专业：计算机科学与技术
姓名：卢佳盈
学号：3180103570
日期：2020/12/5

浙江大学实验报告

课程名称：计算机视觉 指导老师：宋明黎 成绩：

实验名称：制作个人视频

一、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；

二、实验内容和原理

（简述实验有关的基本原理）

2.1 开发环境

- Windows X64
- Visual Studio 2017
- opencv-3.4.0

2.2 运行方式

命令行输入 Project1.exe C:/input

目标文件夹请使用反斜杠

```
D:\openCV\code\Project1\x64\Debug>Project1.exe  D:/openCV/code/Project1/input
```

2.3 原理介绍

【视频信号获取】

视频信号是重要的视觉信息来源。视频由一系列图像构成，这些图像称为帧。帧以固定的时间间隔获取（称为帧速率，通常用帧/秒表示）。读取视频序列。要从视频序列读取帧，只需创建一个 cv::VideoCapture 类的实例，然后再一个循环中提取并显示视频的每帧。

【视频信息获取】

在本次试验中，需要采集视频的长、宽、帧率等信息，采 videoCapture.get()函数，基本参数如下

param	define
cv2.VideoCapture.get(0)	视频文件的当前位置（播放）以毫秒为单位
cv2.VideoCapture.get(1)	基于以0开始的被捕获或解码的帧索引
cv2.VideoCapture.get(2)	视频文件的相对位置（播放）：0=电影开始，1=影片的结尾。
cv2.VideoCapture.get(3)	在视频流的帧的宽度
cv2.VideoCapture.get(4)	在视频流的帧的高度
cv2.VideoCapture.get(5)	帧速率
cv2.VideoCapture.get(6)	编解码的4字-字符代码
cv2.VideoCapture.get(7)	视频文件中的帧数
cv2.VideoCapture.get(8)	返回对象的格式
cv2.VideoCapture.get(9)	返回后端特定的值，该值指示当前捕获模式
cv2.VideoCapture.get(10)	图像的亮度(仅适用于照相机)
cv2.VideoCapture.get(11)	图像的对比度(仅适用于照相机)
cv2.VideoCapture.get(12)	图像的饱和度(仅适用于照相机)
cv2.VideoCapture.get(13)	色调图像(仅适用于照相机)
cv2.VideoCapture.get(14)	图像增益(仅适用于照相机)（Gain在摄影中表示白平衡提升）
cv2.VideoCapture.get(15)	曝光(仅适用于照相机)
cv2.VideoCapture.get(16)	指示是否应将图像转换为RGB布尔标志
cv2.VideoCapture.get(17)	× 暂时不支持
cv2.VideoCapture.get(18)	立体摄像机的矫正标注（目前只有DC1394 v.2.x后端支持这个功能）

【字幕添加】

字幕功能的实现需要使用 putText 函数，API 如下：

void putText(Mat& img, const string& text, Point org, int fontFace,double fontScale， Scalar color, int thickness=1, int lineType=8);

参数 1：Mat& img，待写字的图片

参数 2：const string& text，待写入的字

参数 3：Point org， 第一个字符左下角坐标

参数 4：int fontFace，字体类型，FONT_HERSHEY_SIMPLEX ，FONT_HERSHEY_PLAIN ，FONT_HERSHEY_DUPLEX 等

参数 5：，double fontScale，字体大小，我们设置为 2 号

参数 6：Scalar color，字体颜色，颜色用 Scalar（）表示

参数 7：int thickness，字体粗细

参数 8：int lineType，线型

【转场效果】

使用融合函数 addWeighted，将两张相同大小，相同类型的图片融合，API 如下

void cvAddWeighted(const CvArr* src1, double alpha,const CvArr* src2, double beta,double gamma, CvArr* dst);

参数 1：src1，第一个原数组.

