数字视音频复习

音乐

音乐的基本要素

音高（频率）、音强（振幅）、音长（时值）、音色（发声体谐波特性）

计算机表示方法：

音乐的制作流程

作词作曲，编曲，录音，混音

音乐合成技术

1. **软件**

   overture

   SONAR

   Adobe Audition

2. **语言编程**

    1. 基于Nyquist的音乐合成（lisp，基本声音的合成），OpenAL（三维音效），Matlab，Flash Action Script

3. 更倾向于音乐艺术与软件工程的结合
    - 创意
    - 协作
    - 软件/语言的使用

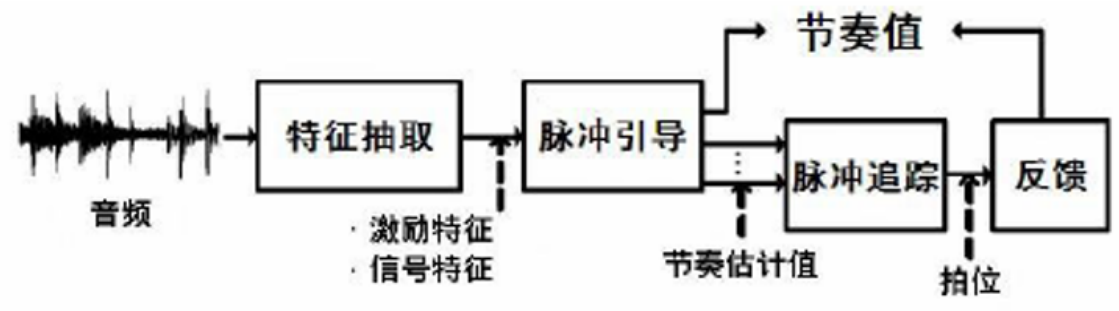
音乐分析

表现形式

- 节奏：组织起来的音的长短关系
- 旋律：长短，高低，强弱不同的一连串乐音有组织的进行
- 和声：和弦（三个或以上乐音组合） + 和声进行（和弦的横向组织）

节奏识别

框架：



特征提取方式：时域分析、频域分析

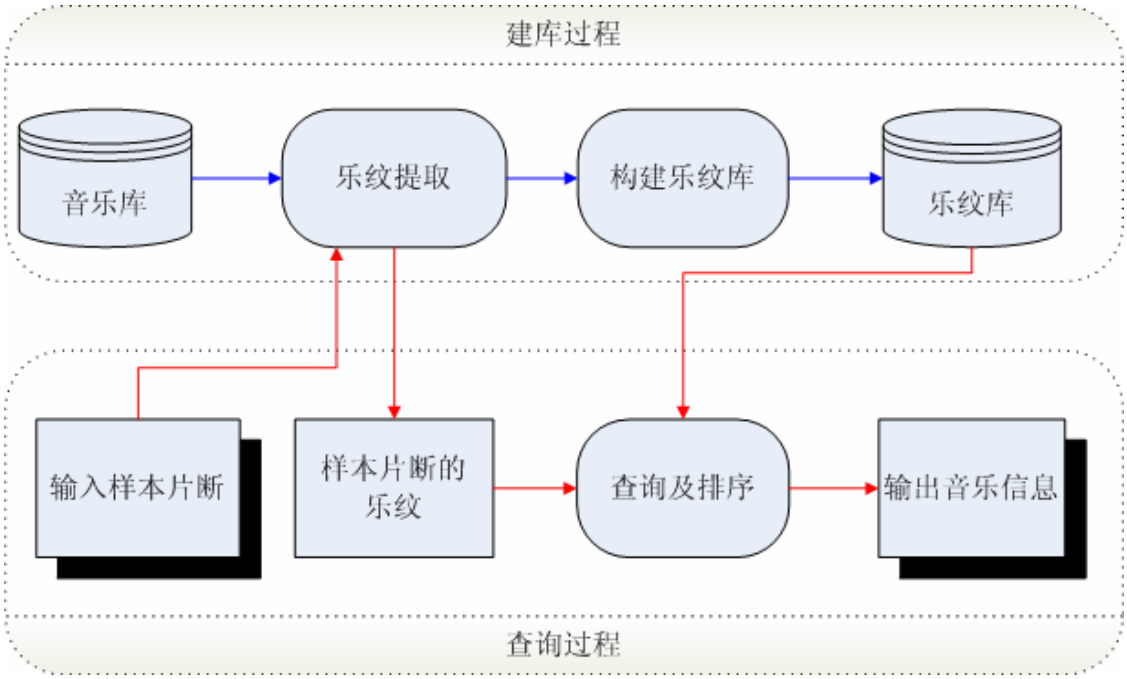
音准评分

1. 旋律评分模块：求取演唱者演唱旋律，并与歌曲原旋律对比，匹配越好给分越高
2. 抢拍与慢拍分析模块：分析演唱者抢拍与慢拍情况，并酌情给予减分
3. 节奏分析模块：分析演唱者的节奏感是否与歌曲的节奏一致，节奏感越好给分越多
4. 演唱情绪分析模块：分析演唱者演唱的情绪，如果演唱者演唱的情绪与歌曲的意境相符，会有相应的加分
5. 声音圆润饱满度分析模块：分析演唱者演唱的声音是否圆润，是否饱满，越圆润越饱满给分越多
6. 语音识别模块：分析演唱者演唱的歌词是否与歌曲的歌词相符，错误率越少给分赵多

音乐检索

乐纹检索

1. 框架：



频域分析→通过特征点对索引技术构建乐纹库→三重链表查询

2. 乐纹概念与特性

概念：可以代表一段音乐重要声学特征的基于内容的紧致数字签名

特性：鲁棒性、区分性

哼唱检索

1. 框架：

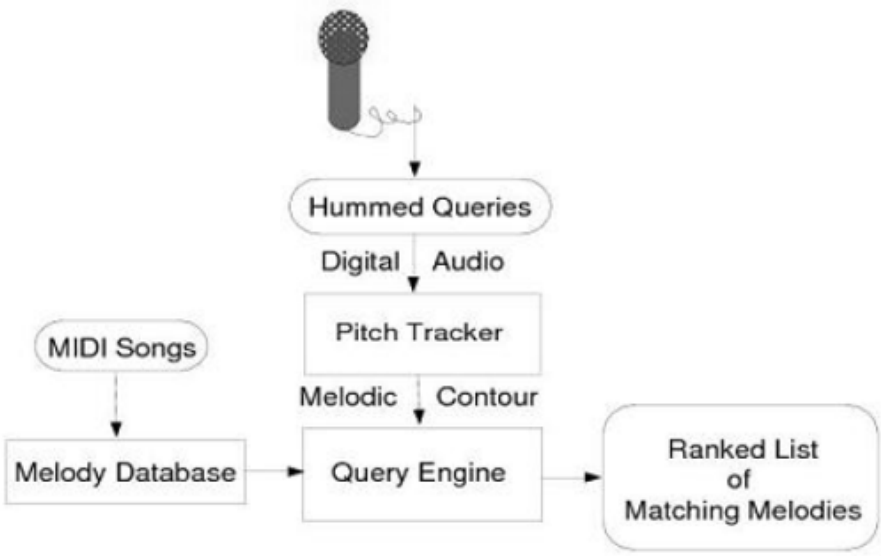


图 1.1: Ghias的哼唱检索系统框架 [2]

