Face Swapping Final Report

Shiwei Ge
Jiawei He

November 2022

1 Goals and Objectives

- **Fulfilled Objectives:**
  - Get face landmarks from images
  - Crop face automatically from the images
  - Knit the face to the output image
  - Face swapping both in images and videos

2 Proposed approach

1. Get face landmarks
2. Triangulate the face based on face landmarks
3. Apply affine transformation to each corresponding triangle area
4. Combine all the triangle areas into a new face
5. Knit the new face to the output image
6. Apply face swapping to each frame of video
7. Complete seamless swapping in a video

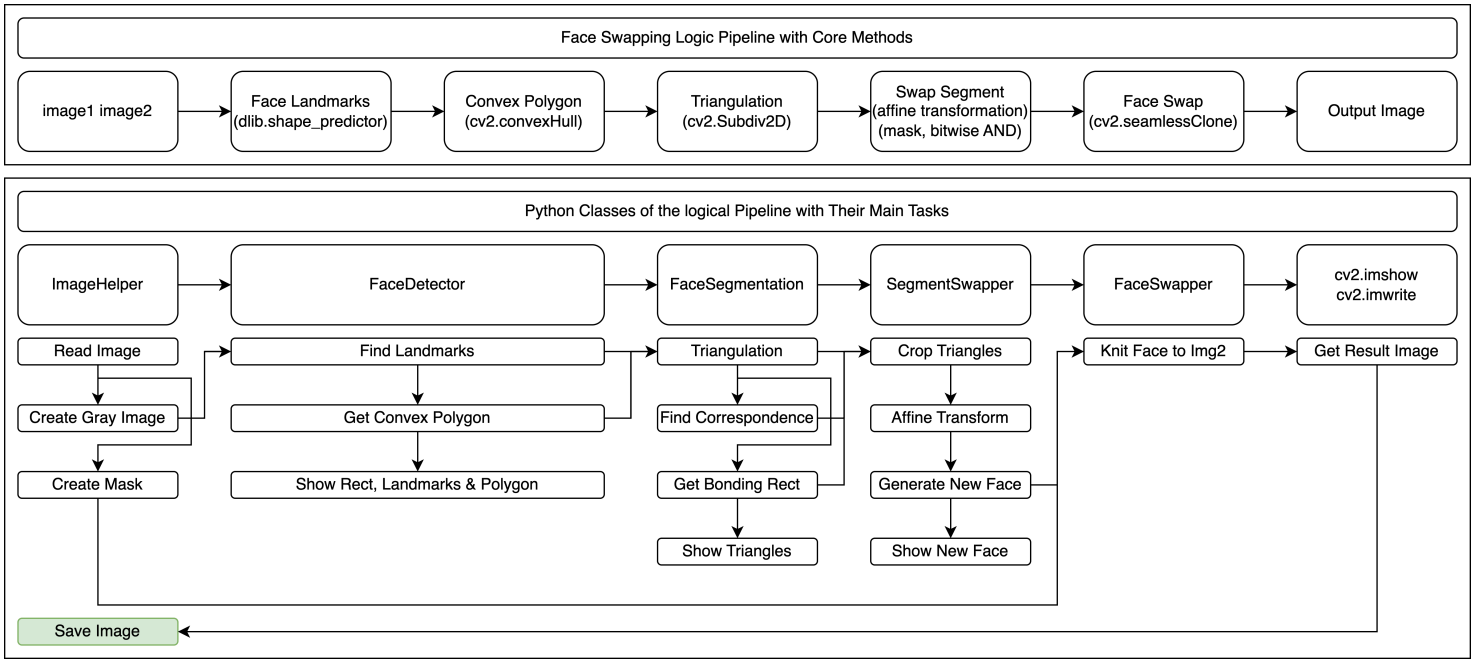

Figure 1: Our Pipeline

3 Landmarks

For face swapping in images, the first step is to retrieve source image and destination image. After this step, it's time to find out the landmarks in both images using face detector from dlib library.

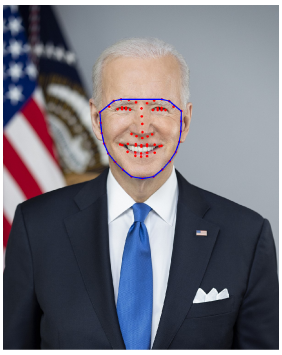

Figure 2: Joe landmarks

Figure 3: Trump Landmarks

4 Triangulation

After getting both landmarks from images, we're going then to segment the face into triangles. This step is the core of our face swapping, as later we will simply exchange each triangle with the correspondent triangle of the destination image.

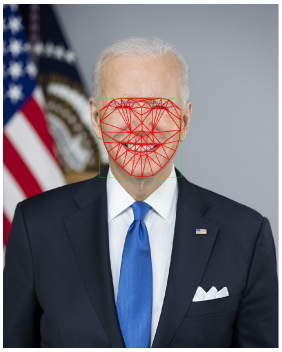

Figure 4: Joe Triangulation

Figure 5: Trump Triangulation

5 Linking Triangles

Once we have cutted and warped all the triangles we need to link them together. We simply rebuild the face using the triangulation pattern, with the only difference that this time we put the warped triangle. At the end of this operation the face is ready to be replaced.

