**目录**

[百度抠图总流程 3](#_Toc533868921)

[1、 原理介绍 3](#_Toc533868922)

[2、 总的流程图 3](#_Toc533868923)

[第二节：遍历文件夹的子文件 3](#_Toc533868924)

[1、 功能 3](#_Toc533868925)

[2、 代码 4](#_Toc533868926)

[3、 接口描述 4](#_Toc533868927)

[第三节：抠图 4](#_Toc533868928)

[1、 功能 4](#_Toc533868929)

[2、 流程图 4](#_Toc533868930)

[3、 代码 5](#_Toc533868931)

[4、 接口描述 8](#_Toc533868932)

[第四节：人脸检测 8](#_Toc533868933)

[1、 功能 8](#_Toc533868934)

[2、 代码 8](#_Toc533868935)

[3、 接口描述 8](#_Toc533868936)

百度SDK抠图程序

杭州图铭科技有限公司

2018.12

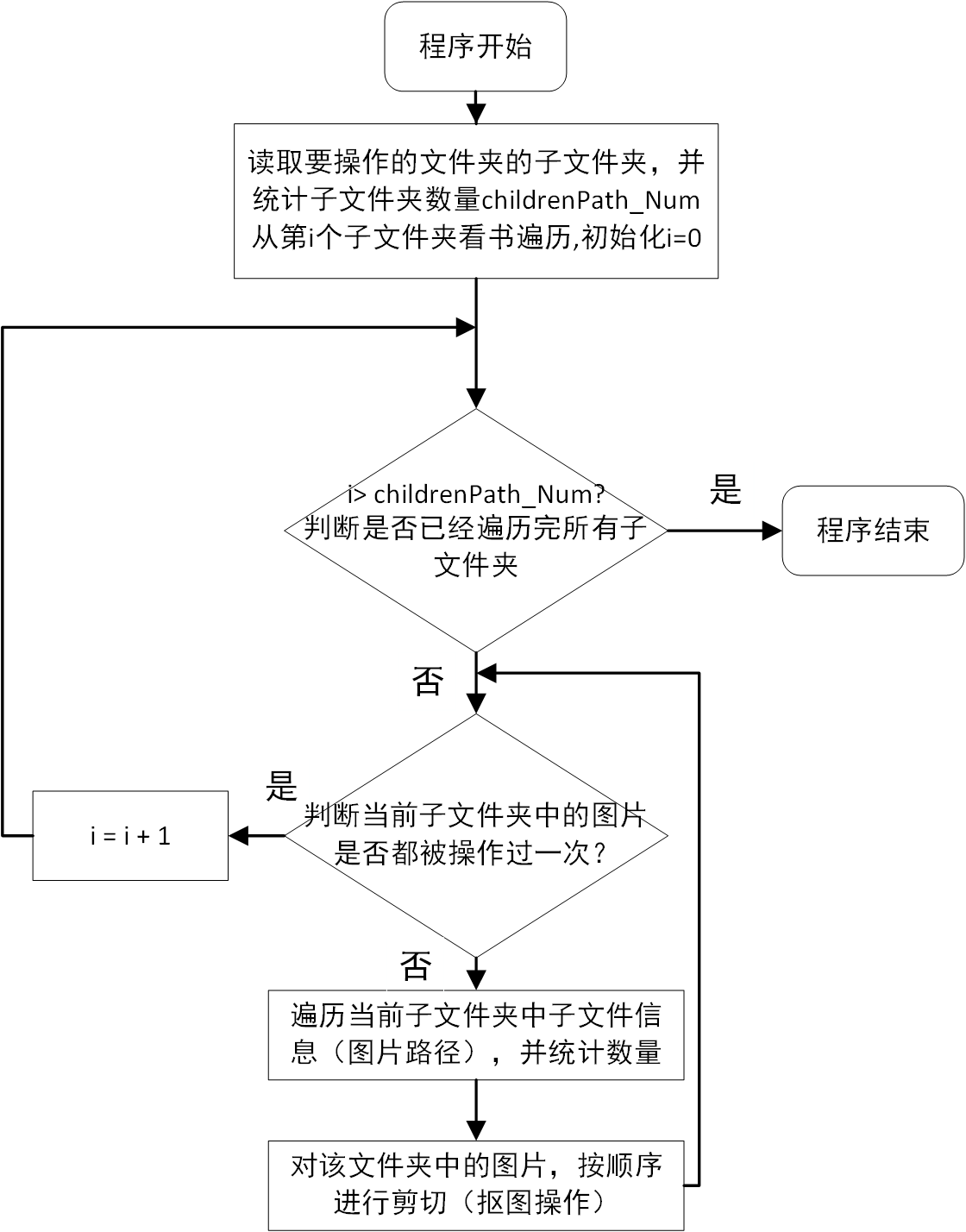
# 百度抠图总流程

1. 原理介绍

**利用百度开发的人脸识别Python SDK进行人脸检测，以获得图片中人脸数量以及他们的人脸框信息，再利用opencv2将每张人脸从原图上抠下来，并保存到目的文件夹中。**

**实现程序在链接中：**[**https://github.com/xuaikun/BaiduSDK\_FaceMatch**](https://github.com/xuaikun/BaiduSDK_FaceMatch)

1. 总的流程图