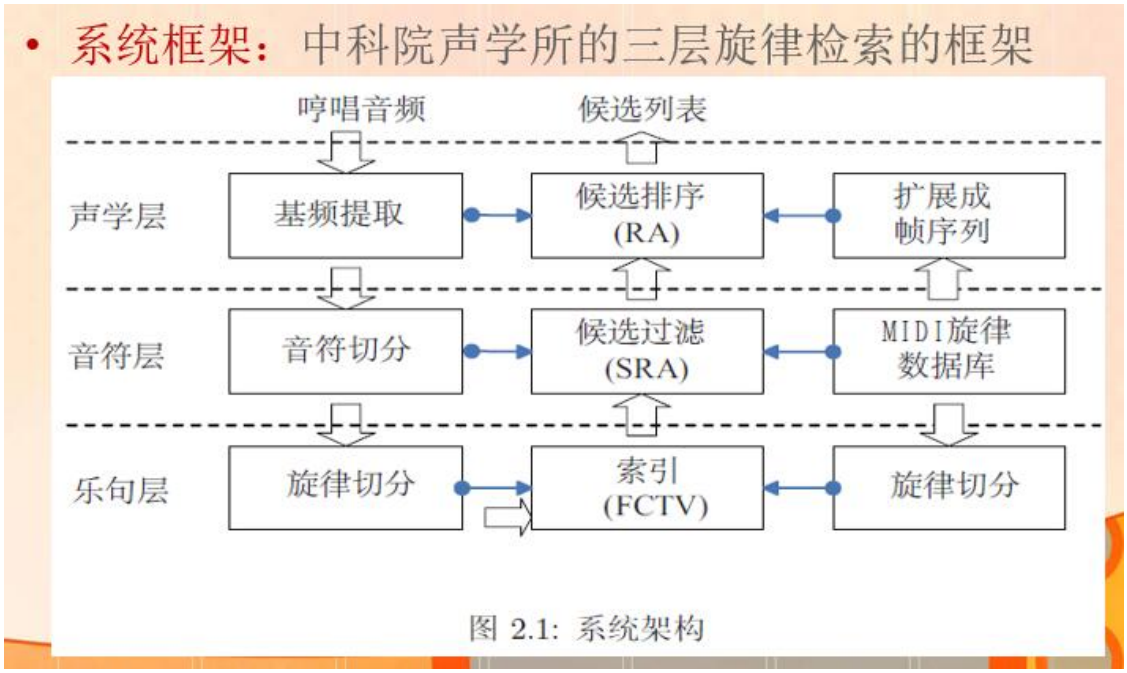


图 2.1: 系统架构

2. 旋律表示：三层表示
    - 在最低的声学层上，哼唱的旋律经过基频提取，被表示为基于帧的音高序列
    - 在较高的符号层（或音符层）上，系统综合基频曲线、谐波和能量等信息，将基频序列切分成格式化的音符序列；
    - 最后，系统将在乐句的层次上寻找旋律中的轮廓点，并试图确定数据库中的旋律乐句边界
3. 旋律特征提取

基频提取、音符切分、轮廓点提取等各级旋律提取结果

4. 旋律查询

   基于轮廓因子索引的快速检索法 midomi公司

视频

视频数据压缩

无损压缩与有损压缩

- 无损压缩：压缩的数据和原始数据完全一样。
- 有损压缩：压缩的数据和原始数据不相同，但非常相近。

H.261

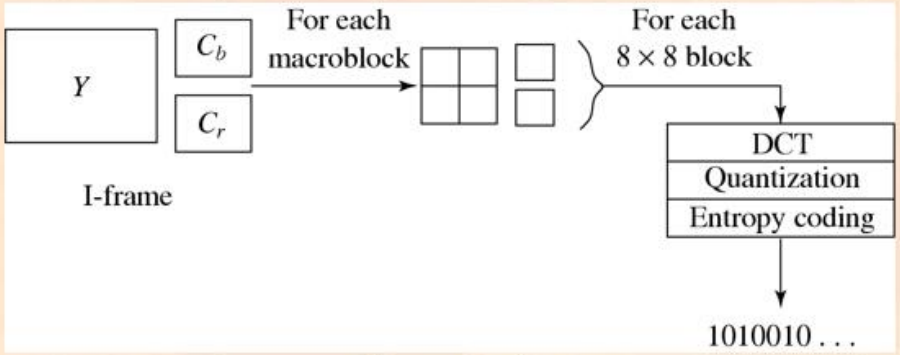
- Two types of image frames are defined: Intra-frames (**I-frames**) and Inter-frames (**P-frames**):

  - **I-frames** are treated as independent images. Transform coding method similar to JPEG is applied within each I-frame, hence “Intra”.

  -**P-frames** are not independent: coded by a forward predictive coding method (prediction from a previous P-frame is allowed — not just from a previous I-frame).

  I P P P I P P P I...

- I帧编码：



I-frame Coding.

Macroblocks are of size 16 x 16 pixels for the Y frame, and 8 x 8 for Cb and Cr frames, since 4:2:0 chroma subsampling is employed. A macroblock consists of four Y, one Cb, and one Cr 8 x 8 blocks.

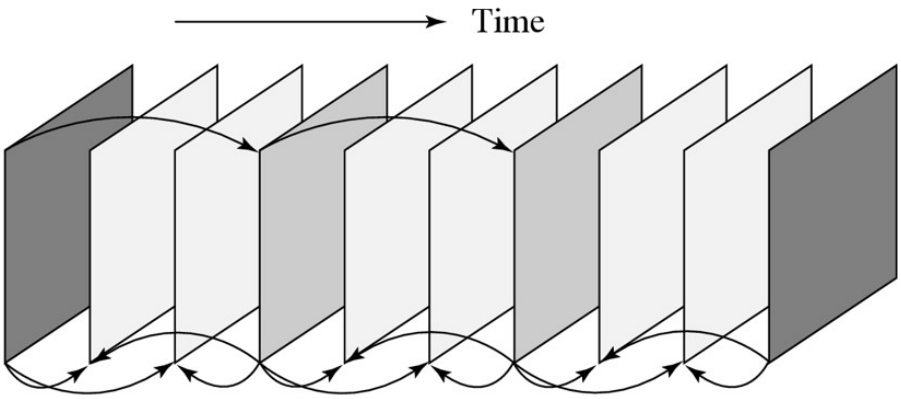
For each 8 x 8 block a DCT transform is applied, the DCT coefficients then go through quantization zigzag scan and entropy coding.

- 量化方式：

$$QDCT = round(\frac{DCT}{\text{step size}}) = round(\frac{DCT}{8})$$

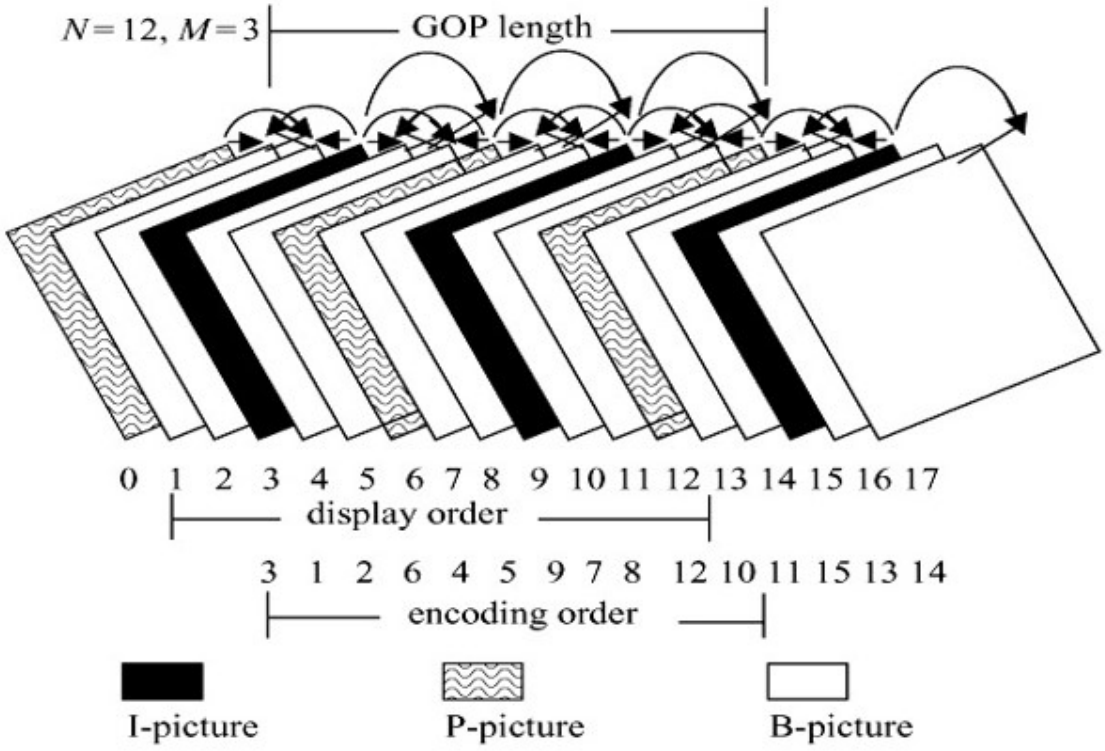
MPEG-1

引入B帧的概念，双向动作补偿：前后取平均



Display order I B B P B B P B B I

Coding and transmission order I P B B P B B I B B



For DCT coefficients in Intra mode:

$$QDCT[i,j] = round\left(\frac{8\times DCT[i,j]}{step\_size[i,j]}\right) = round\left(\frac{8\times DCT[i,j]}{Q_1[i,j]*scale}\right)$$

For DCT coefficients in Inter mode:

$$QDCT[i,j] = \left\lfloor\frac{8\times DCT[i,j]}{step\_size[i,j]}\right\rfloor = \left\lfloor\frac{8\times DCT[i,j]}{Q_2[i,j]*scale}\right\rfloor$$

I帧(Intra-Frame)是帧内压缩，不使用运动补偿，提供中等的压缩比。由于I帧不依赖于其他帧，所以是随机存取的入点，同时是解码中的基准帧。

P帧(Predicated-Frame)根据前面的I帧或P帧进行预测，使用运动补偿算法进行压缩，因而压缩比要比I帧高，数据量平均达到I帧的 1/3左右。P帧是对前后的B帧和后继的P帧进行解码的基准帧。P帧本身是有误差的，如果P帧的前一个基准帧也是P帧，就会造成误差传播。

