# **2. 做一个颜值测试器**

## **2.1. 案例描述**

人脸识别系统以[人脸识别技术](https://baike.so.com/doc/6882364-7099854.html" \t "https://baike.so.com/doc/_blank)为核心，是一项新兴的[生物识别技术](https://baike.so.com/doc/5914583-6127494.html" \t "https://baike.so.com/doc/_blank)，是当今国际科技领域攻关的高精尖技术。它广泛采用区域特征分析算法，融合了[计算机图像处理](https://baike.so.com/doc/1402275-1482440.html" \t "https://baike.so.com/doc/_blank)技术与[生物统计学](https://baike.so.com/doc/5909832-6122738.html" \t "https://baike.so.com/doc/_blank)原理于一体，利用计算机图像处理技术从视频中提取人像特征点，利用生物统计学的原理进行分析建立数学模型，具有广阔的发展前景。

## **2.2. 原理分析**

人脸识别技术包含三个部分:

****(1)人脸检测****

面貌检测是指在动态的场景与复杂的背景中判断是否存在面像，并分离出这种面像。

****(2)人脸跟踪****

面貌跟踪是指对被检测到的面貌进行动态目标跟踪。具体采用基于模型的方法或基于运动与模型相结合的方法。此外，利用肤色模型跟踪也不失为一种简单而有效的手段。

****(3)人脸比对****

面貌比对是对被检测到的面貌像进行身份确认或在面像库中进行目标搜索。这实际上就是说，将采样到的面像与库存的面像依次进行比对，并找出最佳的匹配对象。所以，面像的描述决定了面像识别的具体方法与性能。

## **2.3. 用摄像头获取颜值的代码实现**

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Thu Oct 18 13:02:42 2018

@author: james

"""

import cv2

import requests

import json

import threading

import time

import base64

import numpy as np

import xugu

servo=xugu.Servo(12)

access\_token="24.42017d996da0a23f3c4567b8dfd16a21.2592000.1550220300.282335-14563606"

API\_KEY="ijoqlG1PdSsdxtXc7DNn68jh"

SECRET\_KEY="Erpr184wiWCG7ZZglFKKG3Zm3up6xUmi"

face\_num=0

frame=None

now\_time=0

face\_info={}

def cvimg\_to\_b64(img):

try:

image = cv2.imencode('.jpg', img)[1]

base64\_data = str(base64.b64encode(image))[2:-1]

return base64\_data

except Exception as e:

return "error"

def face\_search(img64):

url="https://aip.baidubce.com/rest/2.0/face/v3/search"

url=url+"?access\_token="+access\_token

data={

"image":img64,

"image\_type":"BASE64",

"group\_id\_list":"test\_group"}

try:

response=requests.post(url,files=None,data=data)

res\_text=response.text

res\_json=json.loads(res\_text)

return res\_json

except Exception:

return "error"

def get\_ai\_access\_token():

url="https://aip.baidubce.com/oauth/2.0/token?grant\_type=" +\

"client\_credentials&client\_id=%s&client\_secret=%s" %(API\_KEY,SECRET\_KEY)

try:

response=requests.get(url)

res\_text=response.text

res\_json=json.loads(res\_text)

return str(res\_json["access\_token"])

except Exception:

return "error"

def get\_face\_info(img64):

url="https://aip.baidubce.com/rest/2.0/face/v3/detect"

url = url + "?access\_token=" + access\_token

data = {"face\_field":"age,beauty,expression,faceshape,gender,glasses,landmark,race,qualities",

"image\_type":"BASE64","image":img64,"max\_face\_num":5}

try:

response = requests.post(url,data=data)

res\_text=response.text

res\_json=json.loads(res\_text)

return res\_json

except Exception:

return "error"

def post\_request(frame,face\_num,nt):

global face\_info

if(face\_num>0) and (time.time()-nt>3):

global now\_time

now\_time=time.time()

img64=cvimg\_to\_b64(frame)

res=get\_face\_info(img64)

try:

status=res['error\_msg']

except Exception:

status=""

if(status=="SUCCESS"):

face\_info['gender']= res['result']['face\_list'][0]['gender']['type']

face\_info['age']=res['result']['face\_list'][0]['age']

face\_info['beauty']=res['result']['face\_list'][0]['beauty']

return

else:

face\_info={}

def faceDetect(img,face\_cascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')):

size=img.shape[:2]

divisor = 8

h,w=size

minSize=(w//divisor,h//divisor)

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, 1.2, 1,cv2.CASCADE\_SCALE\_IMAGE,minSize)

for (x,y,w,h) in faces:

cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)

return img,len(faces)

def put\_Text(cvimg,text,location,size=2):

cvimg=cv2.putText(cvimg, text, location,cv2.FONT\_HERSHEY\_SIMPLEX, size, (51, 102, 255), 3)

return cvimg

def check\_token():

print(int(time.time()) - 1547628298)

if int(time.time()) - 1547628298 >= 2592000:

global access\_token

token = get\_ai\_access\_token()

if token != "error":

access\_token = get\_ai\_access\_token()

def main():

check\_token()

cap = cv2.VideoCapture(0)

global now\_time

now\_time=time.time()

while(True):

global face\_info

global servo

ret, frame = cap.read()

if ret == True:

frame1,face\_num = faceDetect(frame)

frame1=cv2.flip(frame1,1,dst=None)

frame1=cv2.resize(frame1,(1280,800),interpolation=cv2.INTER\_LINEAR)

t=threading.Thread(target=post\_request,args=(frame,face\_num,now\_time,), name='POST\_REQUEST')

t.start()

print(face\_info)

if face\_info=={} :

frame1=put\_Text(frame1,"Waiting...",(50,50))

else:

try:

frame1=put\_Text(frame1,str(int(face\_info['age'])),(300,50))

frame1=put\_Text(frame1,str(face\_info['gender']),(300,120))

frame1=put\_Text(frame1,str(int(face\_info['beauty'])),(300,190))

frame1=put\_Text(frame1,"Age:",(50,50))

frame1=put\_Text(frame1,"Gender:",(50,120))

frame1=put\_Text(frame1,"Beauty:",(50,190))

servo.write\_angle(int(face\_info['beauty']\*3))

except Exception:

pass

cv2.imshow('Magic Image',frame1)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

main()