Shot Cut

序号	学号	专业班级	姓名	性别
1	3181010940	数媒1801班	李沛瑶	女

1. Project Introduction

（1） 选题

本次 project 作业， 我选择的题目是《镜头切割》 。

（2） 工作简介，即要做什么事情

编写程序进行视频的镜头切割（2种以上算法）

程序输入：视频片段（mp4格式）

程序输出：切分后的镜头片段（jpg格式）

我选取的算法有：

绝对帧间差法

感知哈希法

基于聚类的关键帧提取

（3） 开发环境及系统运行要求，包括所用的开发工具、开发包、开源库、系统运行要求等

开发环境：python3.8

需要用到的包：

numpy　1.13.3

opencv　3.3.1

pillow　　8.0.1

scipy 1.5.2

2. Technical Details

（1） 工程实践当中所用到的理论知识阐述

1. 绝对帧间差法：

在实际实现时，将两个相邻的帧图像看做两个向量，使用np.linalg.norm()函数计算两个向量差的范式，也就是欧式距离，表示两个帧图像的差异。

2. 感知哈希法：

实验中，将每一帧图像分别转化为8*8大小的灰度图，并且比较在多少个像素点上，两帧相邻图像的差大于一个阈值，这个像素点的数量代表了两帧图像的差异，如果这个值大于一定的阈值，就判定该处需要镜头切割。

3. 图像相关系数法

依旧把帧图像看做向量，求两个相邻的图像的相似度，只需要使用余弦定理求两个向量的夹角，就可以将图像的相关性映射到 $0-1$ 之间，只需设置一个阈值，当相似度低于这个阈值时，判定为需要切割的点。

（2） 具体的算法，请用文字、示意图或者是伪代码等形式进行描述

1. 绝对帧间差法：

```
    pre_frame ← frames[0]
    for i ← 1 to frames_number:
        now_frame ← frames[i]
        diff[i] ← the norm of Vector(now_frame - pre_frame)
        pre_frame ← now_frame

    threshold ← the mean of diff * 1.75

    for i ← 1 to frames_number-1:
        if diff[i] > threshold & diff[i] > diff[i - 1] & diff[i] > diff[i + 1]:
            add i to set cut_id
```

2. 感知哈希法：

```
    threshold = 50

    pre_frame = frames[0]
    transform pre_frame to 8*8 gray picture
    for i ← 1 to frames_number:
        now_frame = frames[i]
        transform now_frame to 8*8 gray picture

        count ← 0
        for j ← 0 to 7:
            for k ← 0 to 7:
                if abs(pre_frame[j][k] - now_frame[j][k]) > 20:
                    count ← count + 1
        diff[i] ← count
        pre_frame ← now_frame

    for i ← 1 to frames_number-1:
        if diff[i] > threshold:
            add i to set cut_id
```

3. 图像相关系数法

```
    threshold = 0.5

    pre_frame = frames[0]
    for i ← 1 to frames_number:
        now_frame = frames[i]
        diff[i] ← corr2(now_frame,pre_frame)
        pre_frame ← now_frame

    for i ← 1 to frames_number-1:
        if diff[i] < threshold:
            add i to set cut_id
```

```
corr2(a, b):
    avec = a - mean(a)
    bvec = b - mean(b)

    r = Cosine value of angle of avec and bvec
    return r
```

（3） 程序开发中重要的技术细节

使用的重要函数：

1. VideoCapture() 函数

该函数来自opencv库，主要用于处理摄像头或者文件中的视频流

- VideoCapture()中参数是0，表示打开笔记本的内置摄像头
- 参数是视频文件路径则打开视频，如cap = cv2.VideoCapture("../test.mp4")

2、np.linalg.norm()函数

来源于numpy库，主要用于求范数，本实验中计算欧式距离时使用

函数原型：

```
np.linalg.norm(x, ord=None, axis=None, keepdims=False)
```

函数详解：

- x：输入矩阵
- ord：取值1，2，np.inf分别表示1范式，2范式，无穷范数
- axis：0，1，none默认，0代表按列处理，1代表按行处理
- keepdims：是否保持维数

3、np.convolve()函数

```
cv.VideoWriter(path, fourcc, fps, size)
```

来源于opencv库，用于创建新的视频文件并写出
函数详解：

- path：视频写出路径
- fourcc：视频编码格式
- fps：帧速率
- size：帧图像尺寸

3. Experiment Results

首先，使用的测试视频是popo.mp4，该视频一共有2691帧。

1. 绝对帧间差法

该方法由于计算过于简单，效果很差，因此在此不再赘述。

2. 感知哈希法：

如图，使用该算法的识别结果，对于较为剧烈的场景转换，识别效果比较优秀，主要的场景转换都能被识别到并且正确切割，但是该算法对黑场过渡的识别性不强，主要是因为黑场过渡过程中的相邻帧会整体发生像素值改变，如统一增加或者统一减少，就会造成被算法识别为变化过大而进行镜头切割。

3. 图像相关系数法

如图，可以看出，该算法对黑场过渡的检测效果非常好，分析认为这个原因是由于检测图像的相似性时，相邻两帧图像如果只在亮度上有区别，那他们的相关性将会非常高，并不会被切割开。但是该算法对同一个场景镜头中的剧烈运动变化检测效果不是很好，如下图，该镜头被从中间切分开来，主要是由于该镜头是一个车祸现场，场景内的景物变化非常大，运动剧烈，导致相邻帧图像相似度降低。