

计算机图形学期末项目

田睿

17300750084 计算机科学与技术

1 任务一：编程实现音乐节奏或旋律的可视化

参考网易云音乐的音乐特效，使用python实现音乐旋律的可视化。利用librosa包作为处理音频特征的工具。使用OpenGL (pyopengl) 作为可视化工具，以PyQt作为前端窗口呈现OpenGL的绘图效果。

1.1 旋律与节奏

在任务一中，关键在于从音乐中提取可以通过可视化（绘图）呈现的信息。我们听到的音乐可以用不同表现形式定义：如音量、音高、音色、节奏等。

旋律由音乐中的音符序列组成，而每个音符具备音高属性，因此分析音乐的旋律特征，需要掌握“音高信息”。而音高由周期性的声音波形的基频决定，进一步地，需要从音乐的频率特征中表现旋律特征。

音乐节奏则包含较为广义的定义，主要指音符与时间相关的组织、强弱规律，提取节拍特征，我们需要检测起始点，即信号的突变的开始（某一段截拍的开始）。由于在时域中，许多乐声重叠，难以寻找起始点，但转化到频域中后，处在频谱中不同位置的乐声会分离开，更容易得到节拍图（Tempogram）。

1.2 任务分析[4]

音频文件中存储了音乐旋律的采样信号。不仅如此，流媒体的文件信息往往经过编码，以我们常用的MP3文件为例，其文件的最小结构为帧，不仅包括数据帧（我们需要的声音信号），还有标签帧。获取节奏或旋律信息，我们无需关心标签帧中的附加信息。因此，任务的第一步为解码。不同媒体流的编码方式不同，为了兼容多种音频软件，同时更加便捷、有效地提取音频信息。可以使用第三方库完成解码工作，过滤音频软件存储的信息，如功能完备的ffmpeg等。结合之后提取音乐特征的需求，我选用了python中的第音频处理库：librosa。

解码后，我们可以得到用数字信号表示的音频（脉冲编码调制下的数字信号）。采样信号和音频的采样频率有关，并且是以时间为序列记录的。 $N = sr * t$, N 为得到的音频信号中的样本数量， sr 为采样频率， t 为音频的总长度。考虑到分析旋律以及节拍特征时，频域相比时域都是更好的选择，故需要通过时域转换到频域的算法，如快速傅立叶变换（Fast Fourier Transform）等将数字信号转换到频域中。

由上文的分析，针对旋律和节奏的分析。我分别选择了较为经典的特征分析算法：倒谱分析（梅尔频率倒谱系数 Mel-Frequency Cepstral Coefficients）、基于FFT的频率分析：Constant Q transform (CQT)、节拍图（Tempogram）来提取不同的音频特征，其表现的音乐特征可见图1

得到音频特征后，需要可视化特征。借鉴网易云的特征显示，我分别使用粒子系统以及条状图来表现音乐特征。粒子系统以及条状图都以环形显示。而粒子的高度（条状图的高度）则表示特征的强度。

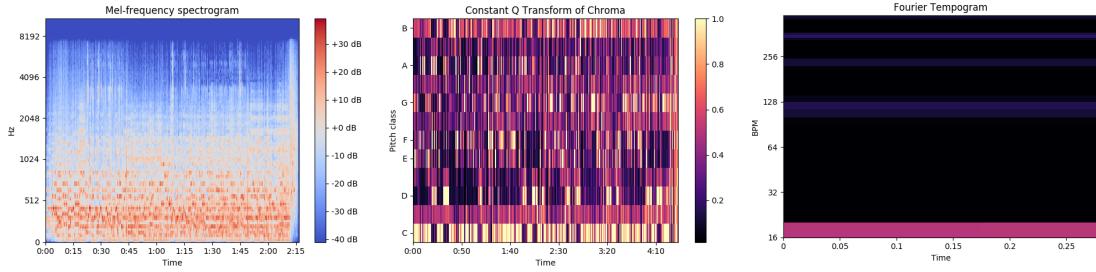


Figure 1: 从左到右依次为同一首音乐的梅尔频率倒谱 (mel-frequency spectrogram) 、半音 (chroma) CQT谱以及傅立叶节拍图 (fourier-tempogram)

1.3 主要算法

1.3.1 Mel-Frequency Cepstral Coefficients

倒频谱指信号频谱取对数的傅里叶变换后的新频谱。其中，梅尔倒频谱系数是被用来表示音频特征的重要指标之一。当数字信号转换到梅尔倒频谱上，每一个频带在梅尔刻度上均匀分布。由于梅尔倒频谱是非线性的，与之相似的是，人的听觉系统也是非线性的，因此，MFCC可以更接近人的感受来提取音高信息。

