Inverse Face Warping using Delaunay Triagulation
- Inverse Face Warping using Thin Plate Splines
- Blending to get the result

A. Facial Landmark Detection

The traditional approach to face Swap starts with finding important points on the face called the Facial Landmarks. For this purpose we use the OpenCV Dlib Library which is a facial landmark detector that can be used to detect facial landmarks in real-time with high quality predictions. The Dlib library uses a pre-trained model estimate the location of 68 (x, y) - coordinates that map to facial structures on the face. A gray-scale image fed to the Dlib functions produce a list of 68 points each corresponding to a unique facial landmark. This list provides with correspondences for both images which are critical for the pipeline ahead. Following Facial Landmarks results are obtained with Dlib:

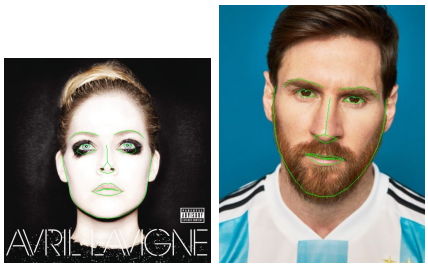

Fig. 1. Facial Landmarks

B. Delaunay Triagulation and Warping

In this step, we utilize the facial landmarks obtained from the previous step to warp the faces onto one another. Ideally we require the 3D information of both faces to warp them efficiently, but we only have 2D information in the traditional approach. Using the facial landmarks we perform Delaunay Triangulation which quickly and efficiently divides the face into triangular areas. Delaunay Triangulation is obtained by drawing the dual of the Voronoi diagram, i.e., connecting each two neighboring sites in the Voronoi diagram and can be constructed in O(nlogn) time. We want the triangulation to be consistent with the image boundary such that texture regions won't fade into the background while warping. Delaunay Triangulation tries the maximize the smallest angle in each triangle. We implement Delaunay Traingulation with the help of OpenCV methods Subdiv2D() and getTriangleList(). Below are some of the Triangulation results:

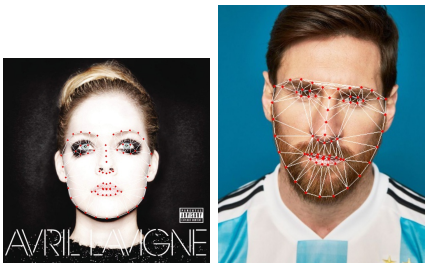

Fig. 2. Delaunay Triangulation

The methods mentioned before provided us with the corresponding triangles in source and destination images. With the help of The Barycentric Coordinate System , we compute the Barycentric coordinates for each pixel of every triangle in the destination face. Now that we have the Barycentric Coordinates of all pixels, we check for the points that lie inside the destination image with the contraint:

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

and

$$\alpha + \beta + \gamma \in (0,1]$$

Next we compute the corresponding pixel position in the source image using the same Barycentric equation shown

in the last step. And finally, we copy back the values of corresponding pixel coordinates in the source image to the destination image using Bilinear Interpolation.

Fig. 3. Delaunay Custom Result

C. Thin Plate Splines and Warping

Using the Delaunay Triangulation and Barycentric Coordinates is not the best method to warp a 3D human face with 2D information with its complexity. There are quite a few techniques out there that provide a smooth interpolation between a set of control points. Thin Plate Splines is one such technique. It interpolates a surface that passes through each control point. A set of 3 points will thus generate a flat plane. Performing Thin Plate Splines is analogous to bending a sheet of metal. For our case, we computed X-coordinate spline and a Y-coordinate spline which maps the feature points from the destination image to the corresponding feature points in the source image.

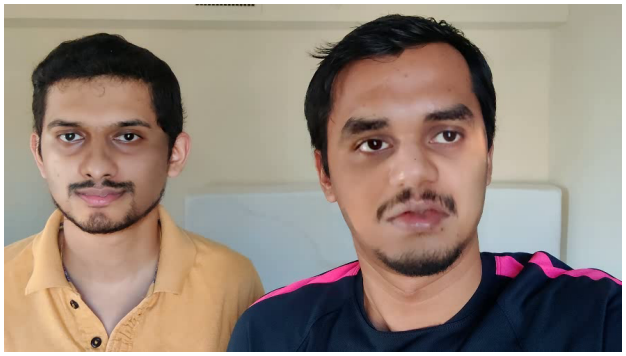

Fig. 4. TPS Custom Result

D. Blending

Replacing the faces with the two previous steps produces a result that has swapped faces but with largely inconsistent texture. This is where Blending the two faces together steps in the pipeline. For this purpose we used the seamlessClone() provided by OpenCV which uses Poison Blending. Figures show the results of this step:

II. Phase - 2 : Deep Learning Approach

The Deep Learning approach uses a pre-trained supervised encoder-decoder model to obtain face fiducials. The Position map Regression Network method illustrated in [1] is a method to jointly regress dense alignment and 3D face shape in an end-to-end manner. It generates full 3D meshes of faces and the correspondence of these meshes from the given image. The PRNet uses residual blocks and Convolutional Neural Network in its architecture to encode an image to 8x8x512 feature maps and again decoding it back into a 256x256x3 image. The algorithm is able to provides good face swap results because it uses the full 3D mesh of the face masks. We used the paper's GitHub link along with another GitHub resource to implement this phase.

A. Results

PRNet performs very well for the face swap between rambo and the Test1 video frames. The algorithm also works nicely on a custom video with our faces, where the performance still deteriorates when the face orientations are not ideal. Overall the PRNet provides excellent face swap results as it's altogether a superior technique to the traditional approaches.

Although, the PRNet performs great in ideal conditions(like in Test1 frames), it has difficulty when the orientation(in our face swap and Test3 frames) and lighting conditions do not favor the image identification with dlib so the full 3D mesh grids get messy.

Fig. 5. Delaunay Test2 Result

Fig. 6. TPS Test1 Result

Fig. 7. TPS Test3 Result

Fig. 8. PRNet Test1 Result

Fig. 9. PRNet Data1 Result

Fig. 10. PRNet Data2 Result

Fig. 11. PRNet Data2 Result

Fig. 12. PRNet Test3 Result

Fig. 13. Failure Case: Unsuccessful swap due to orientations of both faces using PRNet.

Fig. 14. Failure Case: Our tilted and rotated faces caused difficulty for PRNet in some areas

III. Conclusion

It is abundantly evident to us that the PRNet method performs much better face swapping among the methods implemented, assembling swapped frames for different orientations and lighting with more accuracy. Resized input images for swapping with traditional as well as the PRN method produce much quicker results, whereas higher resolution inputs require more time. The Delaunay Triangulation produces good and consistent results, whereas the Thin Plate Splines is not so consistent. Though it's less consistent than Delaunay Triangulation, sometimes TPS does produce better results.

References

[1] Yao Feng, Fan Wu, Xiaohu Shao, Yanfeng Wang, Xi Zhou. "Joint 3D Face Reconstruction and Dense Alignment with Position Map Regression Network" arXiv:1606.03798

[2] Dlib

[3] Thin Plate Splines

[4] GitHub References 1 2