B帧(Bidirectinal-Frame)是基于内插重建的帧，它基于前后的两个I、P帧或P、P帧，它使用双向预测，数据量平均可以达到I帧的 1/9左右。B帧本身不作为基准，因此可以在提供更高的压缩比的情况下不传播误差。 需要指出的是，尽管我们使用帧（Frame）这个词，但是MPEG2本身没有规定进行数字图像压缩时必须使用帧作为单位，对于隔行的视频图像，可以使用场 （Field）作为单位。

视频结构化与非线性编辑部分

镜头检测

镜头是视频流数据的最小物理数据单元，所谓镜头检测就是给定有n个镜头的视频V，找到每个镜头的开始和结尾部分。也被称作边界检测（boundary detection）或转换检测（transition detection）。

镜头边缘检测算法的实质及核心问题

实质：找到一种或几种良好的视频图像**特征**，通过判断相邻图像帧之间的特征是否发生**剧烈变化**，来完成视频镜头边缘检测任务。

核心问题：如何选择特征，如何定义相似度函数

//或者是关键问题：(1) 自适应阈值 (2)渐变镜头数学模型

视频结构化

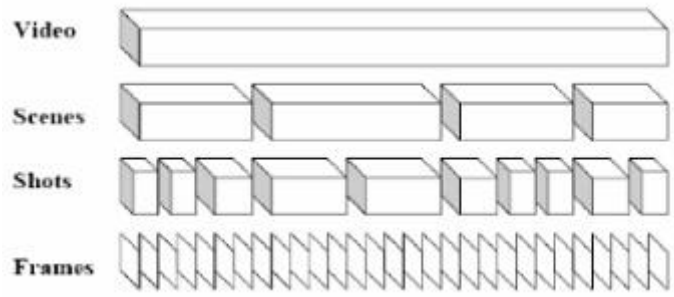
帧（frame）：帧是视频流中的基本组成单元

镜头（shot）：摄像机拍下的不间断帧序列，是视频数据流进一步结构化的基础结构层。

关键帧（key frame）：关键帧是可以用来代表镜头内容的图像

场景（scene）：语义上相关和时间上相邻的若干组镜头组成了一个场景，场景是视频所蕴涵的高层抽象概念和语义的表达

组（group）：组是介于物理镜头和语义场景之间的结构。



视频结构化分析包含哪些基本步骤、内容（又：视频目录生成构造的主要步骤

- 镜头边缘检测
- 关键帧提取
- 时空特征提取
- 时间可适性成组
- 场景结构构造

镜头时间特征和空间特征的区别

镜头时间特征：包含运动信息，即镜头中前后两帧的差异累积

是基于镜头中所有视频帧得到的

$$Act_i = \frac{1}{N_i - 1} \sum_{k=1}^{N_i - 1} Diff_{k,k-1}$$

$$Diff_{k,k-1} = Dist(Hist(k), Hist(k-1))$$

在上面公式中， Act_i 和 N_i 分别表示镜头 i 的运动信息和包含的帧数； $Diff_{k,k-1}$ 分别表示表示帧 k 和 $k-1$ 的颜色直方图差； $Hist(k)$ 和 $Hist(k-1)$ 分别表示帧 k 和 $k-1$ 的颜色直方图； $Dist()$ 是直方图的距离，这里用的是直方图求交。

镜头空间特征：

基于镜头中的关键帧得到的

镜头的空间信息特征从关键帧得到，表示为：

$$Hist(b_i) \text{ 和 } Hist(e_i)$$

其中 b_i 和 e_i 分别表示镜头 i 的首帧和尾帧， $Hist(b_i)$ 表示首帧直方图值， $Hist(e_i)$ 表示尾帧直方图值。

镜头边缘检测算法

绝对帧间差法

- 判断相邻图像帧之间特征的绝对差是否大
- 具体实现时，判断两个相邻帧差别的方法可以是：计算相邻两个图像帧中所有像素的色彩亮度之和，两帧的差别就定义为各自对应像素的亮度和之差

图像像素差法

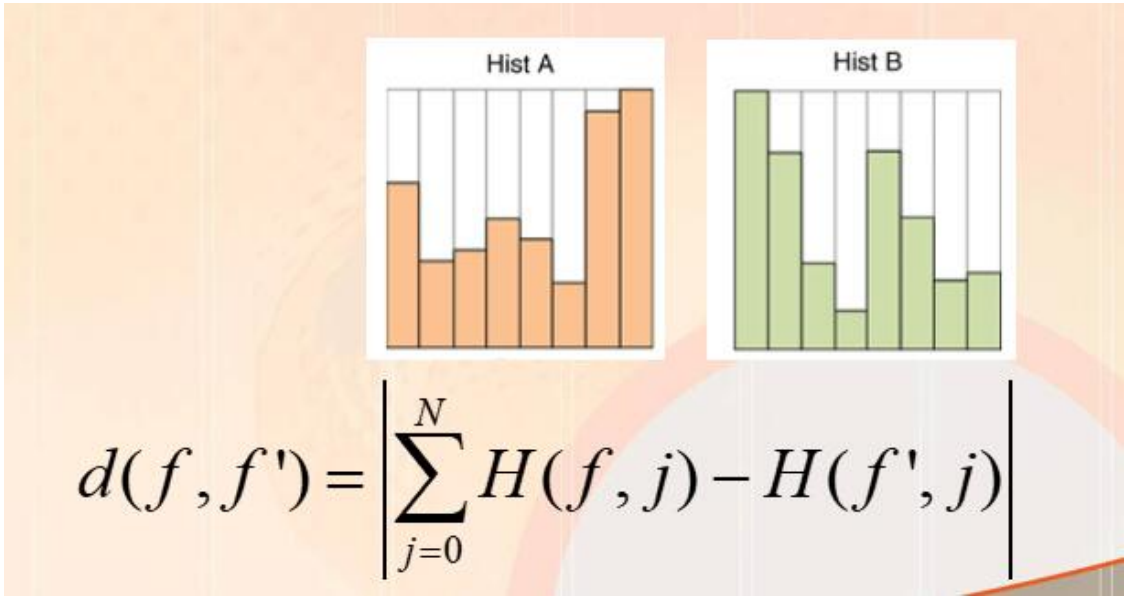
- 首先统计两幅图像对应像素变化超过阈值的像素点个数。
- 然后，将变化的像素点个数与第二个预定的阈值比较，如超过范围，则认为这两帧之间发生较大变化，判断其为镜头边界
- 但该法对镜头移动十分敏感，对噪声的容错性较差。

