第一次作业

3140102743 李彦瑞

1、实验目的和要求:

对输入的一个彩色视频与五张以上照片,用OpenCV 实现以下功能或要求:

- 1)命令行格式: "xxx. exe 放视频与照片的文件夹路径", (例如 MyMakeVideo. exe
- C:\input) 【假设该文件夹下面只有一个 avi 视频文件与若干jpg文件】
- 2) 将输入的视频与照片处理成同样长宽后,合在一起生成一个视频;
- 3) 这个新视频中,编程生成一个片头,然后按幻灯片形式播放这些输入照片, 最后按视频原来速度播放输入的视频:
- 4) 新视频中要在底部打上含自己学号与姓名等信息的字幕:
- 5) 有能力的同学,可以编程实现镜头切换效果;
- 6) 视频文件无需上载 , 但在实验报告里必须贴图展示输入输出效果

2、实验环境

Python3. 6+opencv3

3、实验 步骤

1) 考虑到要使用命令行参数, 所以首先了解了有关怎么读取命令行参数

命令行参数的读取

命令行参数的读取采用了 Python 中的 sys 模块以及 getopt 模块

if __name__ == "__main__":

main(sys.argv[1:])

以上两行程序表示当该文件被当做主函数运行时,调用 main 函数并把 sys. argv 中的参数传入 main 函数,其中 sys. argv [0]是打开文件的文件名,之后的为命令行参数。

输入的 sys. argv[]在 main 函数中被调用,调用中使用了 getopt,调用方法如下 def main(argv):

opts, args = getopt.getopt(argv, "")

其中返回的 opts 为操作名,在一般的编程中可能是 - h, -o 之类的字符;返回的 args

为操作名之后跟的参数,在本次作业中,没有操作名,因此 opts 返回为空,args 返回为地址。

2) 获得了地址,接下来需要通过这个地址得到地址中的图片与视频

```
这主要用了 python 中的 os 模块
    以下是获得图片文件的程序:
Const_Image_Format = [".jpg", ".jpeg", ".bmp", ".png"]
定义图片的后缀:
class FileFilt:
    def __init__(self):
        self.fileList = []
        self.counter = 0
    def FindFile(self, dirr, fileformat, filtrate = 1):
        for s in os. listdir(dirr):
            newDir = os.path.join(dirr, s)
            if os. path. isfile (newDir):
                if filtrate:
                    if newDir and (os. path. splitext (newDir) [1] in fileformat):
                        self. fileList. append (newDir)
                        self.counter += 1
                else:
                    self.fileList.append(newDir)
```

定义了一个 FileFilt 类,其中主要有两个属性,一个是获得的文件的地址的列表 self.fileList,另一个是获得文件的数量 self.counter

定义了方法 FindFile,调用了给的地址 dirr,用 s 遍历该地址下的所用文件,用 (os. path. splitext(newDir))将文件的后缀提取出来并与之前定义的图像的后缀对比,如果符合则将该文件以及之前的地址合并后加入 fileList 中。

```
最后在主程序中
finder=FileFilt()
finder.FindFile(dirr=args[0], fileformat=Const Image Format)
```

self.counter+=1

```
imglist=finder.fileList
令 imglist 为返回的地址的列表
```

```
视频文件的获取方法类似。
3) 获得文件后用 opency 将图片和视频播放,并且完成要求
获得文件地址列表为 imglist, vidlist
将两者分别传入处理函数 imgprocess 和 vidprocess 中
定义 imgprocess 和 vidprocess 如下
def imgprocess(imglist):
   for picd in imglist:
        thispic = cv2. imdecode (np. fromfile (picd, dtype=np. uint8), 1)
       thispic=cv2.resize(thispic, (500, 400))
       cv2. imshow("play", thispic)
       cv2. waitKey(1000)
def vidprocess(vidlist):
    for vid in vidlist:
       videoCapture = cv2. VideoCapture(vid)
       fps = videoCapture.get(cv2.CAP PROP FPS)
       success, frame = videoCapture.read()
       display=cv2. resize(frame, (500, 400))
       while success:
            display=cv2. resize (frame, (500, 400))
            cv2. putText(display, "liyanrui
3140102743", (150, 300), cv2. FONT_HERSHEY_PLAIN, 1. 3, (0, 0, 255), 2)
           cv2. imshow("play", display) #显示
            cv2.waitKey(int(1000/int(fps))) #延迟
```

success, frame = videoCapture.read() #获取下一帧

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

图片处理:

