Smile recognition

Team members:

Xiaofeng Luo

Liyongshi Chen

Zehui Wang

Abstract:

This project aims to build up a smile recognition and simple gesture recognition system that helps users automatically take selfies when the camera detects their smiles and gestures, which will reduce users' operations and time. The system will catch the general shape of the hand and smile. The challenge for this project will be facial recognition and gesture recognition, one is using Opencv Classifier Haar-cascades to achieve and another is using contour lines of hand shadows.

Introduction:

This project is trying to recognize whether the user is smiling or doing popular camera gestures during the taking pictures. The system will ask the user the request to turn on the camera and to choose the smile or gesture detection, then capture the characters what the system needs to access. Since the characters compared have matched, the system will produce the photo to the default file as the result.

The detection system has separated into two parts, smile detection, and gesture detection. In the part of smile detection, the system is using the Viola-Jones algorithm and haar cascades which are a series of filters used to detect faces by their features one by one. Another part is gesture detection which is using skin detection, convex hull and law of cosine to calculate the number of acute angles so that to determine how many fingers the user raises up and based on its degree to define its basic gesture.

Background:

General background:

This idea basically came from an inspiration of FaceID on IOS devices. As we know Apple has already implemented facial recognition on iPhones since 2017 and this technology seems to be more and more widely used in the future. Furthermore, selfie software has become a new trend that people would love to use. Our team is trying to connect the facial recognition technology to the selfie software and produce a new way to take pictures automatically - Smile recognition photo. Then people don’t need to worry that their facial expressions in the photos don’t look nice and the extra process to press the selfie button.

OpenCV:

The system is implemented by the openCV with its package. OpenCV is one of the most popular open-source BSD-licensed libraries. It includes several hundreds of algorithms about computer vision.

CascadeClassifier:

The cascade classifiers are a series of object detectors. The main feature classifier that we used is Haar Feature which was proposed by Paul Viola and improved by Rainer Lienhart. It has trained with hundreds of sample views of a particular object, such as faces and compared with the same size of an arbitrary image as the opposite sample, and doing calculating in the established steps.

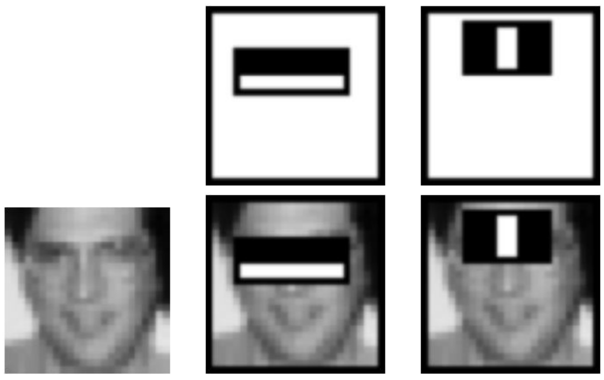
Relevant paper:

Paul Viola and Michal Jones. The paper named “Rapid Object Detection using a Boosted Cascade of Simple Features” has provided using the Adaboost machine learning based on cascade function which is trained by an amount of positive and negative samples.

Proposed Approach

Smile:

Smile detection is mainly using the Viola-Jones algorithm based on haar-like features to detect facial properties. (Fg.1) The detection is produced by two cascades which are the face and smiles. The system loads two XML files named “haarcascade_frontalface_default.xml” and “haarcascade_smile.xml”. Since the Viola-Jones algorithm works with grayscale images, we need to analyze the original image with grayscale in b&w. The system took 4 tuples: x,y, w, and h which correspond to the coordinates of the upper and left corner, width and height of the rectangle respectively. Then store them to a variable face and use OpenCV detectMultiScale function to get coordinates.



Fg.1 Haar features selected by AdaBoost

Gesture:

As for gesture detection, it will be more complicated than the smile detection than we imaged. First of all, it is to read the frame one by one from the camera like the operation steps in smile detection that using OpenCV VideoCapture function. Normally, in order to enhance the effect after processing, a Gaussian filter is required to reduce noise as a preprocessing step. However, we move this step and combine it with skin detection. The principle of gesture detection is simply based on the contours of human fingers, calculating its convex hull around the hand and angles to determine how many fingers the user is using and its basic gesture.

