Digital Image Processing
Final Project Report - Group 4
Face Replacement

Jay Chien b07902021
Po-Heng Chen b07902114
Yi-Min Lin b07902016

1 Motivation

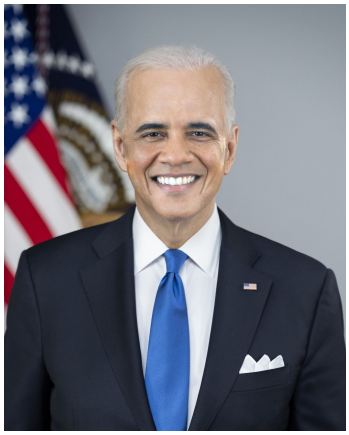
Face replacement is a technique that is widely applied to social medias, APPs, and so on. In this project, we tried several method to bring remarkable effect without using learning-based techniques. In the following, we demonstrated some of them, including multi-resolution blending and Poisson editing, to replace the face area from two pictures.

2 Problem definition

Pictures below shows the result we want to achieve:

(a) source image

(b) target image



(c) result image

We want to take two input images (source and target) and then automatically generate one output image (result image) that replaces the face of the target image with the one of the source image.
To be more specific, the problem includes the following steps:

- To detect the faces and generate masks of the input images automatically.
- To align the faces of the images and resize the image to same size so that we only need one mask (target mask).
- To eliminate the discord between the face of the source and the background of the target in the output image.

3 Algorithm

Face Recognition and Mask

- We used open source face recognition tool (Ageitgey) to detect the facial features
- Use facial features to generate the contour of mask for each face
- We also build specific mask for each facial features

(d) Masks example

Face Alignment

- Compute transition and scaling between source image and target image:
  - Use the centroids of the two images to compute relative transition
  - Use width and height of the white area of masks to compute horizontal and vertical scaling
- Use backward transition to generate the aligned source image

Image Blending

- Direct blending
  - Abstract:
    Directly replace the face of the target image with the face of the source image without any adjustment.
  - Algorithm:
    The result image is combined with two parts: source face (mask[i][j] == 1) and target background (mask[i][j] == 0).
    So we derive the formula:

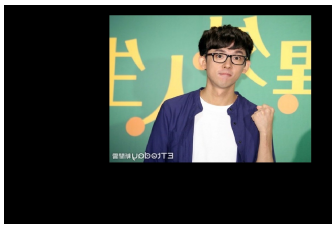
$$\text{result[i][j]} = \text{mask[i][j]} \times \text{source[i][j]} + (1 - \text{mask[i][j]}) \times \text{target[i][j]}$$

- Multi-resolution blending
  - Abstract:
    The method was firstly proposed by Peter Burt And and Adelson (1983). The purpose is to perform smoother blending for "low-frequency" parts (like cheeks, forehead), and to perform sharper blending for "high-frequency" parts (like eyes, mouth).

– Algorithm:
To achieve the goal, we first construct a Gaussian pyramid and a Laplacian pyramid respectively:

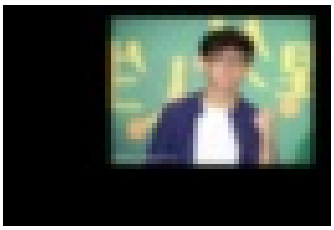
$$
\begin{aligned}
G_0 &:= \text{the original image} \\
G_i &:= (G_{i-1} \circledast K)_{\text{down sampling}\downarrow} \\
L_i &:= G_i - G_i \circledast K \\
i &= 0 \dots n
\end{aligned}
$$

where G_i denotes the i-th image of Gaussian pyramid, and L_i is the i-th image of Laplacian pyramid. K is the 5×5 convolution kernel.



(e) G_0 of source image

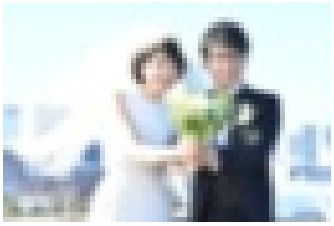
(f) G_2 of source image



(g) G_4 of source image

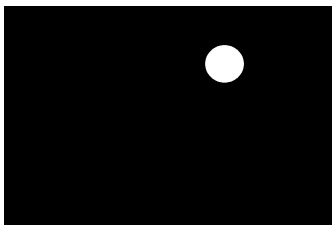
(h) G_0 of target image

(i) G_2 of target image

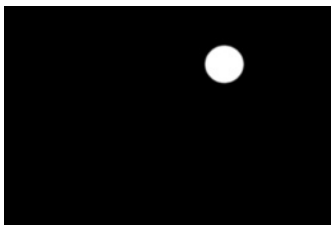


(j) G_4 of target image

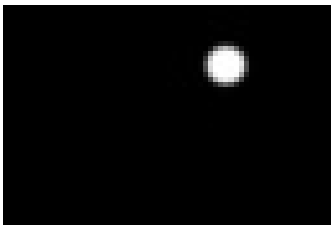
By constructing such pyramids, we have images from high-resolution (original resolution) to low-resolution. And with the same approach, we can build masks for each resolution, and the lower resolution the mask is, the smoother blending it can do.



(k) G_0 of the mask



(l) G_2 of the mask



(m) G_4 of the mask

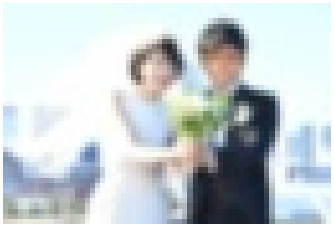
Define the intermediate result image as i-th level as R_i, we can reconstruct the result image as follow:

$$
\begin{aligned}
R_i &:= (R_{i+1})_{\text{up sampling}\uparrow} + L_{i,\text{src}} \times G_{i,\text{mask}} + L_{i,\text{tgt}} \times (1 - G_{i,\text{mask}}) \\
R_n &:= G_{n,\text{src}} \times G_{n,\text{mask}} + G_{n,\text{tgt}} \times (1 - G_{n,\text{mask}})
\end{aligned}
$$

And R_0 is the final result image.

