Make a personal video

—Homework1 for computer vision

Name: Wang Jun

Student ID：3170100186

Date: 2021-11-23

Course: Computer Vision

Instructor: Mingli Song

Chapter 1: The purpose and requirements of the experiment

For the input of a color video and five or more photos, use OpenCV to achieve the following functions or requirements:

- Command format: "xxx.exe is the folder path of videos and photos",(for example, MyMakeVideo.exe C:\input)[Suppose there is only one avi video file and if jpg file under this folder]
- After processing the input video and photo into the same width, combine them together to create a video;
- In this new video, program to create a header, and then play these input photos in the form of a slide show, and finally press the original video and play the input video at speed;
- In the new video, subtitles containing information such as student ID and name should be added at the bottom;
- There are capable students who can program to achieve the lens switching effect;

The video file does not need to be uploaded, but the experiment report must be pasted to show the input and output effect

Chapter 2: Experimental content and principle

OpenCV

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

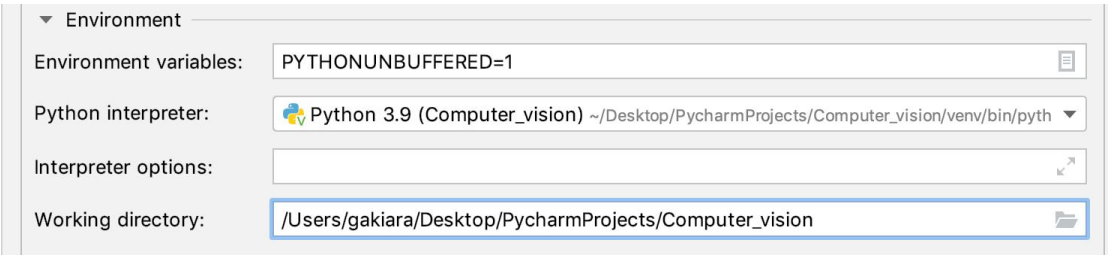
Chapter 3: Experimental procedure and analysis

Install opencv-python

Install opencv-python for the next process

```
(venv) gakiaradeMacBook-Pro:Computer_vision gakiara$ pip install opencv-python
Collecting opencv-python
  Downloading opencv-python-4.5.4.60.tar.gz (89.8 MB)
     |▉▉                              | 8.6 MB 47 kB/s eta 0:28:32
```

Set the working directory:



Use:

```
dirname = "./resources/"
```

Read the resources

-According to the input folder path, read in resource files (a video file and multiple picture files)

-Video files must be in .avi format, picture files in .jpg format, no need to specify the number of pictures

-Read some parameters of the video: height, width, fps, framecount

Load video

```
## load video
videoFile = gb.glob(dirname+"*.avi")
capture = cv2.VideoCapture(videoFile[0])
height = capture.get(cv2.CAP_PROP_FRAME_HEIGHT)
width = capture.get(cv2.CAP_PROP_FRAME_WIDTH)
fps = capture.get(cv2.CAP_PROP_FPS)
framecount = capture.get(cv2.CAP_PROP_FRAME_COUNT)
```

Load Image:

```
## load images
Images=[]
imgfile = gb.glob(dirname+"*.jpg")
for file in imgfile:
    img = cv2.imread(file)
    Images.append(img)
```

Re-size each image

-Get the width and height data of the video

-Use cv2.INTER_NEAREST interpolation method to scale the picture

```
## resize images
for i in range(len(Images)):
    Images[i] = cv2.resize(Images[i], (int(width),int(height)),
                           interpolation=cv2.INTER_NEAREST)
```

Write a new video file

-After subtitles each picture, add a dissolving effect to the two pictures and add it to the new video file

-Add subtitles to each frame of the original video and add it to the new video file

write videofile:

```
## write videofile
videoW = cv2.VideoWriter(dirname+'output.avi',
                         cv2.VideoWriter_fourcc('M','J','P','G'), int(fps),
                        (int(width),int(height)),True)
fps1 = int(fps)
```

write images:

```
## write images
text = '%s %s' % ('3170100186', 'WANG JUN')
WAIT = fps1
for i in range(len(Images)*fps1*2):
    num = i//(fps1*2)
    if num<len(Images)-2:
        weight = (i-num*fps1*2) / WAIT
        img = cv2.putText(Images[num], text, (int(width/2-400),int(height-150)),
                          cv2.FONT_HERSHEY_SIMPLEX, 2, (202,235,216), 4)
        img1 = cv2.putText(Images[num+1], text, (int(width/2-400),int(height-150)),
                           cv2.FONT_HERSHEY_SIMPLEX, 2, (202,235,216), 4)
        res = cv2.addWeighted(img, 1-weight, img1, weight, 0)
    else:
        res = cv2.putText(Images[num-1], text, (int(width/2-400),int(height-150)),
                          cv2.FONT_HERSHEY_SIMPLEX, 2, (202,235,216), 4)
    videoW.write(res)
```

write video:

```
# write video
while(True):
    ret,video = capture.read()
    if ret:
        cv2.putText(video, text, (int(width/2-400),int(height-150)),
                    cv2.FONT_HERSHEY_SIMPLEX, 2, (202,235,216), 4)
        videoW.write(video)
    else:
        break
videoW.release()
```