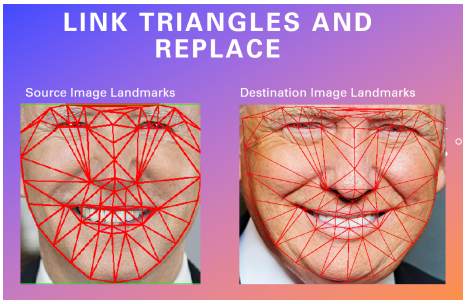

Figure 6: Linking Triangulation

6 Seamless Cloning

Finally the faces are correctly swapped and it's time to adjust the colors so that the source image fits the destination image.

Figure 7: Original Trump

Figure 8: Trump with joe's face

7 Mid-point Progress

Our final goal of the project is to do the face swapping between to videos. For now, we have done the face swapping between to static images. To swapping the face between two images, we have completed a image processing pipeline through python with open-cv and dlib. In the pipeline, we utilized dlib to detect the the face and output the landmarks points from both the input image and the output image. Then we plug the landmarks to the open-cv library to knit the face area from the input image to the remaining area of the output image.

Figure 9: Swap President Trump's face to President Biden's face

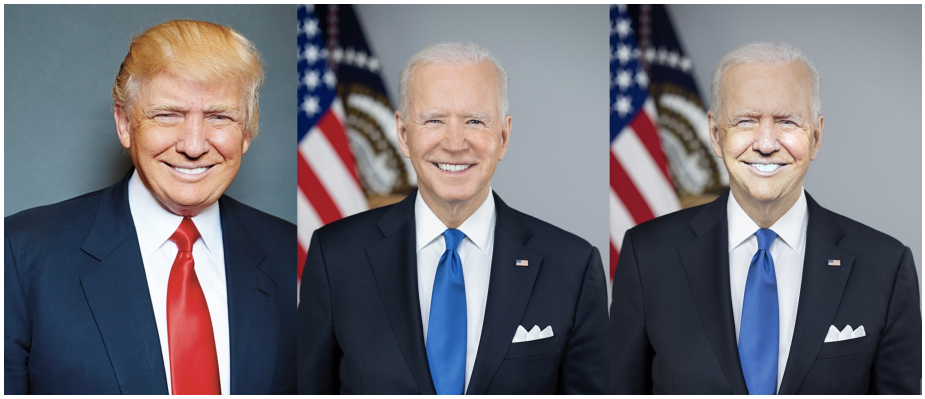

Figure 10: Swap President Biden's face to President Trump's face

8 Final Progress

As for the mid-point review, we completed the face swapping between images. After that, we are improving the efficiency of our code and also completed the face swapping in a video by given a source image. link to result video

What's more exciting is that the application is easy to run, which requires only one simple command line.

Package Requirements

Make sure you installed the following dependencys

```
pip install -r requirements.txt
```

Instructions on how to run the project

The project divided into several parts

> Face Detector Class
>
> Face Segmentation Class
>
> Face Swapping Class
>
> Main Class (integrated with absl)

Command line based application

Since it's a face-swapping application, which consists of swapping face in image and swapping face in a video, the user is required to have either two images which are `source image` and `destination image` or `source image` and `destination video (The video you want to put the source image face on)`

- Face Swapping between images

```
python3 Main.py --i <source-image-path> --o <destination-image-path>
```

- Swapping image into a video

```
python3 Main.py --i <source-image-path> --video <path-to-video>
```

> After running the command above, there's will be `output.avi` file in `images` folder, which is the video after face swapping

Figure 11: Instruction

9 Timeline

In the next phase we are going to apply the image face swapping techniques into video face swapping. We already started working on that but there are still some bugs for us to fixed and the performance of video swapping may be optimized by rearranging the code. We are still working hard on those tasks.

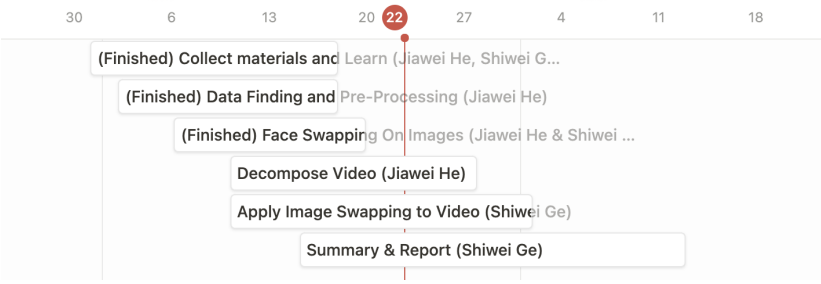

Figure 12: Gantt Chart