参数 2：alpha，第一个数组元素权重

参数 3：src2 第二个原数组

参数 4：beta，第二个数组元素权重

参数 5：gamma，图 1 与图 2 作和后添加的数值

参数 6：dst，输出图片

三、实验步骤与分析

（每个步骤结合对应部分的源代码分析）

3.1 读入视频、获取基本信息

```
//load video
vector<String> videoAddrs = get_image_names(dir+"/*.avi");
String Addr = dir+"/"+videoAddrs[0];
cv::VideoCapture capture(Addr);
double height = capture.get(CAP_PROP_FRAME_HEIGHT);
double width = capture.get(CAP_PROP_FRAME_WIDTH);
double fps = capture.get(CAP_PROP_FPS);
double framecount = capture.get(CAP_PROP_FRAME_COUNT);
```

3.2 载入图片

依次读取目标文件夹中的图片，并将图片处理成与视频相同的长宽，在每张图片的相同位置添加字幕

```
//read images
cv::Size size = Size(width, height);
vector<String> imagesAddr = get_image_names(dir + "/*.jpg");
vector <cv::Mat> images;
for (int i = 0; i < imagesAddr.size(); i++) {
    cv::Mat image = cv::imread(dir+"/"+imagesAddr[i]);
    cv::resize(image, image, size);
    cv::putText(image, text, Point(width / 2 - 150, height - 50), FONT_HERSHEY_DUPL
EX, 2, Scalar(255, 255, 255), 4, 8, false);
    images.push_back(image);
}
```

其中调用的 get_image_names 函数，实现返回符合目标文件名的所有文件名

```
vector<String> get_image_names(string file_path) {
```

```
    vector<String>file_names;
    intptr_t hFile = 0;
    _finddata_t fileInfo;
    hFile = _findfirst(file_path.c_str(), &fileInfo);
    if (hFile != -1) {
        do {
            if ((fileInfo.attrib&_A_SUBDIR)) {
                continue;
            }
            else {
                file_names.push_back(fileInfo.name);
                cout << fileInfo.name << endl;
            }
        } while (_findnext(hFile,&fileInfo)==0);
        _findclose(hFile);
    }
    return file_names;
}
```

3.3 视频的写入

创建一个 VideoWriter，并设置视频属性

```
VideoWriter Writer(dir + "/output.avi", CV_FOURCC('D', 'I', 'V', 'X'), fps, size, t
rue);
```

将所有的图片写入，设置每张图片的停留间隔

在图片与图片间通过 cv::addWeight 函数产生转场过渡帧，实现转场效果

```
for (int i = 0; i < images.size(); i++) {
    for (int j = 0; j < 60; j++) {
        Writer.write(images[i]);
    }
    for (int j = 0; j < 40; j++) {
        if (i == images.size()-1) {
            Writer.write(images[i]);
        }
        else {
            cv::Mat midImg;
            cv::addWeighted(images[i], (40 - j) *0.025, images[i + 1], j *0.025, 3,
 midImg);
            Writer.write(midImg);
        }
    }
}
```

写入原视频，并添加字幕

```
cv::Mat frame;
```

```
capture >> frame;
for (; !frame.empty(); capture >> frame) {
    cv::putText(frame, text, Point(width / 2 - 150, height - 50), FONT_HERSHEY_DUPL
EX, 2, Scalar(255, 255, 255), 4, 8, false);
    Writer.write(frame);
}
```

