Human Facial Expression Modification

Difan Chen
Texas A&M University
401 Joe Routt Boulevard, College Station, TX 77843
dfchen6@tamu.edu

Zhou Wang
Texas A&M University
401 Joe Routt Boulevard, College Station, TX 77843
wangzhou123@tamu.edu

Abstract

Facial expressions represent a variety of emotions. Expressions like happy, angry, bored, surprised are slightly different from each other since the majority part of the face remains the same. In this project, we made it possible to modify facial expressions automatically. We provided two methods, the first one is image morphing with a reference image. The second method is image warping based on local changes of selected feature points.

1. Introduction

In this project, we are going to propose a method to modify human facial expression to express different emotions.

1.1 Significance

People's emotions can be captured through facial expressions. Under some certain circumstance, facial expressions need to be modified. For example, in game or movie Industry, they want to exaggerate one's emotion by moving or warping one's facial features. Thus, our facial expression modification research can provide a basis to more sophisticated applications.

1.2 Scope

The ultimate goal of this project is to realize automatic expression modification. To achieve facial expression modification, there may exists two ways.

The first method is choose a reference image and warp both the input image and reference image to create the morphed image. The reference images have different emotions like laugh, cry, boring, angry, etc. The basic procedure is first define corresponding points and triangulation for both images. Then compute the average shape and get the affine projection to corresponding triangles in each image. Fill the pixels within triangle and add two warped images to get the morphed image.

The second method is warp the input image without reference image. The basic idea here is to adjust facial features of the input image. For example, the different shape of mouth may represent different emotions like happy or sad. So modifying some parts of the image and leave the remaining part the same may also be able to modify facial expressions.

2. Theory

The implement of 2-dimensional image warping and morphing is the key technique in this project. We applied the image warping technique to modify images of human face expression, and the image morphing technique to blend images of human face.

2.1. Digital image

A digital image is usually a numeric representation of a two-dimensional image. The digital images contain a fixed number of rows and columns of pixels. And pixels are the smallest individual element in an image, holding quantized values that represent the brightness of a given color at any specific point.

A digital image can be viewed as a function of pixel coordinates:

$$I(x): D \subseteq \mathbb{R}^n \to c \subseteq \mathbb{R}^m \quad (1)$$

$$x \mapsto I(x) \quad (2)$$

m is the number of channels of the image (for example, 1 if the image is gray-level, 3 if the image is RGB or an arbitrary number for multi-spectral image…)

n is the number of spatial dimensions (for example, 2 if the images are traditional 2D picture, 3 for computer aided tomography image…)

2.2. Image warping

Image warping is a transformation that is applied to the domain of an image, which modified the geometrical properties of the image itself. Ideally, the intensity if the warped image is the same as the intensity of the original image at corresponding points.

This transformation can be represented as warping

function, or mapping function. A 2D mapping T_θ is a function:

$$T_\theta(x): \mathbb{R}^2 \to \mathbb{R}^2 \quad (3)$$

$$(x;\theta) \mapsto T_\theta(x) \quad (4)$$

θ is the vector of parameters of the transformation.

x is the point to be mapped.

There are two common approaches to interpolate the transformation of pixel locations. One is linear interpolate, the other is spline interpolate. We divided human face images into small triangles, and applied linear interpolate method to obtain new locations of pixels in each small triangle. Considering the linear transformation of rotation, scaling and translation, the transformation function of pixels in each triangle can be represented as:

$$T_\theta(x) = s\begin{bmatrix}\cos\phi & -\sin\phi \\ \sin\phi & \cos\phi\end{bmatrix}x + \begin{bmatrix}t_x \\ t_y\end{bmatrix} \quad (5)$$

For a given image to be warped, we need to compute the mapping function T_θ, and we could know the coordinates of all current pixels in warped image. And this is called forward warping. While in practice, we found that the rows and columns of digital images are integers, but the transformed indexes are usually not integers, which means many pixels may not find a suitable location in warped image while many locations in warped image will stay empty. Once we obtain the mapping function T_θ, the inverse function T_θ^{-1} can be obtained easily. In this case, we applied inverse warping. The inverse warping function can be represented as:

$$T_\theta^{-1}(y) = \frac{1}{s}\begin{bmatrix}\cos\phi & \sin\phi \\ -\sin\phi & \cos\phi\end{bmatrix}x + (y - \begin{bmatrix}t_x \\ t_y\end{bmatrix}) \quad (6)$$

We obtain the overall warping function by adjusting the shape of small triangles. The transformation function of each triangle can be computed when given the locations of vertex in both original and warped images.