图像数值差法

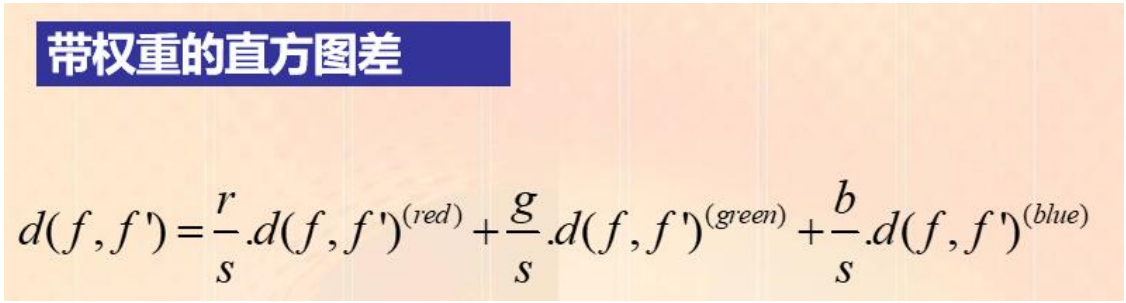
- 将图像分成若干个子块区域，在这些区域中分别比较对应像素数值上的差别。

颜色直方图法

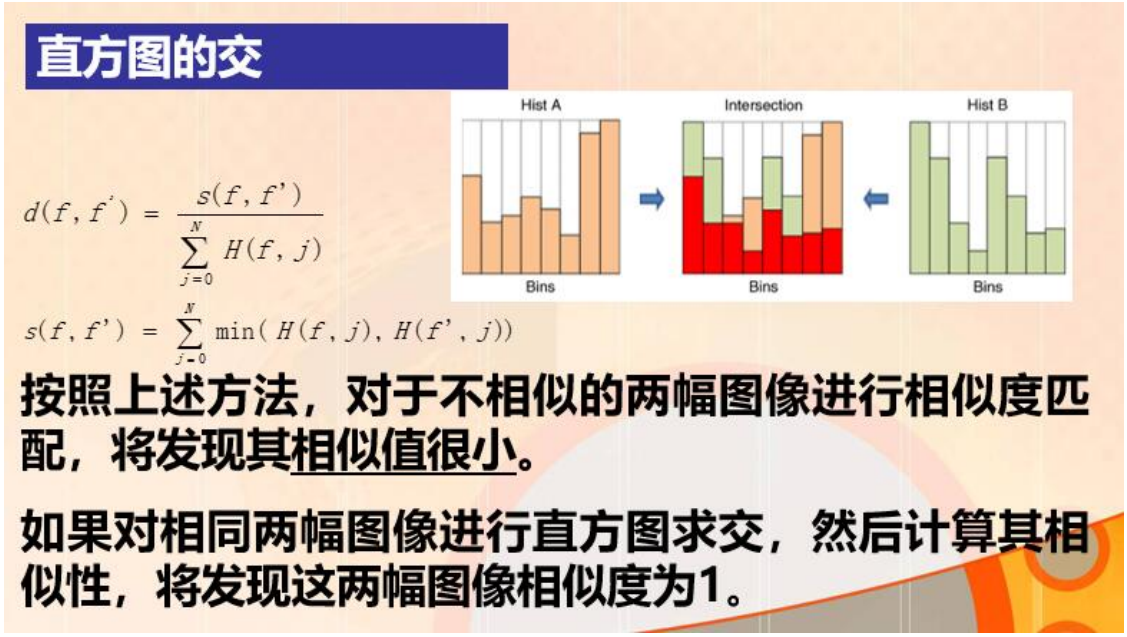
- 通过直方图差度量



- 通过带权重的直方图差



- 直方图的交



压缩域差法

- 在这种方法中，不对图像解压，而是直接用JPEG压缩图像帧的DCT系数作为帧相似度衡量的标准。
- 一般来说，在对视频图像进行处理时候，要对图像解压，因为传统的颜色等特征是定义在像素基础上的，也就是非压缩域基础上的。
- 这样，基于压缩域的视频镜头检测可以省去解压步骤，直接从原始视频数据流中提取特征，从而加快检测速度。
- 由于使用压缩域对视频进行编码，每个压缩域系数保留了原始图像帧中或图像帧间最重要特性，所以压缩域系数可以有效分析视频数据。

矩不变量法

- 图像矩不变量具有比例、旋转和过渡不变性的特点，是用来表示图像帧的好方法，所以可以用来进行镜头边缘检测。

$$m_{pq}=\sum_x\sum_y x^p\cdot y^q\cdot f(x,y)$$

图象矩

$$n_{pq}=\frac{1}{m_{00}^{r}}\cdot\sum_x\sum_y(x-\bar{x})^p\cdot(y-\bar{y})^q\cdot f(x,y)$$

矩不变量

$$r=1+(p+q)/2$$

$$\bar{x}=m_{10}/m_{00}$$

$$\bar{y}=m_{01}/m_{00}$$

$$m_{pq}=\sum_x\sum_y x^p\cdot y^q\cdot f(x,y)$$

$$n_{pq}=\frac{1}{m_{00}^{r}}\cdot\sum_x\sum_y(x-\bar{x})^p\cdot(y-\bar{y})^q\cdot f(x,y)$$

$$r=1+(p+q)/2$$

$$\bar{x}=m_{10}/m_{00}$$

$$\bar{y}=m_{01}/m_{00}$$

对视频镜头边缘检测进行评估时，如下三个矩不变量被使用：

$$\Phi_1=n_{20}+n_{02}$$

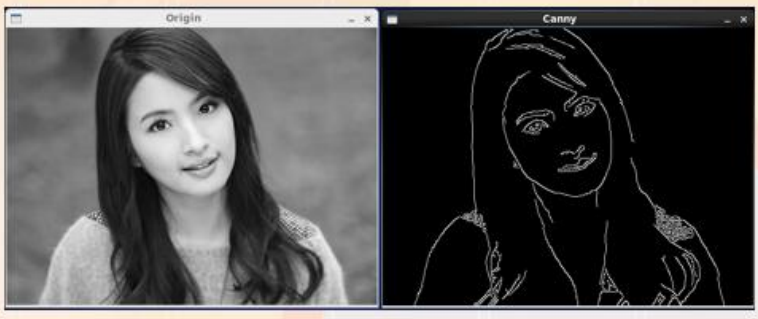
$$\Phi_2=(n_{20}-n_{02})^2+4n_{11}^2$$

$$\Phi_3=(n_{30}-3n_{12})^2+(3n_{21}-n_{03})^2 \quad (8.11)$$

从相邻图像帧 f 和 f' 中提取了矩不变特征，就可以计算这些矩不变特征的欧拉距离 $d(f,f')$：$d(f,f')=\left|\overrightarrow{\sigma_f}-\overrightarrow{\sigma_{f'}}\right|^2$，其中 $\vec{\sigma}=\{\Phi_1,\Phi_2,\Phi_3\}$。如果 $d(f,f')$ 超过一定阈值，则认为 f 和 f' 间出现了镜头转换。

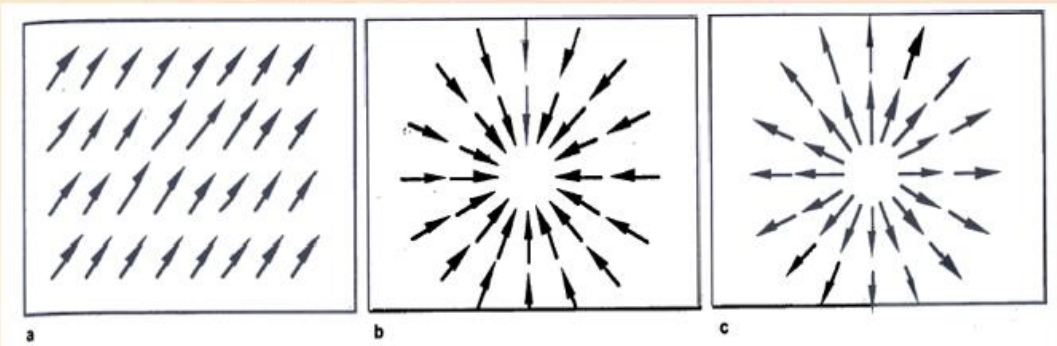
边界跟踪法

- 在镜头的转换中，在距离原来边缘很远的位置会出现新的边缘，而原来的边缘会逐渐消失。因此，镜头转换的判断可以看作是两个图像帧中边缘的比较。
- 计算相邻两帧间进入或者离开图像的边所占百分比，百分比最大的是镜头的切分点。



运动矢量法

- 在这种方法中，计算相邻视频帧之间的运动位移。
- 这个运动位移可以是光流场（optical flow）计算，也可以子块匹配（block matching）等方法。



Motion vector patterns resulting from camera panning and zooming. **a** ↙ Camera panning direction. **b** Camera zoom-out. **c** Camera zoom-in

渐变镜头的数学模型

Dissolve的数学模型

f(x,y)场景A

g(x,y) 场景B

L1：场景A持续时间

L2：场景B持续时间

F：场景A,B Dissolve持续时间

$$S_n(x,y)=(1-\frac{n-L_1}{F})f_n(x,y)+\frac{n-L_1}{F}g_n(x,y)$$

均值：$m_{s,n}=[m_f-\frac{L_1}{F}(m_g-m_f)]-\frac{n}{F}(m_f-m_g)$

方差：

$$\sigma_{s,n}^2=\sigma_f^2 \quad (0\le n\le L_1)$$

$$\sigma_{s,n}^2=[\frac{\sigma_f^2+\sigma_g^2}{F^2}]n^2-[\frac{2L_1(\sigma_f^2+\sigma_g^2)}{F^2}-\frac{2\sigma_f^2}{F}]n+[\sigma_f^2+\frac{L_1^2(\sigma_f^2+\sigma_g^2)}{F^2}+\frac{2L_1\sigma_f^2}{F}] \quad (L_1\le n\le L_1+F)$$

$$\sigma_{s,n}^2=\sigma_g^2 \quad (L_1+F\le n\le L_2)$$

关键帧提取算法

基于镜头边界法

将切分得到镜头中的第一幅图像和最后一幅图像作为镜头关键帧

基于特征转变法

- 在基于视频图像特征提取关键帧方法中，镜头当前帧与最后一个判断为关键帧的图像比较，如有较多特征发生改变，则当前帧为新的一个关键帧。

基于运动分析法

- 在这种方法中，将相机运动造成的图像变化分成两类

1. 由相机焦距变化造成: 选择首、尾两帧为关键帧；
2. 由相机角度变化造成: 如当前帧与上一关键帧重叠小于30%，则选其为关键帧；

基于聚类的关键帧提取

常用K-means。求帧与质心距离，距离大形成新的聚类，否则加入原有聚类；每次计算后都更新质心；非监督过程。

时间可适性组构造

- 实际上就是**镜头成组过程**，是视频目录结构生成过程中的第四步
- 在构造场景结构之前，先用得到的镜头信息构建一个结构，这个结构称为“组（group）”，每个组里包含相似的镜头。

