Face Swapping in Video

CIS 581 Computer Vision, Fall 2021

Group Member

- Xinyuan Zhao: zxy9815@seas.upenn.edu
- Jierui Zhang: jieruiz@seas.upenn.edu
- Meng-Chuan Chang: mcchang@seas.upenn.edu
- Zheng Feng: zfeng627@seas.upenn.edu

Summary

In this project, we will perform automatic face swapping for a selected pair of faces in 2 videos. Videos will be divided into frames and for each frame, faces will be automatically detected, extracted, swapped and color blended. Finally, this program will output 2 videos with swapped faces. For the swapped faces, their edges will be blended to make sure smooth color and lighting transitions. Third party libraries such as Dlib, OpenCV, face-alignment and mediapipe will be used to detect faces and process videos.

Goal and Objective

- Make sure the source face should not emote the target face.
- Make the face swapping seem natural.

Related Works

- https://github.com/ageitgey/face_recognition
- https://github.com/serengil/deepface
- https://github.com/1adrianb/face-alignment
- https://google.github.io/mediapipe/

Approach

The pipeline for face swapping is to perform a single frame face swapping for each frame using the steps described below. And between the frames, we use optical flow to track the faces and make the frame transition smoother. Below is a high level description of single frame face swapping pipeline and a graphical representation of the pipeline is shown in Figure 1.

Face swapping on single frame

1. Extract a single frame image from source and target videos.
2. Detect faces and landmark points (dlib and pytorch). Extract the convex hull of the face as a mask.
3. Use the landmark points to perform Delaunay triangulation.
4. Color the triangulated target face with pixels of the source face.

5. Use the face mask to combine swapped faces with the rest of the images.
6. Gradient blending on swapped faces.

**note: the term “source face” used in this report refers to the face that is pasted onto the target image and overwrites the original face of the target image.*

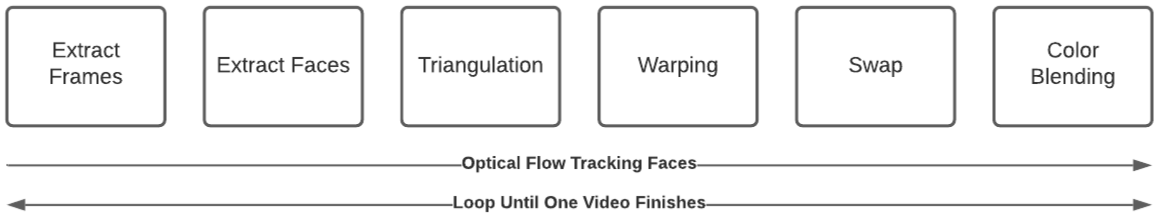

Figure 1: Video Face Swapping Pipeline

Facial Landmark Detection and Convex Hull Extraction

For facial landmark detection, we first used dlib library and extracted 68 landmarks for each frame.To generate convex hull, we used cv2.convexHull to connect the outer points and form a hull. This attempt failed on frames with side faces. To improve the result, we then used mediapipe and face_alignment library to generate more accurate and more number of landmarks. Mediapipe library can generate 468 landmarks per face. To further mediate the problem in side faces, we tried to eliminate the outermost circle of landmarks and reviewed different video outputs based on each point removal. Face-alignment library uses state-of-art face alignment network to align landmark points with faces, which can produce accurate landmark points from any views of the face. For this project, we use the 2D model with dlib as the face detector. Comparing these two facial landmark detectors, the mediapipe can produce high resolution, accurate and consistent landmarks for frames but it fails to align with side faces (Illustrated by Figure 3 in Appendix). Face alignment network is able to produce accurate landmarks for any viewpoint of the faces, but it lacks consistency (location of landmarks can vary each time). Therefore, we decided to mix Mediapipe and face_ alignment package for face detection in the video.

Triangulation, Warping and Swapping

With the facial landmarks and convex hull extracted, we now know the feature points of two faces. To achieve the goal of swapping faces without emoting the original face, we decide to maintain the shape and geometric properties of the original face and to recolor the face using pixels from the source face. This is similar to the Delaunay triangulation and cross-dissolving in the image morphing module. Using the facial landmarks as the control points, we perform Delaunay triangulation on both faces. Then, we fill in the triangulated target face with corresponding pixels of the triangulated source face. Applying this procedure to the frames of 2 videos, we obtain the 2 swapped faces. Finally, we use the face masks given by the convex hull extractions to combine the swapped face with the rest of the original image’s background .