(n) R_0

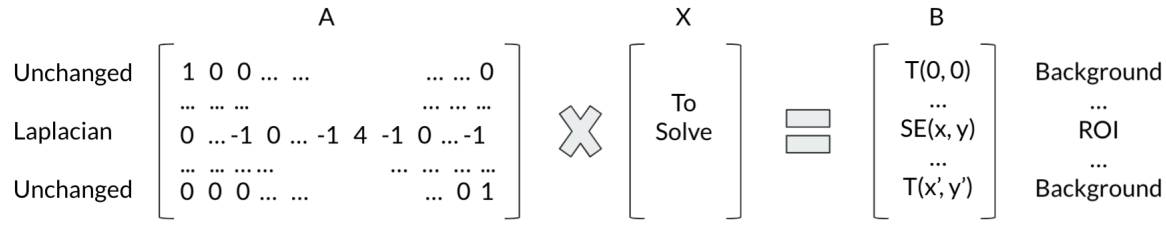
(o) R_2



(p) R_4

- Poisson editing
  - Abstract:
    Pérez et al. (2003) proposed this method to smooth the edge across the ROI(region of interest) and the background, and to make the color and brightness similar inside and outside ROI.
    The idea of Poisson editing is that the visual effect of an image is determined by high-frequency parts. That is, human eyes are more sensitive to the shape and sharp changes of an image. So in this method, we extract the high-frequency features of ROI by Laplacian filter, and then reconstruct the region using the background of target image.

  - Algorithm:
    N is the number of the pixel of the result image.
    1. A is a square matrix in the size of (N^2, N^2)
    2. X is the vector of unknown pixels to be solved in size of N^2
    3. B is the vector of pixels of the image combining source face and target background in size of N^2

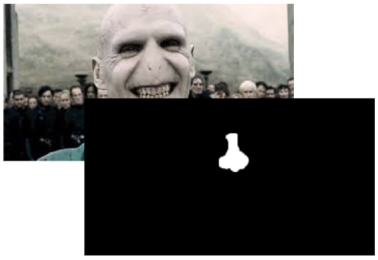


To construct matrix A, we view each row as a pixel's equation to be solved. If the i-th row represents a pixel in the background, the i-th answer to the equation in X should be the same as the i-th pixel in B, so we fill the row with all zeros but the i-th element with 1 in A. If the j-th row represents a pixel of the face, we want the pixel be reconstructed using it's four neighbors in the combined image, so we fill the row's j-th element with 4 and fill the neighbors with -1 in A.
Then we can solve X by $A^{-1} \times B$

– example:
The following demonstrates how to construct Tom Riddle's nose using Tom Cruise's one:

1. Extract Laplacian map of Cruise's nose and paste it to Riddle's nose area.

(q) Cruise's nose and mask



(r) Riddle's nose and mask

(s) The prototype to be reconstructed

2. Use Poisson equation to get result image.

4 Experiments results

- Resulting quality comparison for each method:

- Resulting quality comparison for each method:

- Multi-resolution method with different levels:

2 levels 4 levels 6 levels 8 levels

- Resulting naive face swapping:

source image target image result image

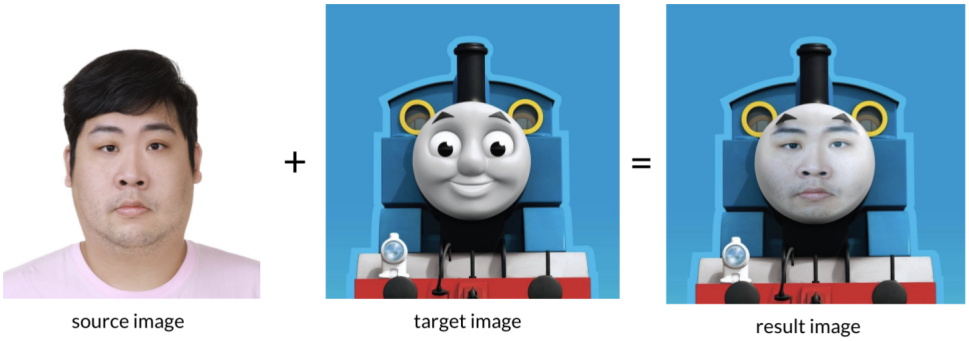
- Resulting specific facial feature swapping:

source image target image change eyes change nose

- Resulting effecting of face alignment:

source image target image result image

- Resulting effecting of image blending:



5 Discussions

Although our algorithm do well on most image pairs, there is still some limitation on source images

- Algorithm can't get good result when faces in image pairs are toward to different orientation.
- The noise or other object near the boundary of mask may affect the process of blending and influence the quality of output.

For the first problem, we may study on how to project image to 3D spaces and then use some machine learning technique to simulate and predict how would the face be like when it change it's orientation.

For the second problem, we may try to do object detection when we making our mask for swapping. For example, we can detect the position of hair in mask region and remove it before blending step. Keeping only face in mask can make sure blending would not be affected.

Finally, we may try to extend our project from image to video. Many similar technique, like building mask for each frame in video, and do image blending can continue to be used on video face swapping. Hoping that we can really make it in future work.

References

Ageitgey. ageitgey/face_recognition. URL https://github.com/ageitgey/face_recognition.

P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM SIGGRAPH 2003 Papers*, 2003.

P. J. B. Peter Burt And and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics,Volume 2,Issue 4*, 1983.