镜头相似

1. **视觉相似性**：相似镜头在视觉上是相似的，也就是具有相似的空间特征（颜色直方图等）。
2. **时间局部性**：相似镜头在时间上会尽可能接近。视觉上相似的镜头如果在时间上相差很远，就不太可能属于同一场景，因此也不属于同一组。

时间可适性成组法

1. 先计算镜头间的颜色相似度（空间特征）

(a) 对镜头 $shot_i$ 和 $shot_j$ 中的四个关键帧计算四种颜色相似度，分别是 $FrameColorSim_{b_i,e_j}$ 、 $FrameColorSim_{e_i,e_j}$ 、 $FrameColorSim_{b_i,b_j}$ 和 $FrameColorSim_{e_i,b_j}$。其中 $FrameColorSim_{x,y}$ 定义为：

$$FrameColorSim_{x,y} = 1 - Diff_{x,y}$$

x 和 y 是任意关键帧，但是 x 出现在 y 前面。

(b) 定义随帧特征差递减的函数：时间引力(temporal attraction) $Attr$：

$$Attr_{bj,ei} = \max(0,1 - \frac{b_j - e_i}{baseLength})$$

$$Attr_{ej,ei} = \max(0,1 - \frac{e_j - e_i}{baseLength})$$

$$Attr_{bj,bi} = \max(0,1 - \frac{b_j - b_i}{baseLength})$$

$$Attr_{ej,bi} = \max(0,1 - \frac{e_j - b_i}{baseLength})$$

$$baseLength = Multiple * avgShotLength$$

其中 $avgShotLength$ 是整个视频的镜头平均长度； $Multiple$ 是控制时间引力(temporal attraction)以多快的速度减少到的零的速度。实验证明 $Multiple$ 为 10 的时候可以得到很好的效果。时间引力(temporal attraction)的定义说明帧的距离越远，时间引力(temporal attraction)的值越小。如果帧之间的特征差大于平均镜头长度的 $Multiple$ 倍，时间引力降为 0。

(c) 把原始的相似度按照下面方式转换成时间可适性的相似度，使之既包含视觉相似度，又包括时间局部性。

$$FrameColorSim'_{b_1,e_i} = Attr_{b_1,e_i} * FrameColorSim_{b_1,e_i}$$

$$FrameColorSim'_{e_1,e_i} = Attr_{e_1,e_i} * FrameColorSim_{e_1,e_i}$$

$$FrameColorSim'_{b_1,b_i} = Attr_{b_1,b_i} * FrameColorSim_{b_1,b_i}$$

$$FrameColorSim'_{e_1,b_i} = Attr_{e_1,b_i} * FrameColorSim_{e_1,b_i}$$

(d) 镜头 $shot_i$ 和 $shot_j$ 之间的颜色相似度 $ShotColorSim$ 定义成上面四个相似度的最大值，即

$$ShotColorSim_{i,j} = \max(FrameColorSim'_{b_1,e_i}, FrameColorSim'_{e_1,e_i}, FrameColorSim'_{b_1,b_i}, FrameColorSim'_{e_1,b_i})$$

2. 再计算镜头间的运动相似度（时间特征）

$$ShotActSim_{i,j} = Attr_{center} * | Act_i - Act_j |$$

$$Attr_{center} = \max\left(0, 1 - \frac{(b_j + e_j)/2 - (b_i + e_i)/2}{baseLength}\right)$$

$Attr_{center}$ 是镜头 $shot_i$ 和 $shot_j$ 中心帧的时间引力。

3. 最后计算总相似度

$$ShotSim_{i,j} = W_c * ShotColorSim_{i,j} + W_A * ShotActSim_{i,j}$$

其中 W_c 和 W_A 分别是颜色和运动分量的权重。

视频场景构造（镜头成组过程）

- 是视频目录生成中的第五步
- 不仅计算当前镜头和组的相似度，而且计算当前镜头和包含所有组的场景的相似度。因为不相似的组在语义上相关也可以组合到同一个场景中，例如两个人互相交谈的过程。

算法描述

输入： 视频镜头序列，$S' = Shot_0, \ldots, Shot_i$

输出：含有场景、组和镜头的视频结构

过程：

1. 初始化：组集合 $Group = \{shot_0\}$、场景集合 $Scene = \{Group_1\}$、$Groupnum = 1$ 和 $Scenenum = 1$;
2. 如果 S 为空，算法结束，退出程序；否则，从 S 中取下一个镜头，记这个镜头为 $shot_i$；
3. 测试是否 $shot_i$ 能组合到现存组中去：

(a) 计算当前镜头 $shot_i$ 与现存每个组之间的相似度: Call findGroupSim( )。

(b) 计算 $shot_i$ 与现存组之间的最大相似度：

$$\max GroupSim_i = \max_g GroupSim_{i,g}$$

$$1 \le g \le Groupnum$$

其中 $GroupSim_{i,g}$ 是镜头 $shot_i$ 和组 $Group_g$ 之间的相似度，令和 $shot_i$ 具有最大相似度的组为 g_{max}。

(c) 测试 $shot_i$ 是否能组合到现存组。

如果 $\max GroupSim_i > groupThreshold$，$groupThreshold$ 为预定义的阈值：

(1) 将加入 $shot_i$ 到组 g_{max} 中；

(2) 更新视频的结构：Call updateGroupScene( )；

(3) 返回步骤 2；

否则：

(1) 建立一个仅包含 $shot_i$ 的新组 $Group_i$；

(2) 令 $Groupnum = Groupnum + 1$；

4 测试 $shot_i$ 是否能组合到现存场景中去

(a) 计算当前镜头 $shot_i$ 与现存场景之间的相似度: Call findSceneSim( )。

(b) $shot_i$ 与现存场景之间的最大相似度：

$$\max SceneSim_k = \max_s SceneSim_{k,s}$$

$$1 \le s \le Scenenum$$

其中 $Scenesim_{i,g}$ 是 $shot_i$ 和 $Scene_s$ 的相似度，令和镜头 $shot_i$ 具有最大相似度的场景为 $S_{\max}$ 。

(c) 测试是否 $shot_i$ 是否能组合到现存场景中：

如果 $\max\ SceneSim_i > SceneThreshold$ ，其中 $SceneThreshold$ 为预定义的阈值：

(1) 将镜头 $shot_i$ 加入到场景 $S_{\max}$ 中；

(2) 更新视频的结构：Call updateScene( )。

否则：

(1) 建立一个仅包含 $shot_i$ 和 $Group_i$ 的新场景 $Scene_i$；

(2) 令 $Scenenum = Scenenum + 1$；

5 返回步骤 2；

函数说明：findGroupSim把相似的镜头组合到一个组中，findSceneSim把镜头（或者仅包含一个镜头的组）组合到场景中；然后，updateGroupScene和updateScene把语义上相关的镜头组合到同一场景。

视频时序结构图构造

步骤：

- 视频解码
- 视频切分
- 关键帧提取
- 视频聚类分析
- 构造时序图
- 按照时序图浏览

构造方法：

视频流的时序结构图 VG 是一个三元组(VC，E，W)，其中：VC 是视频流中的所有视频类的集合，E 是 VC 中元素之间关系的集合，W 是权值的集合；E 中的每个元素对应一条有向边 $e_{ij}=<vc_i, vc_j>$，它表示存在两个相邻镜头 sh_k，sh_{k+1}，$sh_k \in vc_i$，$sh_{k+1} \in vc_j$，$vc_i \in VC$，$vc_j \in VC$，且 $i \neq j$；W 中的每个元素 w_{ij} 表示有向边 e_{ij} 的重复次数。

如存在分解后的视频流 $V_{sh}=\{sh_0, sh_1, sh_2, sh_3, sh_4, sh_5, sh_6, sh_7, sh_8, sh_9, sh_{10}, sh_{11}\}$；

聚类后的视频流包含五个视频类 $V_{vc}=\{vc_0, vc_1, vc_2, vc_3, vc_4\}$；

其中：$vc_0=\{sh_0, sh_2, sh_5\}$，$vc_1=\{sh_1, sh_4, sh_7\}$，$vc_2=\{sh_3, sh_6\}$，$vc_3=\{sh_8, sh_{10}\}$，$vc_4=\{sh_9, sh_{11}\}$；