梅尔倒频谱系数通常用以下方法得到的[2]：

- 1 将音频信号分到多个帧(与librosa中的n_mels相对应)
- 2 信号输入高通滤波器，滤去低频信息，强化信号
- 3 对信号作傅立叶变换，将时域信号转换到频域
- 4 将每个帧的频谱输入到梅尔滤波器，得到梅尔刻度
- 5 对每个梅尔刻度的值取对数
- 6 对5的结果做离散傅立叶变换，转换到倒频谱域
- 7 MFCC为6中的幅度

其中，第4、5步可以表达为 (Y_m 为第4步结果， X_k 为第3步结果)：

$$Y_m = \log\left(\sum_{k=f_m-1}^{f_m+1} |X_k|^2 B_m[k]\right)$$

$$B_m[k] = \begin{cases} 0 & \text{for } k < f_{m-1} \text{ and } k > f_{m+1} \\ \frac{k-f_{m-1}}{f_m-f_{m-1}} & \text{for } f_{m-1} < k < f_m \\ \frac{f_{m+1}-k}{f_{m+1}-f_m} & \text{for } f_m < k < f_{m+1} \end{cases}$$

在实际的项目中，我使用librosa.feature.melspectrogram来实现从数字信号到梅尔频率倒谱域的转变。

1.3.2 Constant Q transform[1]

Constant Q transform也是建立在傅立叶变换的基础上的，它将傅立叶变换的频率输入转变为幅度的输出，这对于音频文件的处理显然更有效。CQT将会滤去大量高频的信息，由于人耳的听觉范围在20Hz到20kHz，输出数据中，这样的删减效果十分显著。

CQT算法可以被认为是一系列对数间隔的滤波器 f_k , 第 k 个滤波器有谱宽 δf_k , 是上一个滤波器谱宽的两倍:

$$\begin{aligned}\delta f_k &= 2^{1/n} \cdot \delta f_{k-1} \\ &= (2^{1/n})^k \cdot \delta f_{min}\end{aligned}$$

- 1 输入数据作短时距傅立叶变换 (short-time Fourier transform)
- 1 基于采样频率为 rs 的数据, 设置 k 个桶, 第 k 个桶的窗口长度为: $D[k] = \frac{rs}{\delta f_k} = \frac{rs}{f_k} Q$ 。且有 $Q = \frac{f_k}{\delta f_k}$ 。并采用归一化处理。
- 2 对于每个窗口, 设置窗口函数。函数与窗口的序号有关, 也与窗口长度有关, 如 Hamming 窗口: $W[k, d] = \alpha - (1 - \alpha) \cos \frac{2\pi d}{D[k]-1}$, $\alpha = 25/36$, $0 \leq d \leq D[k] - 1$, 电子信号的频率从 $\frac{2\pi k}{D}$ 变为 $\frac{2\pi Q d}{D}$
- 3 基于短时距傅立叶变换 (short-time Fourier transform), 最终得到的计算CQT的公式是: $X[k] = \frac{1}{D[k]} \sum_{n=0}^{D[k]-1} W[k, d] x[d] e^{-j2\pi Q d}$

在实际的项目中, 我通过librosa.feature.chroma_cqt来实现Constant Q transform。

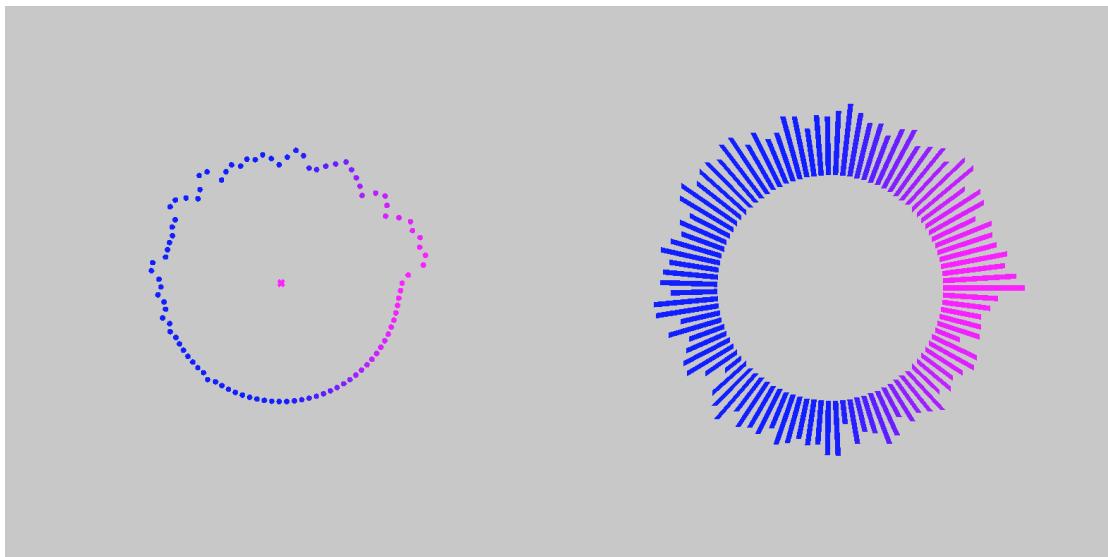


Figure 2: 实际呈现出的可视化效果, 左侧为粒子样式, 右侧为柱形图样式

2 任务二：编程画一个真实感静态景物[3]