# 第二节：遍历文件夹的子文件

1. 功能

**读取当前路径的所有子文件的数量和路径**

1. 代码

**def fileInFolder(filepath):**

**pathDir = os.listdir(filepath) # 获取filepath文件夹下的所有的文件**

**files = []**

**for allDir in pathDir:**

**child = os.path.join('%s\\%s' % (filepath, allDir))**

**files.append(child.decode('gbk')) # .decode('gbk')是解决中文显示乱码问题**

**# print child**

**# if os.path.isdir(child):**

**# print child**

**# simplepath = os.path.split(child)**

**# print simplepath**

**return files**

1. 接口描述

**深度优先遍历图片代码接口**

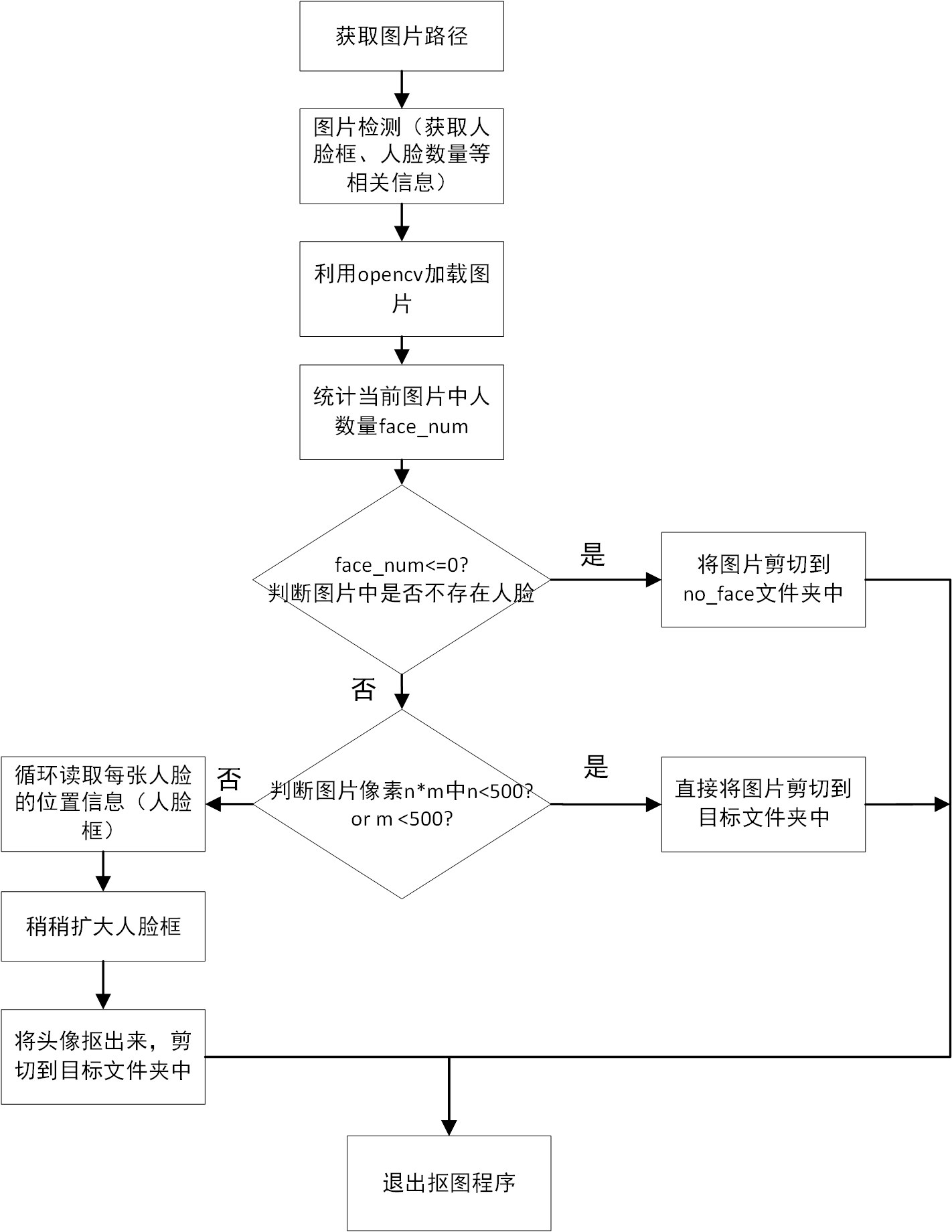
|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **In/out/Par** | **Size/precision** | **Description/value** |
| **Filepath** | **in** | **string** | **输入需要操作的文件路径** |