Due to the principle, it is easy to understand that the background in this detection is unnecessary where it should be thrown out and the human finger images should be extracted. Hence, skin detection is required. There were two skin detections that we tried in the code. One is based on 2 simple color threshold ranges in HSV color space, and another one is based on YCrCb color space. In method 1, we were giving upper and lower threshold values to determine the skin color range as [0,20,70] and [20,255,255]. However, as the article mentions that HSV color space is more intuitive than RGB color space where containing saturation and unsaturation (shades of gray) varies (Kolkur, 2017). In a simple way to say that, HSV space is sensitive to brightness and it does affect the accuracy of skin detection. As for method 2, YCrCb is an encoded non-linear RGB signal and Fg.1 will be the formula that transforms color from RGB to YCrCb. When it is used, its luminance is computed from the nonlinear RGB, which leads to a piece of separated luminance information in this color space (Kolkur, 2017). Hence, when processing the skin detection in two ways, YCrCb color space did better than HSV color space in reducing the light effects. In the end, we decided to use the YCrCb one in our project and Fg.2 and Fg.3 on below are the comparison output images of two different skin detections, as you can see the one which using YCrCb is more clear with a smooth contour line:

$$\begin{aligned} Y &= 0.299R + 0.287G + 0.11B \\ Cr &= R - Y \\ Cb &= B - Y \end{aligned}$$

Figure 2.1. The transformation formula from RGB to YCbCr

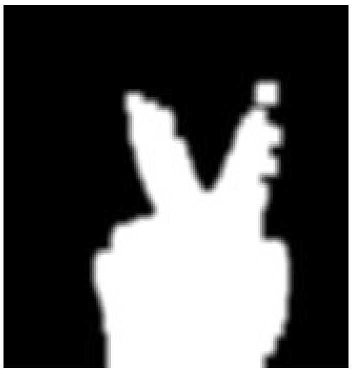


Figure 2.2. Skin detection based on HSV

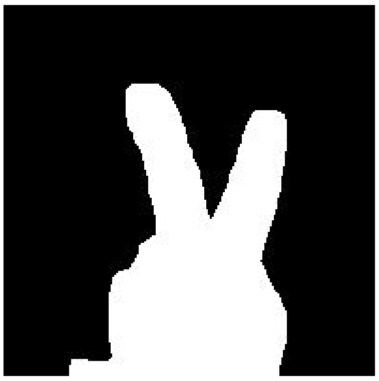
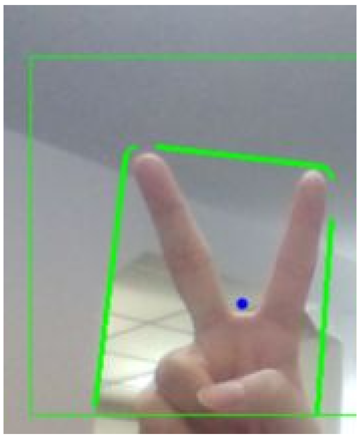


Figure 2.3. Skin detection based on YCrCb

From skin detection, we exclude the effect of background and some noises and get the basic shape of fingers on the frame. Based on the shape of a hand, it is easy to calculate its contours and the convex hull (can be imaged as the shape enclosed by a rubber band) around hand by using OpenCV functions, findContours() and convexHull(). Then, the most important essential function called convexityDefects() will help to find the convexity defect of the convex hull. The function will return three points: the beginning, the end and the far points and according to these three points and the law of cosines, $c^2 = a^2 + b^2$. to calculate how angles that the graph contains. Under normal situations, the angle between two fingers must be acute (exception for some special cases like Shaka sign). Hence, as long as the degree of an angle smaller than 90 degrees, the number of acute angles increases, which means the user raised fingers. In Fg.4, it shows an example when detecting the gesture. As you can see there is one acute angle in the graph which is illustrated out with a blue point and it means the user raises up two fingers.



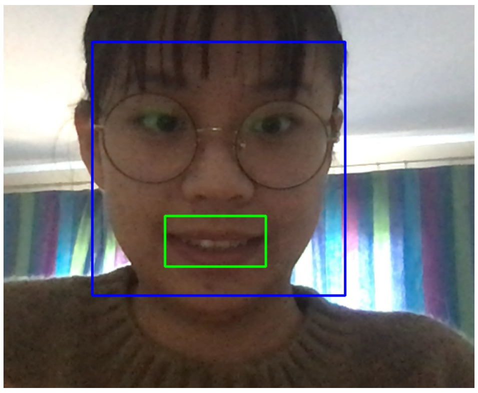
Fg.4. The gesture detection example

Results:

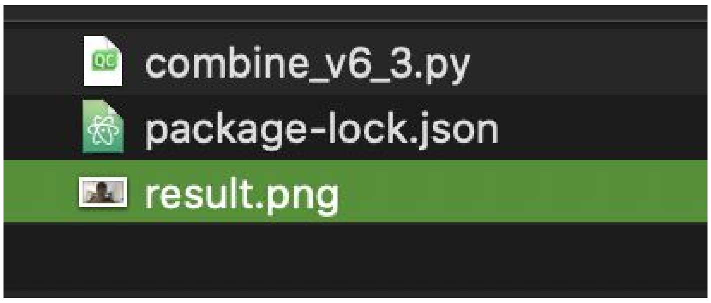
The project can recognize the smiley face and gestures to help the user taking photos automatically without redundant operation. In the project, the system will ask the request to turn on the user’s camera and let the user make a choice in which detection the user is preferring.