Most primary feature points on human face, such as pupils, lips and brows… were defined as those triangle vertex, and formed the triangles on the image through triangulation.

Finally, combining all the warped triangles, the warped whole image is obtained.

2.3. Image Morphing

Image morphing is a special effect in motion pictures and animations that changes one image or shape into another through a seamless transition. Most often it is used to depict one person turning into another through technological means or as part of a fantasy or surreal sequence.

Our morphing technique is based on the warping. We implemented this morphing of two human face images by creating a sequence of intermediate images which are the sums of warping images with weighted average. Also based on the extraction of feature points on human faces, with different weighted average, we find an intermediate feature point for each pair of feature points, and warp them to an intermediate location, and cross-dissolve the pixels from two sources. The intermediate location and the ratios when they cross-dissolve depend on the weighted average.

Assume the cross-dissolve ratio is t, and $0 \le t \le 1$.

The new location of intermediate pixel is

$$l_{new} = t \cdot l_0 + (1-t) \cdot l_1 \quad (7)$$

The pixel value of intermediate pixel is

$$p_{new} = (1-t) \cdot p_0 + t \cdot p_1 \quad (8)$$

Similar to the idea in image warping, all these operation is done after image is divided to small triangles, and combining all the morphed triangles, the morphed image can be obtained.

3. Implementation

This part talks about how to translate the theoretical problems to MATLAB codes. For the first method, it involves inverse warping and image morphing. As for the second method, feature points automatic detection and image warping will be introduced. At last, the MATLAB GUI will be explained.

3.1. Image morphing with reference

3.1.1 Image warping

Inputs for this function are source image (HxWx3), coordinates of keypoints in the source image (2xN), coordinates of where keypoints end up after the warp (2xN) and list of triangles (Kx3). Then for each triangle, we can compute transformation which maps it to from the target back to the source. So each pixel within that triangle should be able to use the same transformation matrix to find its corresponding position in the source image. Another thing we need to do is to figure out what triangle it lives in so we know what transformation to apply for each pixel in the target image. Then find source coordinates for all target pixels lying in each triangle. Now we know where each point in the target image came from in the source, we can interpolate to figure out the exact values. In this project, warp1.m is the function of image warping.

3.1.2 Image morphing

Image morphing stands for image averaging. According to inputs of image warping, we need to figure out the coordinates of feature points for both the source and target

images. In our case, the source images are two input image, the target images is combination of feature points coordinates from two input images, specifically, pts_target = (1-t)*input_points + t*base_points.

The following codes can self-explain the procedures of image morphing.

```
  % now produce the frames of the morph sequenceA
  for fnum = 1:61
    t = (fnum-1)/61;
    pts_target = (1-t)*input_points + t*base_points;            % intermediate key-point locations
    I1_warp = warp1(I1,input_points,pts_target,tri1);              % warp image 1
    I2_warp = warp1(I2,base_points,pts_target,tri1);             % warp image 2
    Iresult = uint8((1-t)*I1_warp + t*I2_warp);                  % blend the two warped images

imwrite(Iresult,sprintf('frame_%2.2d.jpg',fnum),'jpg')
  end
```

An result of image morphing example is shown in Figure 1.

Figure 1. Result of Image Morphing

3.2. Image warping based on local changes of selected feature points

The purpose of the second method is to modify one's expression automatically. From the view of implementation, it can be divided into three steps: Subtract feature points, create new coordinates of feature points for target image and image warping for final result.

3.2.1 Subtract feature points

To gain a better accuracy and time efficiency, we used a face detection API called ReKognition. The idea here is to send the image data in base 64 to the server, and the server will send back the coordinates of feature points in the original image.