Gradient Blending

To make the color of the swapped face more natural and consistent in the target image's background (e.g. lighting condition), we use poisson blending to reconstruct the color of the face in the target frame. We decided to use the existing color blending function, cv2.seamlessClone in OpenCV library. To implement this function, we need the target frame without the face (use mask generated by the convex hull extraction), the face in the source frame with mask, and the centroid to put the face (use cv2.boundingRect). The color blending works well and it solves the issue of color discontinuity between the face and the background. Although we solved the main problem, there are still some minor issues.

One issue is the illumination. We copy the gradient in the face region and apply those values to the target frame. In this scenario, the gradient would keep the original illuminated condition (e.g. shadow). Therefore, if the illumination is different between the source and target, we would see some unnatural color intensity in our result (for example, mismatch of the light condition to dark region on the face shown in Figure 2 in Appendix)

The other minor issue is the pick of boundary color. In the condition of the face turning with an angle, our convex hull cannot capture the correct landmark position, especially the outline of the face (shown in Figure 3). While we implement poisson blending to this target frame, the function would reconstruct the face with the incorrect boundary color, leading to the unnatural color. In Figure 2, the color intensity of the left eye is much higher (brighter). To solve this problem, we may choose better face landmark detection model

Optical Flow and Video Processing

If we produce the final results by directly putting swapped frames together, the result videos will look discontinuous. The motion between frames is abrupt and not smooth. Since we detect facial landmarks every frame, these landmarks might be at different locations each frame and as a result, the warped face will be very different each frame (the landmarks may not perfectly track the intended features in the video, thus the motion of the swapped face may not be continuous as in the original frames). This discontinuous landmark detection will cause the faces in result videos shaking and blurry. To solve this problem, we decided to use Lucas Kanade Optical Flow to track the facial landmark points of each frame. Optical flow is able to detect landmark points of the current frame given the corresponding points of the previous frame, so that the transition between frames will become more continuous and smoother. Using optical flow to detect landmark points will also make the processing faster. To ensure accuracy, we use dlib landmark detection every 3 frames and use optical flow tracked landmarks for the rest of the frames.

Results

To produce optimized results, we mix the 2 methods of face detection such that fine mesh (mediapipe) is used for frames with full facial features, and face-alignment network is used for frames with side faces or missing features.

Video 1: Source face = FrankUnderwood, target face = MrRobot
https://youtu.be/4XVPV-WeJU0

Video 2: Source face = MrRobot, target face = FrankUnderwood
https://youtu.be/Qr69snANs3k

Discussion

Overall, the swapped faces kept their original emotions. The color blending also looks smooth. However, there are clearly features that can be further improved.

First, the color blending can be improved as the swapped faces have unbalanced brightness (Too dark on the left and too bright on the right side). This unnatural light intensity on faces might be caused by the illumination of the source frame. To solve this, we need to adjust the gradient according to the illumination of the target frame so the brightness of faces is consistent with the target frame's background.

There are also some noticeable deformation of features of the swapped face. We believe this is due to the fact that in some frames the source image's head/face rotated and faced away from the camera while the target image's head/face faced directly at the camera. Even though the features of the side face can be roughly tracked and marked with the face-alignment library, coloring the triangulated target's face with pixels of the triangulated source (side) face will not look natural (in general, this problem occurs when the 3D orientations of source and target faces are significantly different in the depth component). The orientation of the swapped face will not match the orientation of the unswapped part of the head in the original video, creating unnatural results.

In addition, the motion transition between frames is not as smooth as the original frames. To further improve this, we can track the motion of each landmark point and smooth its high frequency bouncing motions by applying a low pass filter.

Future Works

Illumination

Adjust the gradient based on the illumination in the target frame.

Compensating for differences in face orientations

We were not able to figure out a solution to accommodate for the scenario of the source and target faces having different orientations in the same frame. We can try to find some algorithms to detect the orientations of the two faces. Ideally, with the orientation information and the given source image, we need to find a way to 1. Transform the 2D shape of source face's landmarks to the shape as if the photo is taken when the source face is facing in the same orientation as the target face; 2. The pixel values of the transformed source face should be adjusted accordingly, e.g. the intensity change of the source face's pixels if facing in the same orientation as the target face.

If the orientation difference is 90 degrees or larger, the feature loss cannot be re-obtained by transformations. So in that case we might need to interpolate the entire face from previous frames with most facial features available.