Smile:

The system will check the face of the user to analyze whether the user is smiling. If the data that the user's face is matching with features that we set, the system will identify the smiley face on the screen with a checkbox (Fg.1). Since the user is satisfied with the image that shows on the screen, the system will store it as the result into the folder.(Fg.2)



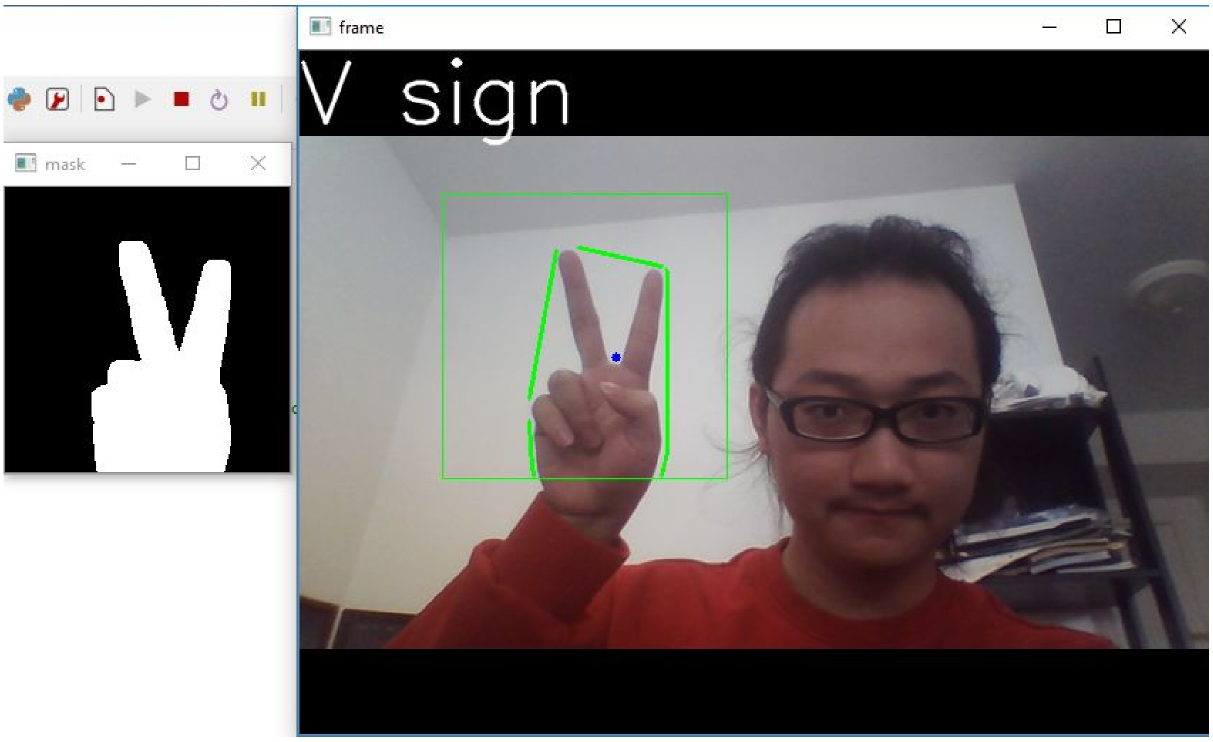
Fg.1 The system is showing the smiley part with the green box


combine_v6_3.py
package-lock.json
result.png

Fg.2 The resulting photo has been stored into the file folder

Gesture:

In this project, the gesture detection system will recognize and display a title of what gesture the user is now raising according to the user's gestures put in a specific recognition box (Example as Fg.3). On the other hand, a window named mask will display a result of skin detection of the checkbox that lets the user more clearly see what gesture now he is raising. When a user is raising a V sign gesture, the system will automatically save an image to a file. However, limitations of the system are also obvious. The skin detection will still be affected by illumination and when the user opens the system in a very dark environment, it will get stuck because of not recognizing any skin in the box.



Fg.3 The result of recognizing V sign

List of Work:

Equal work was performed by both project members.

GitHub Page:

https://github.com/littletonyhui/comp4102project

Demo Presentation Video address:

https://www.youtube.com/watch?v=U45CqwcWZTI&feature=youtu.be

References:

Srikanth Sriram, Babu Illuri. (Oct.2014) Real-Time Smile Detection using Haar Classifiers on SoC, International Journal of Computer Application(0975-8887), Volume 104, pp. 30-34.

P. Viola and M. Jones. (2001) “Rapid object detection using a boosted cascade of simple features,” in Proc. IEEE Conf. Comput. Vis. Pattern Recog., pp. 511–518.

Kolkur, S., Kalbande, D., Shimpi, P., Bapat, C., & Jatakia, J. (2016, December 1). Human Skin Detection Using RGB, HSV, and YCbCr Color Models. Retrieved from https://www.atlantis-press.com/proceedings/iccasp-16/25871632

Introduction of OpenCV doc. https://docs.opencv.org/master/d1/dfb/intro.html