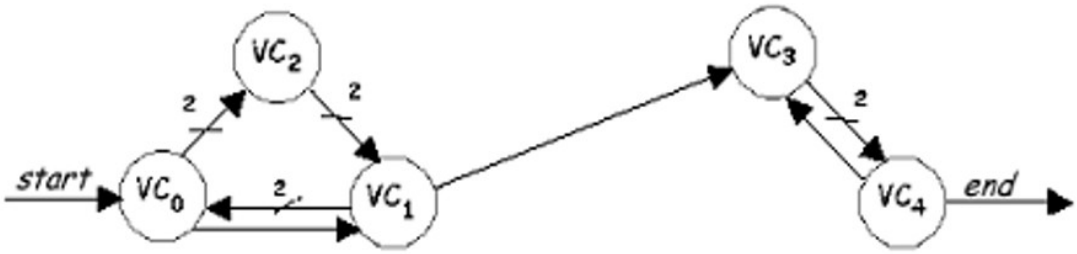
则其时序结构图 VG=(VC，E，W)，其中：

$VC=V_{vc}=\{vc_0, vc_1, vc_2, vc_3, vc_4\}$；

$E=\{e_{start}, e_{01}, e_{10}, e_{02}, e_{21}, e_{10}, e_{02}, e_{21}, e_{13}, e_{34}, e_{43}, e_{34}, e_{end}\}$；

$W=\{w_{01}=1, w_{10}=2, w_{02}=2, w_{21}=2, w_{13}=1, w_{34}=2, w_{43}=1\}$。

e_{start}，e_{end} 是为了图示方便而加入的虚拟边。



视频例子的相似匹配

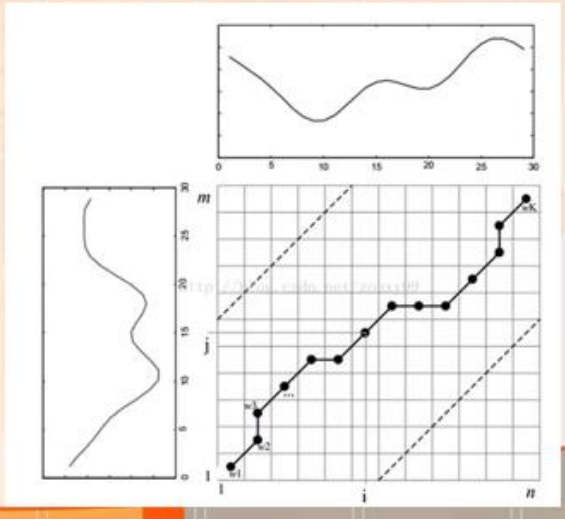
常用弹性度量算法，本质上是比较包含元素数目不同的集合之间相似性

弹性匹配是一种采用动态时间变形（Dynamic Time Warp，DTW）技术的非线性匹配算法，已成功地运用于语音识别、签名认证和手写体识别等领域。

动态时间规整：

一种衡量（长度不同）序列相似度的常见方法

- 构造一个矩阵网格，矩阵元素(i, j)表示两个输入序列的第i个元素和第j个元素之间的相似度（距离），使用动态规划算法寻找一条从网格左下角出发到达右上角的最短路径。



高斯聚类（Gaussian Clustering）

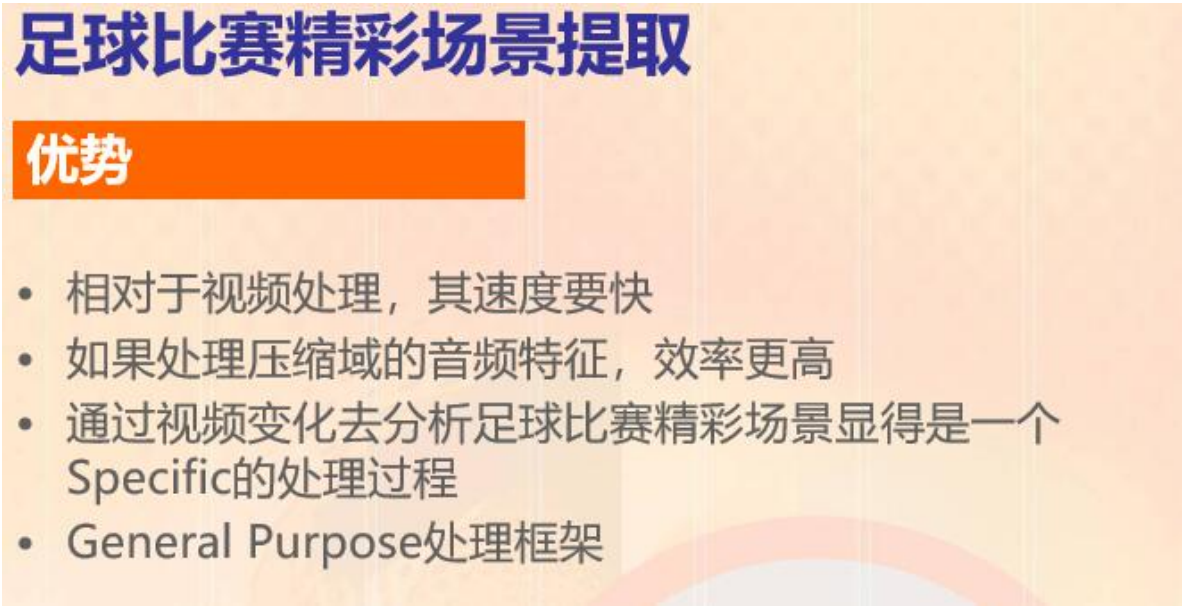
假设存在 $nShot$ 个镜头，下面我们要通过**高斯聚类（Gaussian Clustering）**来将相似镜头合并到一起。$nShot$ 个镜头经过特征提取后，表示为 χ 集合，从 χ 中随机任意选取 3 个行向量作为聚类中心，由 k **平均聚类**得到每个聚类子集新的参数 θ_i：如聚类中心、先验概率、均值、协方差与后验概率。

混合高斯聚类

更好地模拟同一类别中数据的差异性

视频场景分析

例子：足球比赛精彩场景提取



精彩场景提取步骤

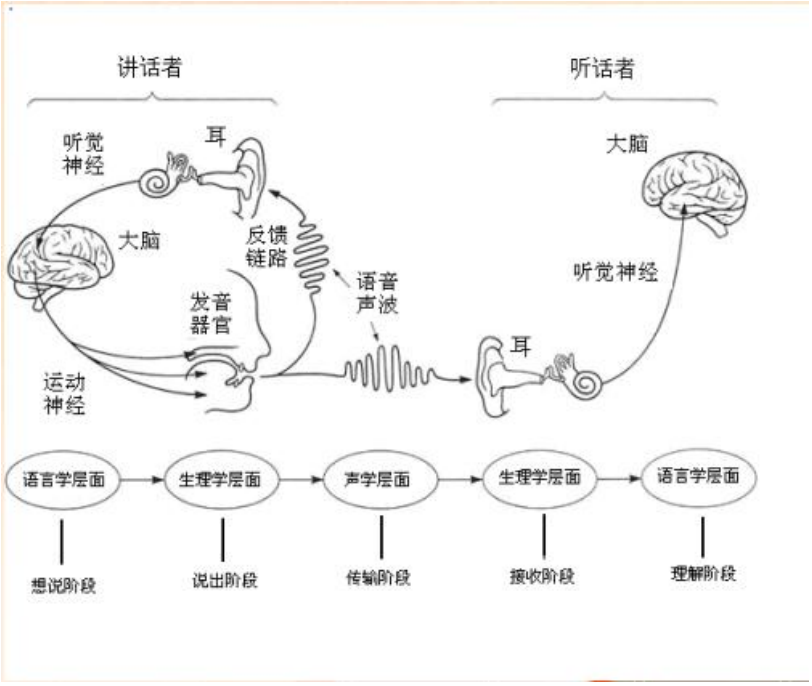
- 提取压缩域音频特征
- 解说员兴奋解说识别
- 现场激昂欢呼声识别

语音

语言处理中的知识

- **语音学与音系学—研究语言的语音**
- **形态学—研究词的有意义的组合**
- **句法学—研究词与词之间的结构关系**
- **语义学—研究意义**
- **语用学—研究如何用语言来达成一定的目的**
- **话语学—研究大于段的语言单位**