由于输入的地址可能包含中文,如果含有中文用 imread 是无法打开的,因此在读取文件的时候采用了 cv2. imdecode (np. fromfile (picd, dtype=np. uint8), 1),这个方法可以打开地址中有中文的图像文件。

之后进行了调整大小,然后输出,再用 cv2. waitKey (1000) 等待 1s 之后换另一张图片。

视频处理:

用 videoCapture = cv2. VideoCapture (vid) 将视频文件读入。

之后用 fps = videoCapture.get(cv2.CAP_PROP_FPS)获得读入视频原始的 FPS 以便之后的等速度输出。

之后用 success, frame = videoCapture.read()

获取每一帧图像, success 返回是否读到, frame 返回图像, 之后用 resize 改变大小, 再用 cv2. putText 放置文字。在放文字时, 发现皱纹不能显示, 网上搜索可以用 freetype 将中文转化成位图再放到视频上。

之后用 imshow 显示图像,再 cv2. waitKey(int(1000/int(fps)))等待相应的时间最后增加了

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

break

使得播放一半时可以按 q 退出

4) 打包 py 文件为 exe

使用模块 pyinstaller 在 cmd 中输入 pyinstaller xxx. py 运行后生成一个文件夹,

 certifi	<u></u> Include	∥ lib2to3
_cffi_backend.pyd	_ctypes.pyd	_decimal.pyd
_multiprocessing.pyd	_socket.pyd	_ssl.pyd
api-ms-win-core-datetime-l1-1-0	api-ms-win-core-debug-I1-1-0.dll	api-ms-win-core-errorhandling-l
api-ms-win-core-file-I2-1-0.dll	api-ms-win-core-handle-l1-1-0.dll	api-ms-win-core-heap-I1-1-0.dll
api-ms-win-core-localization-l1-2	api-ms-win-core-memory-I1-1-0	api-ms-win-core-namedpipe-l1-1
api-ms-win-core-processthreads	api-ms-win-core-profile-I1-1-0.dll	api-ms-win-core-rtlsupport-l1-1-
api-ms-win-core-synch-l1-2-0.dll	api-ms-win-core-sysinfo-l1-1-0.dll	api-ms-win-core-timezone-l1-1-0
api-ms-win-crt-convert-l1-1-0.dll	api-ms-win-crt-environment-l1-1	api-ms-win-crt-filesystem-l1-1-0
api-ms-win-crt-math-l1-1-0.dll	api-ms-win-crt-multibyte-l1-1-0.dll	api-ms-win-crt-process-l1-1-0.dll
api-ms-win-crt-string-l1-1-0.dll	api-ms-win-crt-time-l1-1-0.dll	api-ms-win-crt-utility-l1-1-0.dll
a mfc140u.dll	MSVCP140.dll	🗟 numpy.core.multiarray.pyd
違 numpy.linalgumath_linalg.pyd	🗟 numpy. linalg. lapack_lite.pyd	🗟 numpy.random.mtrand.pyd
a opencv_core330.dll	a opencv_dnn330.dll	a opencv_features2d330.dll
a opencv_imgcodecs330.dll	a opencv_imgproc330.dll	a opencv_ml330.dll
a opencv_shape330.dll	a opencv_stitching330.dll	a opencv_video330.dll
PyQt5.Qt.pyd	PyQt5.QtCore.pyd	PyQt5.QtGui.pyd
a python3.dll	a python36.dll	g pythoncom36.dll
Qt5Gui.dll	Qt5PrintSupport.dll	Qt5Svg.dll
📝 select.pyd	shi.exe	shi.exe.manifest
📝 unicodedata.pyd	VCRUNTIME140.dll	🗟 win 32 api. pyd
📝 win 32 trace. py d	🖟 win32ui.pyd	win32wnet.pyd

用 cmd 进入这个文件夹运行 xxx. exe 加上存放图片和视频的文件夹的地址可以成功运

4、实验结果

行。





第一幅图是图片的展示, 之后是视频的展示。

5、讨论与分析

这一次的作业,主要是让我学会了使用很多的 python 的模块,并学会了简单的对图片, 视频文件的读入, 为之后的学习打下了基础, 也增强了我的编程能力。