任务二中我使用OpenGL以及glfw、glew开源程序库, 实现了两个模型的绘制。绘制的过程中主要使用OpenGL3中的着色器, 实现blinn-phong算法、shadow mapping算法、以及波松采样。

2.1 任务分析

由于要实现真实感静态景物的绘制, 所以要尽量在物理特性上与真实物体相似。为了更接近于真实世界的物体, 我选用了开源的3d模型, 分别是一个乐高积木人以及一个松鼠的卡通3d像。我准备表现出桌面上放置了两个玩具的场景。而两个3d模型的进一步仿真可以用贴图技术 (texture) 来实现。Opengl对次的支持是很好的。除去在几何上表现真实感, 营造真实感图形的另一个重点在于建立正确的光照模型。在这样的前提

下，我使用点光源，以blinn-phong光照模拟场景的光照，并对魔性的贴图计算漫射以及镜面反射分量以增加光线追踪的真实感。同时使用阴影贴图技术渲染阴影，使用Poisson采样技术随机采样阴影边界，柔和阴影。增加场景真实感。

2.2 基本算法

2.2.1 纹理贴图

纹理贴图的过程为将二维坐标映射到三维的模型表面。为了能够实现这样的映射，需要制定三维表面的每个顶点分别映射到纹理的哪一个目标。在我的视线中，我使用了OpenGL中集成的纹理贴图方法，且使用的3d模型中保存有纹理坐标信息，指明3d模型的某一片表面应该从纹理图片的什么部分采样。在OpenGL的视线中，纹理贴图在采集片段颜色后，会做相应的片段插值。OpenGL中，纹理坐标不依赖于分辨率，但是纹理的分辨率可能很小，与之对应，3d模型可能很大。所以需要通过纹理过滤的方式来帮助平滑纹理，现在假设纹理坐标落于纹理图像的一个点上，点的周边有四个像素（点落在其中一个像素的范围内）：

- 临近过滤(nearest filtering) 纹理坐标选择距离像素的中心点与纹理坐标最接近的点。
- 线性过滤(linear filtering) 纹理坐标对四个颜色进行线性处理：假设四个像素点的中心坐标与纹理坐标的距离分别为： d_1, d_2, d_3, d_4 ，它们的颜色分别为 c_1, c_2, c_3, c_4 ，则可得到最终纹理坐标的渲染颜色 c : $c = \sum_{j=1}^4 c_j * \frac{d_j}{\sum_{i=1}^4 d_i}$

2.2.2 坐标映射

在真实感绘图中有一个重要的课题：3维空间坐标的表示。在OpenGL中所有的可见位置坐标都应该在 $(-1.0 \square 1.0)$ （标准化设备坐标）之间，以便于传入光栅器，转化为屏幕上的坐标。不仅如此，众所周知，人眼看到的世界是有透视效果的，为了真实还原这一呈现，需要将实际的模型3d坐标投射到透视的坐标系中。在OpenGL中，有5个坐标系被认为很重要，这五个坐标系从上到下表示了绘制图形的真实处理过程，每一个空间之间的转换都由矩阵的乘法完成：

- 1 局部空间（物体空间）：即我们所使用的3d模型所使用的坐标系
- 2 世界空间：即我们绘制的“世界”的坐标
- 3 观察空间：即作为观察者的视线下看到的空间，直观想象，视线的方向是多变的，不同视角所看到的物体显然是不同的。
- 4 剪裁空间：即将不可见点筛去的空间，这一空间中将包含重要的透视映射
- 5 屏幕坐标：OpenGL传递给光栅器的坐标。这一过程由OpenGL内置的glViewport函数完成。