语音链



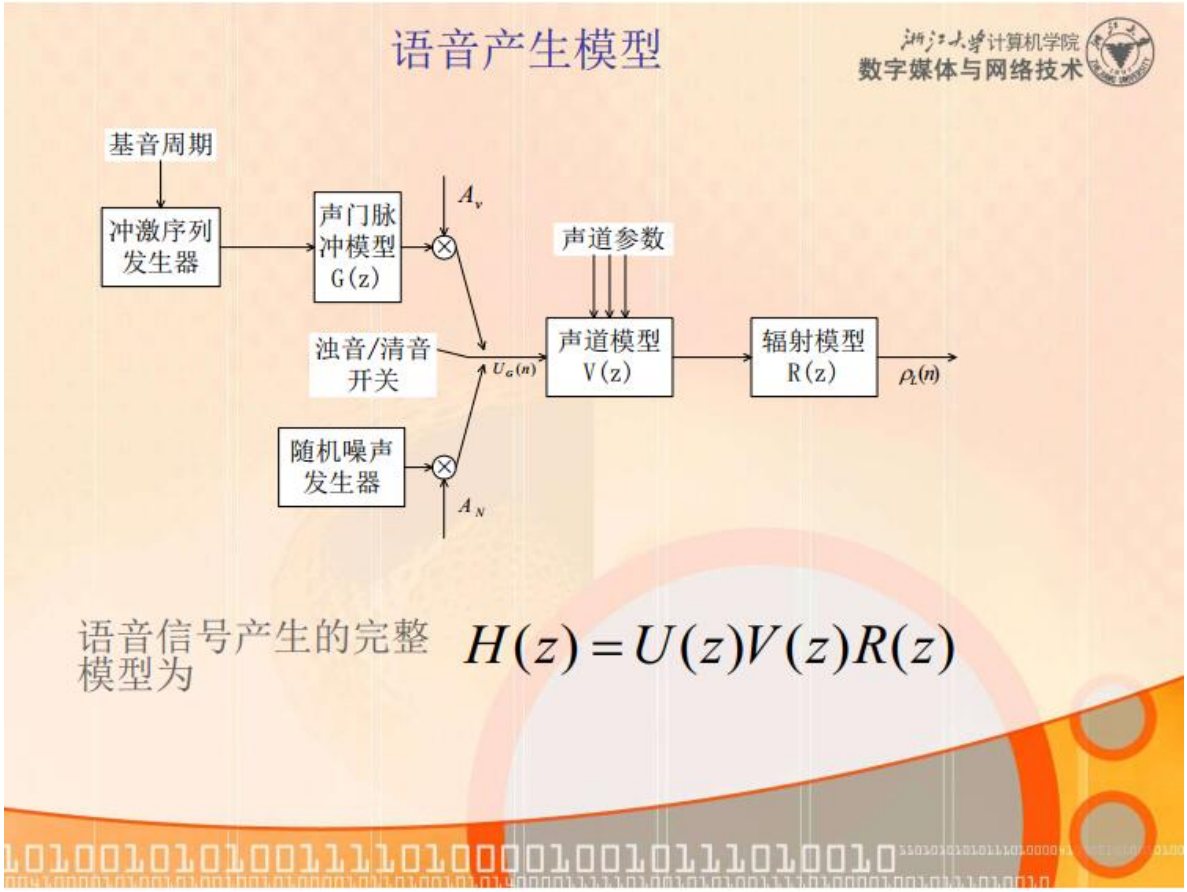
“发音-传递-感知 ”三阶段

语音产生原理

1、动力源：人的呼吸器官
2、发音体：喉是人类专职的发音器官
3、共鸣器 ： 口腔、鼻腔、咽腔

声门振动的快慢，决定声音的基本频率（即音高）。
口腔、鼻腔、舌头的位置、嘴型等，决定声音的内容（即音色）。
肺部压缩空气的力量大小，决定音量大小

语音产生数字模型



重要假设-短时平稳假设

语音信号特性是随时间而变化的，本质上是一个非平稳过程。但不同的语音是由人的口腔肌肉运动构成声道的某种形状而产生的响应，而这种肌肉运动频率相对于语音频率来说是缓慢的，因而在一个短时间范围内，其特性基本保持不变，即相对稳定，可以视作一个准稳态过程。基于这样的考虑，对语音信号进行分段考虑，每一段称为一帧（frame）.一般假设为10-30ms的短时间隔

语音分析技术

语音时域分析

1. 主要参数：

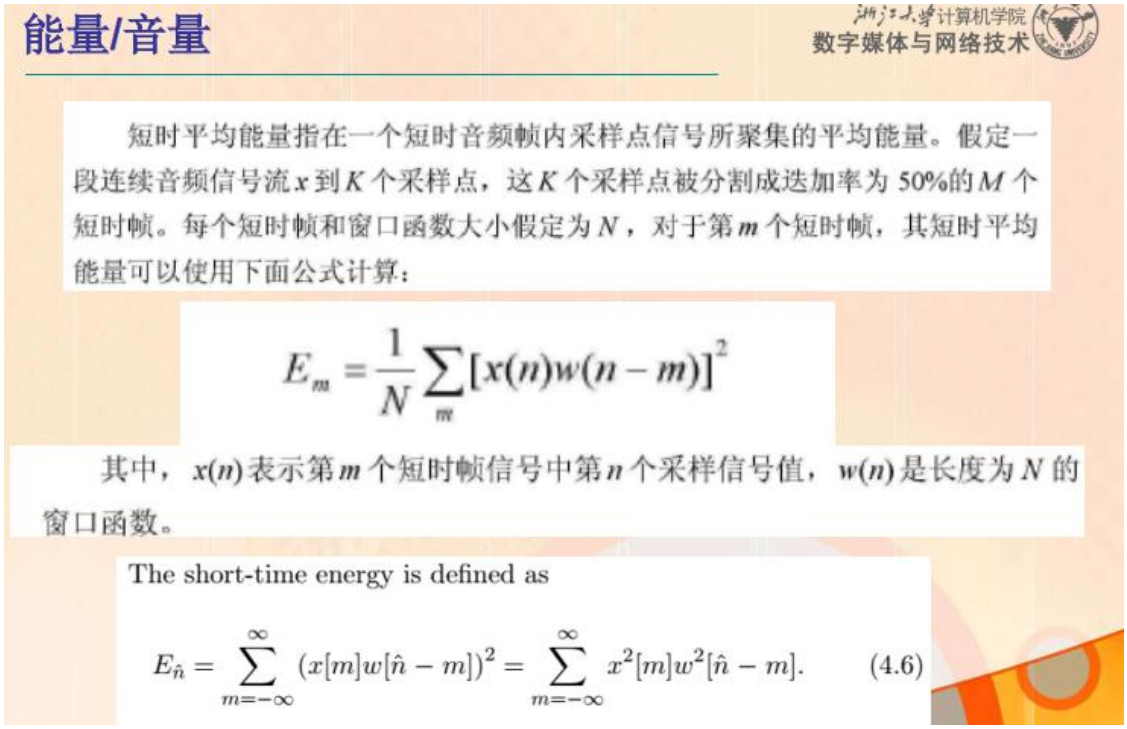
   音量(Volume)、过零率(Zero Crossing Rate)、音高/基音周期(Pitch)

2. 预处理

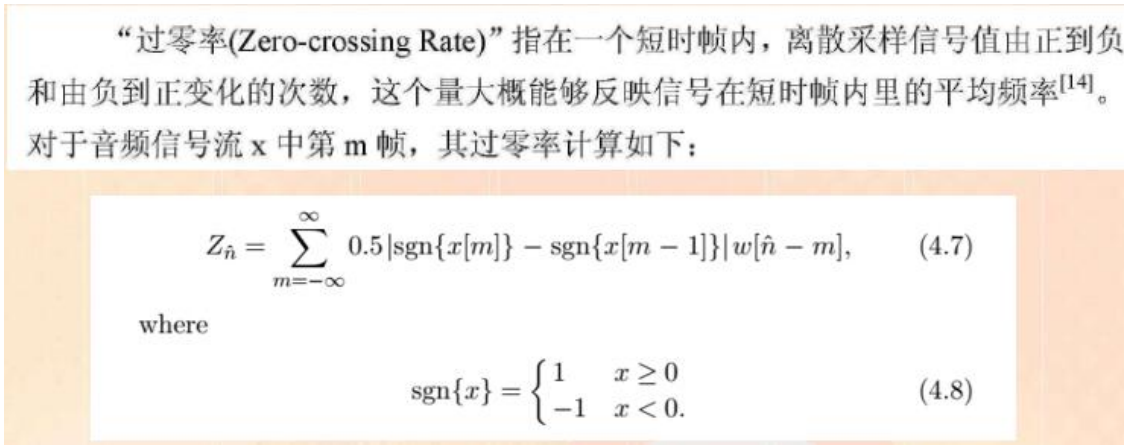
   采样、量化、预加重、短时加窗

   需要提升$H(z)=(1-uz^{-1})$ 其中u取值$0.94-0.97$

3. 能量



4. 过零率



5. 端点检测

“端点检测”（End-point Detection，EPD）的目标是检测语音的开始与结束的位置

端点检测出错，在语音识别上会造成不良后果：
False Rejection：将Speech 误认为 Silence/Noise，造成语音识别率下降
False Acceptance：将Silence/Noise误认为 Speech，造成语音识别率下降。

1.基于短时能量和短时过零率的端点检测：语音段能量比噪声段能量大，可以很好区分；过零率粗略描述频率，判别清音和浊音，有声和无声；过零率检测清音，短时能量检测浊音，两者配合。开始出在静音段，两者之一超过最低门限，进入过渡段，都降到门限之下，回到静音段。两者之一超过高门限，进入语音段，这时若两者都降到门限下，则为噪声段。