# 第三节：抠图

1. 功能

**利用百度人脸识别Python SDK检测图像中人脸数量已经人脸框信息，再利用opencv对人脸框进行抠图。**

1. 流程图



1. 代码

**def cut\_photo(photo\_name):**

**# 将一张图片放进来就能将图片进行处理成单个头像的图片**

**# image1 = "F:\\photo\_test\\mul\_face\\00000.jpg"**

**error\_Falg = True**

**image1 = photo\_name**

**result = face\_detection(image1)**

**try:**

**result = result.get("result")**

**face\_list = result.get("face\_list")**

**except :**

**print "进行任意异常处理"**

**error\_Falg = False**

**else:**

**# 没有异常**

**count = 0**

**SplitPath = os.path.split(image1) # 将需要处理的文件夹的名字提取出来**

**# SplitPath[0] 路径名称的前半部分**

**# SplitPath[1] 路径名称的最后一级**

**# SplitPath\_new = os.path.split(SplitPath[0]) # 将需要处理的文件夹的名字提取出来**

**img = cv2.imread(image1) # 读取图片**

**# print "img.shape[0] =", img.shape[0] # 622 y ~ top**

**# print "img.shape[1] =", img.shape[1] # 534 x ~ left**

**if len(face\_list) <= 0:**

**# 图片不存在人脸，直接删除**

**no\_face = os.path.join(resultPath, 'no\_face')**

**isExist = os.path.exists(no\_face)**

**if not isExist:**

**print "不存在该路径，创建对应路径"**

**os.makedirs(no\_face)**

**shutil.move(image1, no\_face)**

**else:**

**# 保证一定存在人脸**

**if img.shape[0] <= 500 or img.shape[1] <= 500:**

**# 图片已经在之前被处理过了，不需要再次处理，直接剪切到对应文件夹中**

**single\_face = os.path.join(resultPath, 'single\_face')**

**isExist = os.path.exists(single\_face)**

**if not isExist:**

**print "不存在该路径，创建对应路径"**

**os.makedirs(single\_face)**

**shutil.move(image1, single\_face)**

**else:**

**# 图片未经过操作，需要进行抠图**

**# face\_list 保存着几个人脸的位置信息**

**for i in range(0, len(face\_list)):**

**get\_list = face\_list[i]**

**# 获取某个人的人脸位置信息**

**location = get\_list.get("location")**

**# 将人脸位置信息具体化**

**width = location.get("width") # w**

**top = location.get("top") # y**

**height = location.get("height") # h**

**left = location.get("left") # x**

**# 将图像的相关像素整形化**

**x = int(left)**

**y = int(top)**

**w = int(width)**

**h = int(height)**

**try:**

**# 测试异常**

**# 为了避免操作图片时超过像素**

**Y = y - int(y\*0.2)**

**if Y < 0:**

**Y = 0;**

**Y\_H = y + h + int(y\*0.18)**

**if Y\_H > img.shape[0]:**

**Y\_H = img.shape[0]**

**X = x - int(x\*0.03)**

**if X < 0:**

**X = 0**

**X\_W = x + w + int(x\*0.03)**

**if X\_W > img.shape[1]:**

**X\_W = img.shape[1]**

**# 将图片从原图上抠出来，把外部边框扩大一点**

**f = img[Y:Y\_H, X:X\_W]**

**except :**

**print "进行任意异常处理"**

**# 说明这个图像不清晰，先暂时不处理**

**break**

**else:**

**# 未发生异常**

**# 重新命名时，可以用时间戳+计数值**

**partPath = [str(int(time.time())), str(count)]**

**NewPath = os.path.join(resultPath, 'single\_face')**

**isExist = os.path.exists(NewPath)**

**if not isExist:**

**print "不存在该路径，创建对应路径"**

**os.makedirs(NewPath)**

**# 新的图片命名**

**New\_Name = partPath[0] + partPath[1] + SplitPath[1]**

**NewPath = os.path.join(NewPath, New\_Name)**

**# 将抠出的图片进行保存**

**cv2.imwrite(NewPath, f)**

**count = count + 1**

**operation\_ok = os.path.join(resultPath, 'operation\_ok')**

**isExist = os.path.exists(operation\_ok)**

**if not isExist:**

**print "不存在该路径，创建对应路径"**

**os.makedirs(operation\_ok)**

**shutil.move(image1, operation\_ok)**

**return error\_Falg**

1. 接口描述

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **In/out/Par** | **Size/precision** | **Description/value** |
| **Photo\_name** | **in** | **string** | **图片路径** |

# 第四节：人脸检测

1. 功能

**获取图像中人脸的数量以及对应人脸的人脸框等信息。**

1. 代码

**def face\_detection(image\_name):**

**image = base64.b64encode(open(image\_name, 'rb').read())**

**image\_type = "BASE64"**

**"""调用人脸检测"""**

**options = {}**

**options["max\_face\_num"] = 10**

**options["face\_field"] = "location" # age和gender之间只能存在英文逗号不能存在空格**

**result = client.detect(image, image\_type, options)**

**return result**

1. 接口描述

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **In/out/Par** | **Size/precision** | **Description/value** |
| **Image\_name** | **in** | **string** | **图片路径** |