计算机图形学Project1程序说明

编程实现音乐节奏旋律的可视化

文件组织

> main.py 代码文件，运行main.py即可看到结果
>
> ACCA13.wav 进行可视化的音乐

环境依赖

> numpy 处理音频数据，进行傅里叶变换
>
> pygame 对频域信息进行可视化
>
> wave 读取wav格式音乐文件
>
> math 利用三角函数进行RGB设置
>
> time 获取当前时间
>
> pyaudio 进行音频流输入输出

代码说明

1.初始化

设置缓冲区大小，打开音频文件，设置窗口长和宽

```
CHUNK = 1024
wf = wave.open("ACCA13.wav", 'rb')
p = pyaudio.PyAudio()
stream = p.open(format=p.get_format_from_width(wf.getsampwidth()),
                channels=wf.getnchannels(),
                rate=wf.getframerate(),
                output=True)
window_height = 450
window_width = 850

pygame.init()
pygame.display.set_caption('实时频域')
screen = pygame.display.set_mode((window_width, window_height), 0, 32)
```

2.信号处理部分

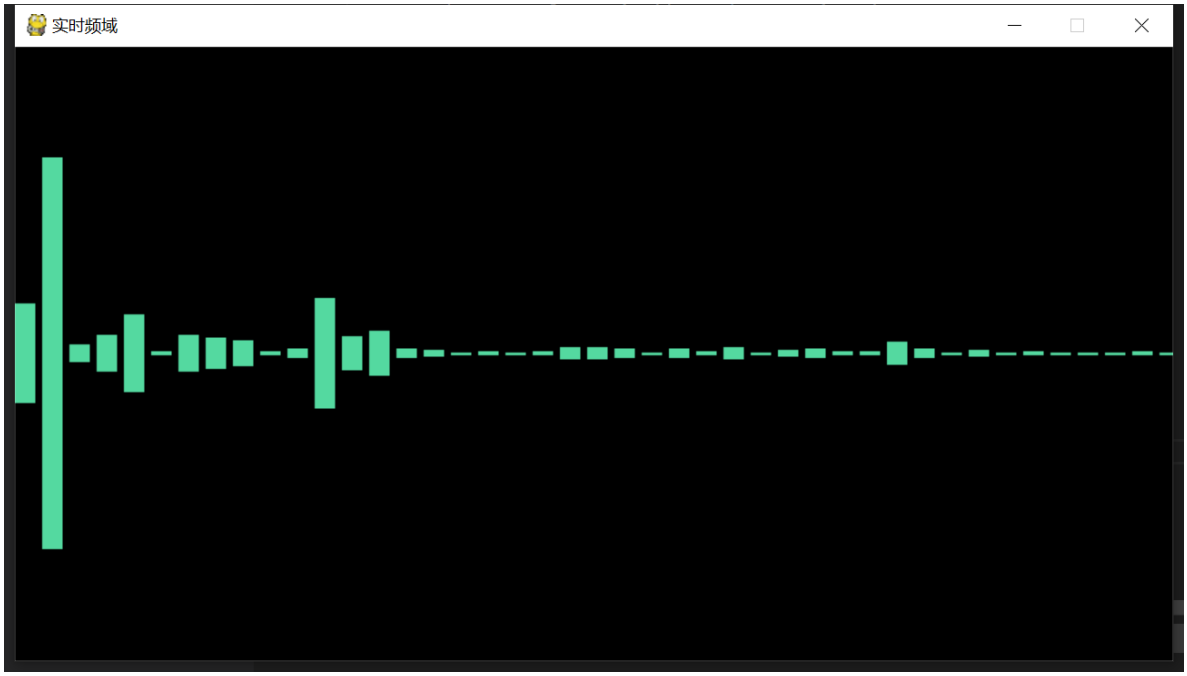
```
# 把data由字符串以十六进制的方式转变为数组
numpydata = np.fromstring(data, dtype=np.int16)
# 傅里叶变换获取实数部分
freqarray = np.real(np.fft.fft(numpydata))
```

3.利用傅里叶变换结果进行可视化

利用了三角函数和时间函数对RGB赋值以实现颜色的变化

```
# 从频域中的2048个数据中每隔count个数据中选取一条
for n in range(0, freqarray.size, count):
    height = abs(int(freqarray[n]/10000)) / 2
    R = abs(math.sin(time.time())) * 255
    G = abs(math.sin(3*time.time())) * 255
    B = abs(math.sin(2*time.time())) * 255
    pygame.draw.rect(screen, (R, G, B), Rect((20*n/count, window_height/2), (15,
-height)))  # 画矩形
    pygame.draw.rect(screen, (R, G, B), Rect((20*n/count, window_height/2), (15,
height)))
pygame.display.update()
```

结果与分析



可以看出，这段音乐大部分的频率较低，高频部分较少。每次绘出的图形就是对切片后的音乐进行傅里叶变换的结果，也就是实时频域。

参考文献

https://www.pygame.org/docs

https://blog.csdn.net/weixin_43800510/article/details/100052775