四、实验结果

（展示实验用到的输入输出图像等）

【输入素材】

img1.jpg

img2.jpg

img3.jpg

img4.jpg

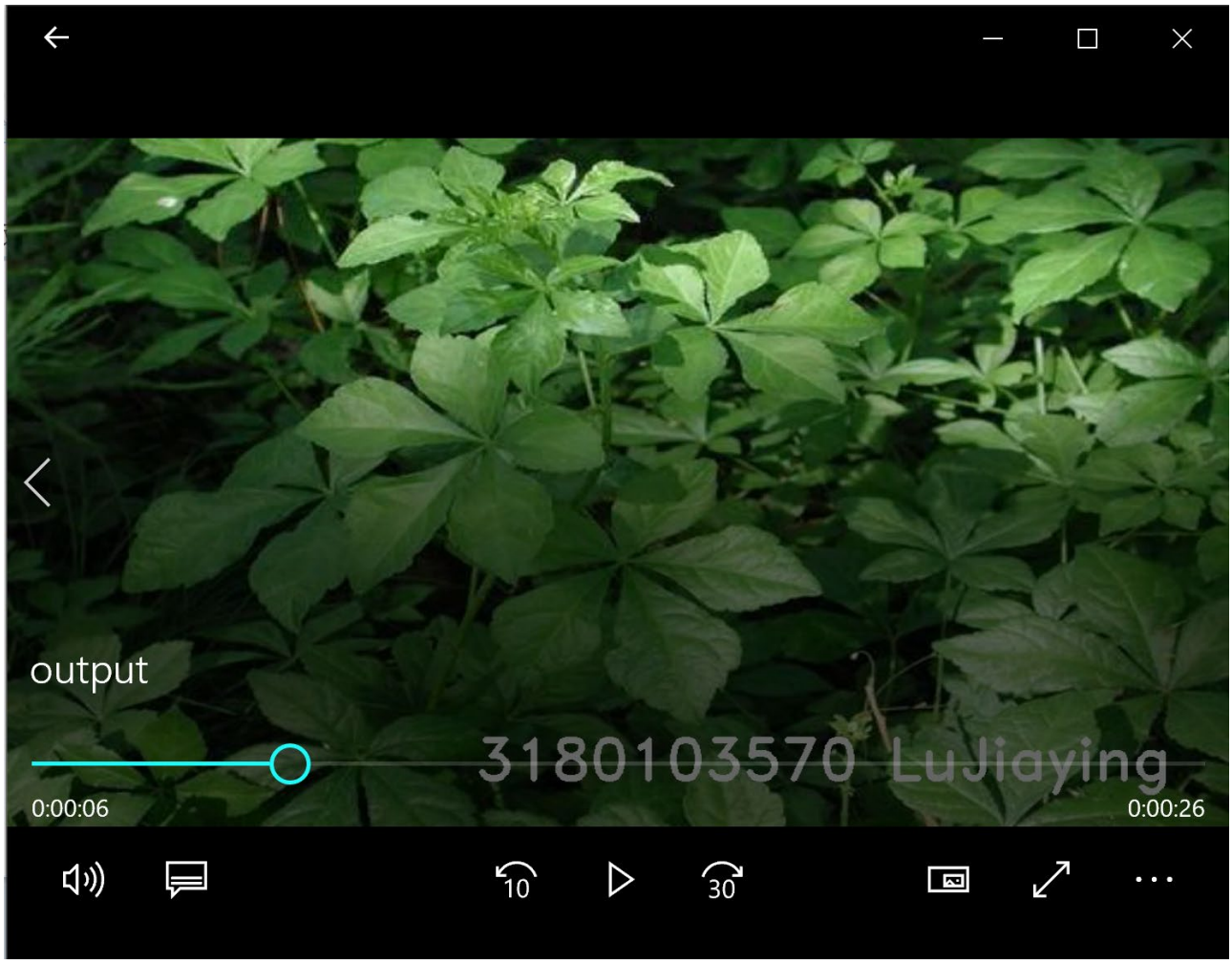
img5.jpg

img6.jpg

inputVideo.avi

【输出结果】

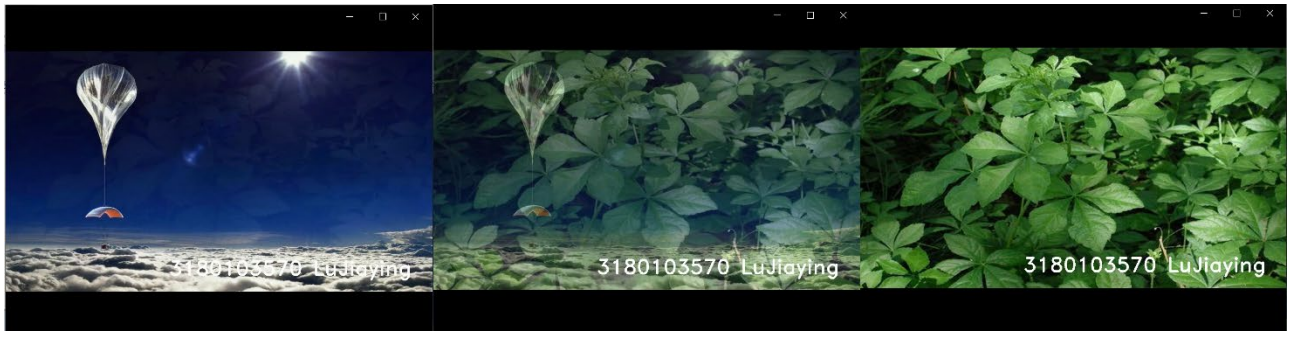
1. 实现字幕的添加



图片部分字幕

视频部分字幕

2. 转场效果



3. 合成视频

五、心得体会

在本次实验中，在配置完环境之后，我通过 opencv 实现了视频与图形的拼接处理，对于 opencv 的函数调用有了基本的认识，还编写代码实现了图片与图片之间的转场效果，为接下来的实验奠定一定基础。