Responsibilities

- Xinyuan Zhao: Video processing, Optical Flow tracking frames
- Zheng Feng: Detect faces and construct convex hulls.
- Jierui Zhang: Face Swap, Color Compensation using Gradient Blending
- Meng-Chuan Chang: Affine transformation and make the mask for target image

Appendix

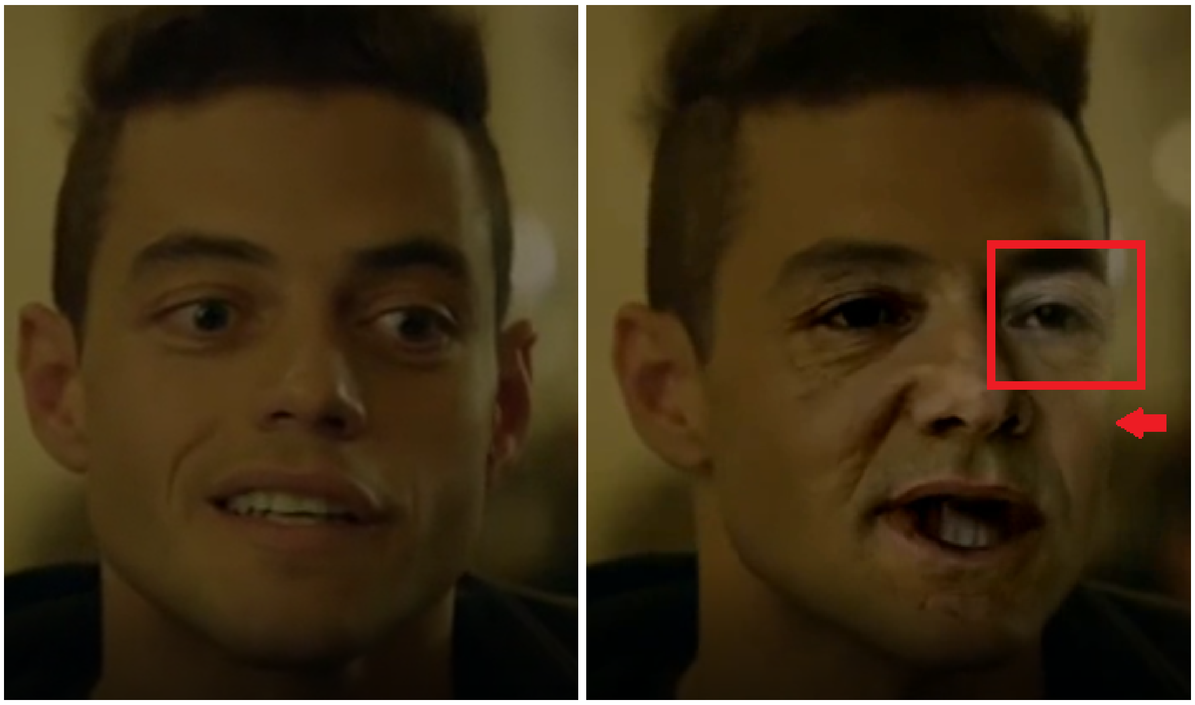
Figure 2. Problem with Gradient Blending

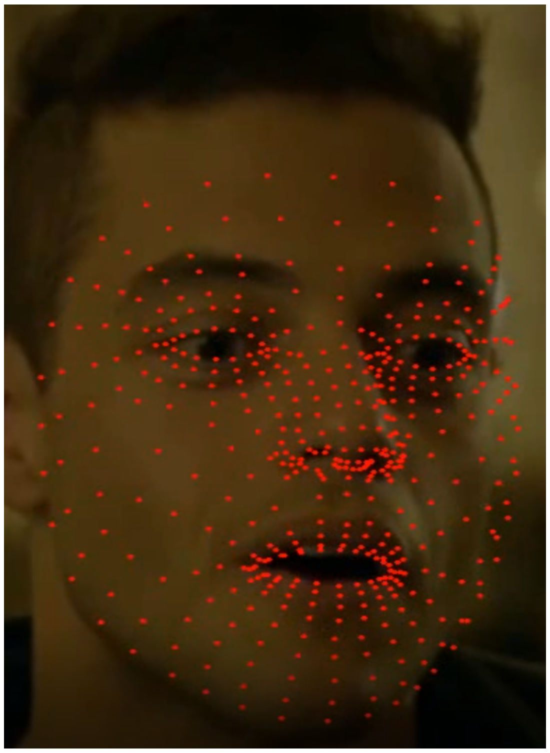

Figure 3. Mediapipe landmarks outer layers unable to accurately capture the edge of the side face