Chapter 4: Experimental environment and operation method

Development environment

- macOS 10.14.6

- python 3.9

- opencv-python 4.5.1.48

Operation mode

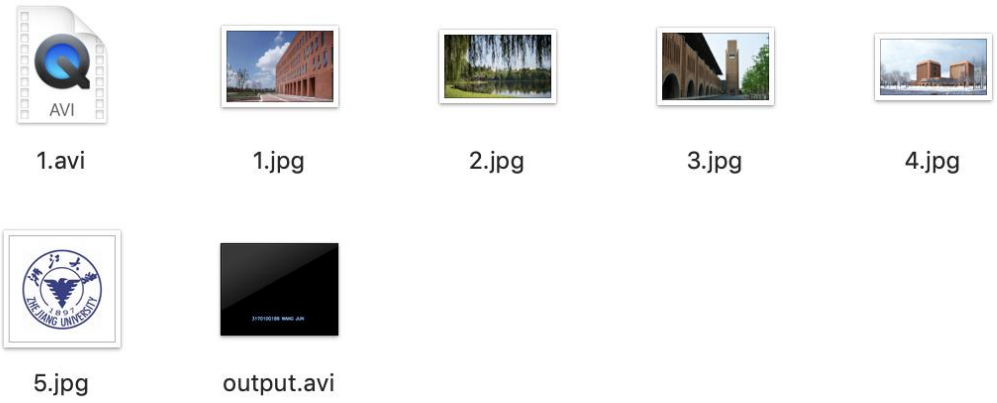
- python hw1.py　./resources/ 或者根据文件夹选择

-python hw1.py　不输入参数默认是./resources 文件夹

Chapter 5: Experimental results

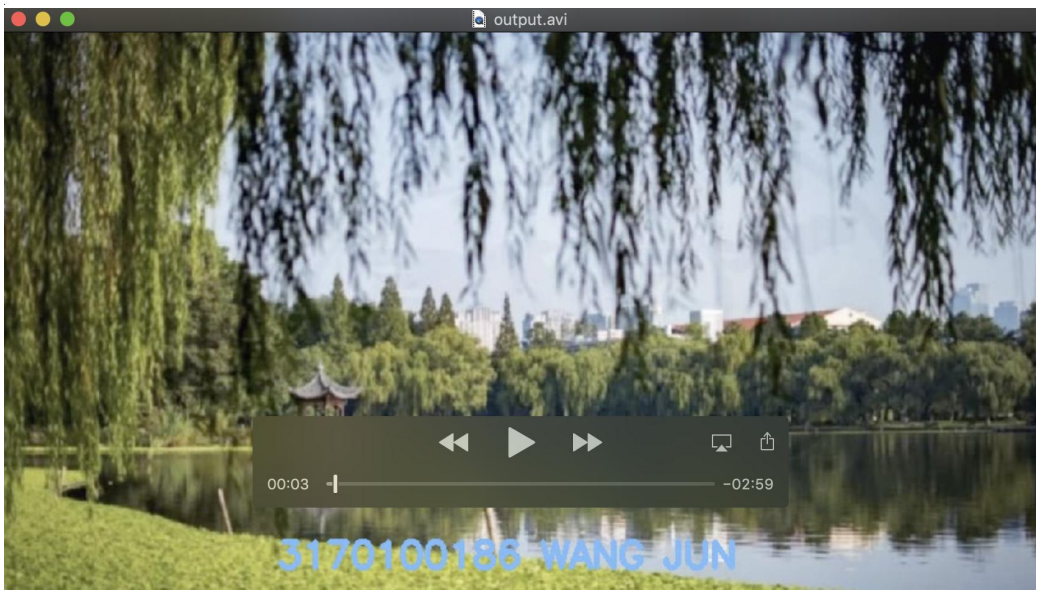
IO Screenshots

INPUT:




output.avi
00:00
-03:02
3170100186 WANG JUN


output.avi
00:01
-03:01
3170100186 WANG JUN


output.avi
00:03
-02:59
3170100186 WANG JUN


output.avi
00:05
-02:57
3170100186 WANG JUN


output.avi
「 将 以 你 的 名 字 开 启 」
00:29
-02:33
3170100186 WANG JUN


output.avi
00:56
-02:06
3170100186 WANG JUN

Chapter 6: Thoughts

OpenCV in python and C++

I use mac-os to finish my work, and find the visual studio in mac is hard to use in C++, so I try to use python instead. And surprisingly, python is much easier than C++ that not only in its readable, but also it did not require you to learn about a new class Mat to be able to operate on images. In a nutshell, for man using a mac os, python would be a better choice, and the deployment of environment is easier.