The following codes is about how to achieve this goal in Python.

```
import requests
import base64
import sys
def findFeature(imagePath):
    # Load the image
    with open(str(imagePath), "rb") as image_file:
            SAMPLE_FACE = base64.b64encode(image_file.read())
    # Prepare the data to be sent
    datasend = {
                    "api_key": "cAyRC4Mm0G2fQqrK",
                    "api_secret": "Qnl03t3bxMOGn9Mp",
                    "jobs": "face_part_detail",
                    "base64": SAMPLE_FACE
            }
    response = requests.post("https://rekognition.com/func/api/", data=datasend)
    print response.content
    mydic = response.json()
    mylist = mydic['face_detection']
myfacedic = mylist[0]
```

Figure 2. Positions of feature points sent back from API

Then we can save the coordinates of features in a local text file. And this local text file can be used for image warping in the following steps.

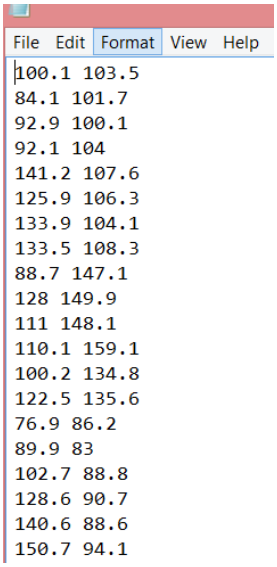

Figure 3. Coordinates of feature points saved in local file

3.2.2 Create new coordinates of feature points for target image

To achieve different facial expressions, we need to modify the coordinates of feature points correspondingly. Unlike in method one of creating intermediate points, here, we only need to create new features around its original position. For example in Figure 4, to make someone smile, we can only move the left feature point of mouth left further and right feature point right further.

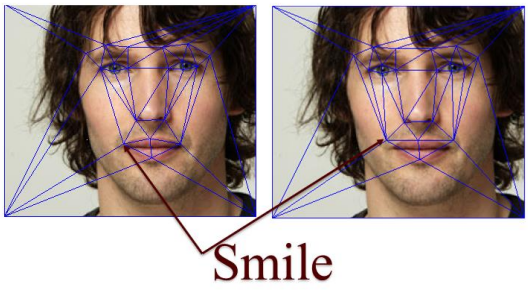

Figure 4. Coordinates of selected feature points change slightly

3.2.3 Image warping for final result

After the previous step, we have all inputs needed for image warping now. For smiling as shown in Figure 5.

Figure 5. Happy effect

For other facial expressions like unhappy and surprise, examples are shown in Figure 6 and Figure 7.

Figure 6. Unhappy effect

Figure 7. Surprise effect

3.3. MATLAB GUI

An MATLAB GUI helps user modify facial expressions more interactively and easily. Besides, we support two ways of loading an image. The user can either use web camera to capture an image or upload a local image. There are three buttons of “Happy”, “Unhappy” and “Surprise” which yields different result of facial expression modification.

The interface is shown as Figure 8.

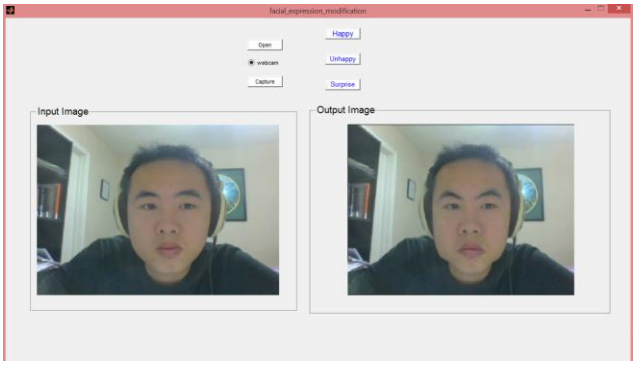

Figure 8. MATLAB GUI

4. Conclusion

Through the implementation, we reached the targets of modifying facial expressions automatically. We are able to change the expression to happy, unhappy and surprised. By theoretical analysis, we modified the selected features points locally for different facial expressions. And after experiments, we found that the results are correct which showed our theoretical analysis is correct. As for the first method, we found that when we feature points for both images are close, the effect of image morphing is optimal. But basically, the aim of modifying one's facial expressions is achieved using the first method.

References

[1] https://rekognition.com/developer/start
[2] Ekman, Paul. "Facial expressions." *Handbook of cognition and emotion* 16 (1999): 301-320.A. Alpher. Frobnication. Journal of Foo, 12(1):234–778, 2002.
[3] http://www.ics.uci.edu/~fowlkes/class/cs116/hwk2/
[4] http://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15463-f11/www/proj3/www/zaw/
[5] http://vimeo.com/69978049