透视映射 透视映射的直观表现是远大近小，或者说，视线并非平行，而会在远处相交。在透视映射中，我们可以将视野 (x, y, z) 想象为一个被截断的金字塔，截断多少将由 w 决定。透视映射矩阵所做的就是要将这样的视野转化到一个立方体的坐标中，将 $x : [l, r] \rightarrow [-1, 1], y : [b, t] \rightarrow [-1, 1], z : [-n, -f] \rightarrow [-1, 1]$ 。其变换矩阵 $p = (x, y, z, w)^T$ 为：

$$p = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

2.2.3 blinn-phong

blinn-phong光照模型的基础是冯氏光照模型（phong lighting model）。而冯氏光照模型主要由三个元素组成：

- 1 环境光照（ambient lighting）：远处总会有光照向物体，因此环境光使得极暗情况下物体仍有颜色
- 2 漫反射光照（diffuse lighting）面向光源的一面比其他面更亮：在散射的计算总，需要考虑表面的法向量 n ，光源位置向量 L_p ，片段位置 F_p 。光的方向向量是光的位置向量和片段的位置向量的向量差。这可以通过两个向量相减得到，并且需要对这个向量差向量 $\delta = L_p - F_p$ 做归一化处理。光的散射分量可以由 $diffuse = (n \cdot \frac{L_p - F_p}{\|L_p - F_p\|}) * c$ 得到。
- 3 镜面反射（specular lighting）模拟光滑平面上的光源的镜像亮点。镜面反射的颜色更倾向于光的颜色：

phong反射具有一定的局限性。当视线和反射角之间夹角大于90度。镜面反射成分会被消除。blinn-phong为了解决这一问题，使用半程向量（位于视线方向和光线方向之间的单位向量）。当半程向量与物体表面的法线向量越接近，镜面反射成分就越大。半程向量可以由光的方向 L 和视线向量 V 简单相加后再进行归一化得到。由此，镜面反射可以由表面法线和半程向量点乘得到。

由于绘制的场景中所有的颜色都来自纹理，因此ambient、diffuse、以及specular的光照模型需要映射到纹理贴图中。

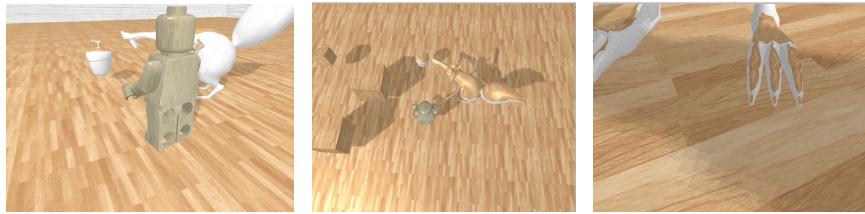


Figure 3: 图1（左）可见lego模型中的镜面反射效果，图2（中）可见透明物体的折射效果，图3（右）可见阴影采样的效果

2.2.4 shadow mapping

阴影映射的思路是：以光源为视角，所有能看到的物体表面都是明亮的。看不见的则在阴影之中。在opengl中，渲染阴影使用了深度贴图，在光的透视视图中渲染场景中的深度纹理。用深度纹理计算阴影。将计算好的深度值（在阴影中为1，不在为0）传递给模型的着色器，调整blinn-phong光照模型。将所有处于阴影中的物体的漫反射光照以及镜面反射都乘以阴影值。

在着色器中，使用函数计算阴影。对于投影坐标超过光的远平面（透视模型的尽头），默认阴影为0。阴影渲染中，由于与光源距离，可能会出现阴影偏移（shadow bias）。并且由于深度贴图的解析值受限，阴影边缘并不平滑，可能出现锯齿状。针对阴影的平滑问题，有几种算法作为解决方案，如pcf：多次采样，调整每次采样的纹理坐标，最后对采样结果取平均。而泊松采样（poisson sampling）则对pcf做进一步改进，设定一个常数数组，常数在[0,1]范围内，对于pcf中的多次取样，每次取样从数组中随机抽取常数作为扰动的偏移量。最后依旧是多次采样取平均。阴影的边界可以有效地变柔和。

在真实感图形中的实现中，我使用的算法还包括：透视视角转换，透明物体的折射算法等。

3 任务三：创作一个Flash动画

任务三中需要实现一个动画。老师给予的反馈是：可以不使用flash软件，但需要展现出动画的形式。于是我选择使用css的svg特性来绘制动画。在绘制动画中，实现的重点可以分为两部分：

- 1) 图形的绘制：svg 中的一个重要方法是"path "，而path中带有属性 " d "，支持多种绘制方式。其中重要的方式包括C: 三次贝塞尔曲线以及Q: 二次贝塞尔曲线的绘制。
- 2) 动态的实现：css中支持的重要特性“transform”帮助我们使得图像动起来，rotate、scale、translate分别对应旋转、放大缩小以及平移。结合css中animate-duration使用，使用animationTime以一定周期，规律地变换图形。

References

- [1] Judith C. Brown. Calculation of a Constant Q Spectral Transform. 1991.
- [2] Qi Tian Changsheng Xu. HMM-based audio keyword generation. 2004.
- [3] Joey DeVries. <http://learnopengl.com/>.
- [4] 李伟，李子晋，高永伟. 理解数字音乐-音乐信息检索技术综述.