2.倒谱特征端点检测。倒谱距离代替短时能量。

6. 基频

自相关法

最低且最强的频率

$$acf(\tau) = \sum_{i=0}^{n-1-\tau} S(i) \bullet S(i+\tau)$$

语音频域分析

1. 主要参数：

   共振峰(Formant)、音高/基音周期(Pitch)

   更好地揭示本质的参数：

   MFCC（梅尔倒谱系数）

   LPCC（线性预测倒谱系数）

2. 分析方法：

   滤波器组法

   傅里叶变化法

   线性预测分析法

3. 短时傅里叶分析

$$X_{\hat{n}}(e^{j\hat{\omega}}) = \sum_{m=-\infty}^{\infty} w(m)x(\hat{n}-m)e^{-j\hat{\omega}(\hat{n}-m)}$$
$$= e^{-j\hat{\omega}\hat{n}} \sum_{m=-\infty}^{\infty} x(\hat{n}-m)w(m)e^{j\hat{\omega}m}$$

   恢复

$$x(\hat{n}) = \frac{1}{2\pi w(0)} \int_{-\pi}^{\pi} X_{\hat{n}}(e^{j\hat{\omega}})e^{j\omega\hat{n}}d\hat{\omega}$$

4. 语图

   横坐标时间，纵坐标频率。

   将语音分成很多帧，每帧对应一个频谱，将其映射到灰度表示，添加时间维度，得到随时间变化的频谱图。

   提取频谱的包络（连接共振峰点的平滑曲线），也就是将频谱分为包络（低频） + 频谱细节（高频）

5. mel频率倒谱系数MFCC

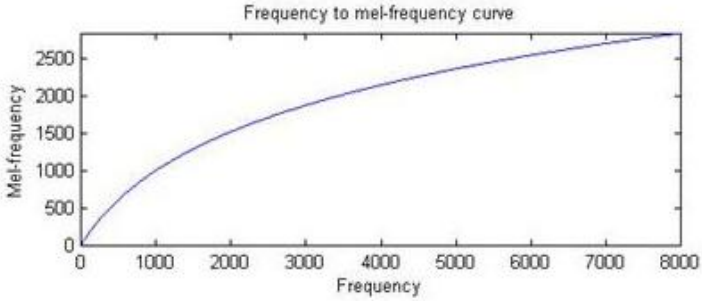
回答这个问题之前，上MFCC的公式

$$\mathrm{Mel}(f) = 2595 \times \lg(1 + \frac{f}{700})$$

f 是频率， $Mel(f)$ 是对应的梅尔倒谱系数。

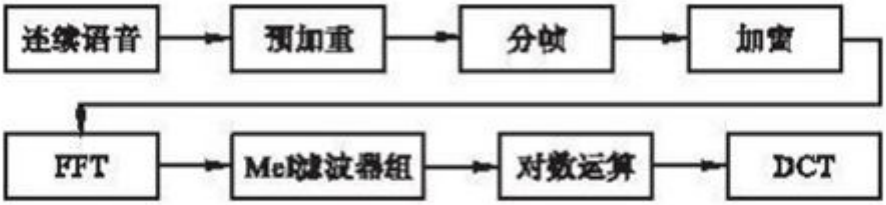
如何理解这个公式：

上式相当于是对信号的频率进行一个非线性处理。如图所示：

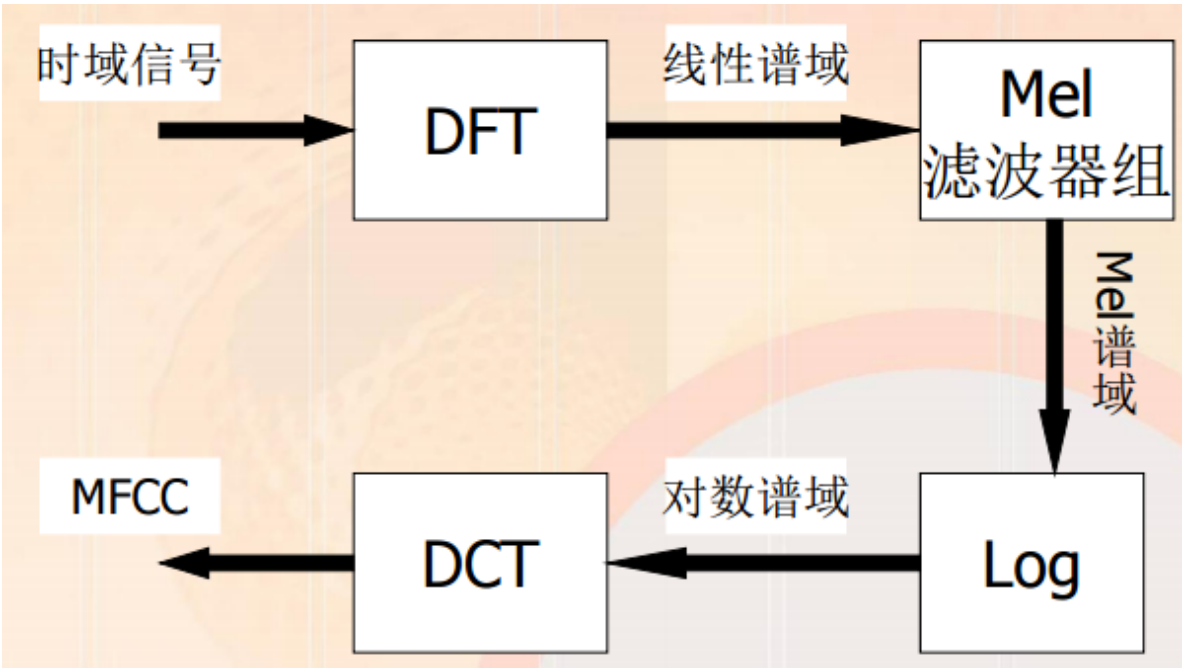


非线性处理起到突出低频，抑制高频的作用

从200HZ到5000HZ的语音信号对语音的清晰度影响比较大，人耳更喜欢低频。



详细过程看https://zhuanlan.zhihu.com/p/60371062



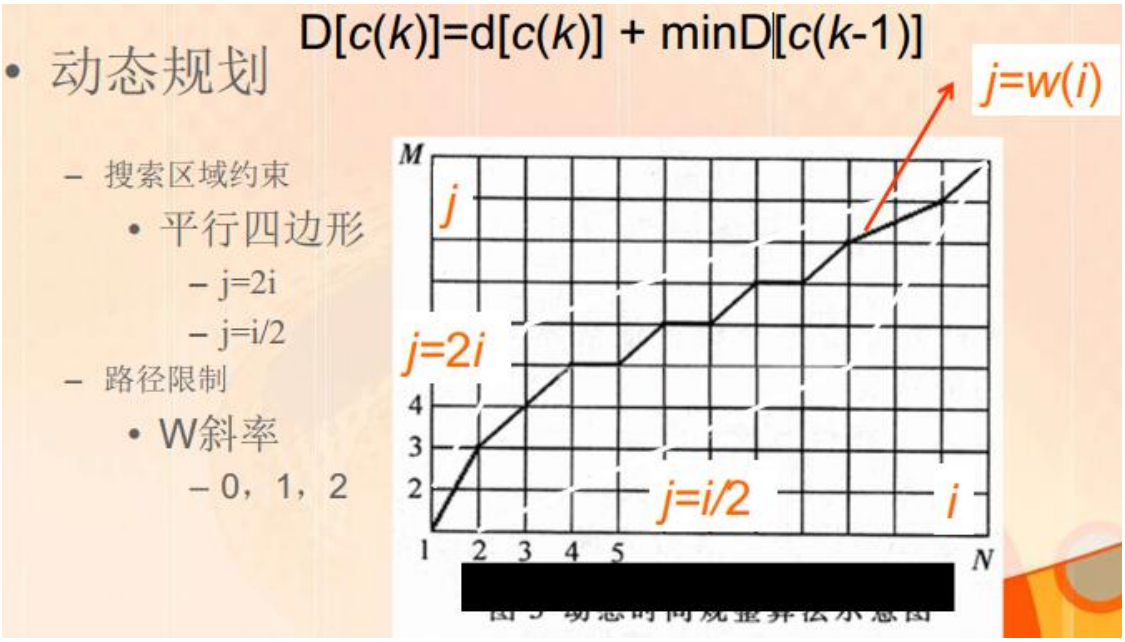
语音识别技术

1. 识别模型：

   动态时间规整(DTW)
   矢量量化(VQ)
   隐马尔科夫模型(HMM)
   神经网络(TDNN)
   模糊逻辑算法

2. 动态时间规整（DTW）

   非线性时间规整模式匹配算法：将时间规整与距离测度结合起来，采用优化技术，以最优匹配为目标，寻找最优的时间规整函数w(i)，从而实现大小(长短)不同的模式的比较



   适用场合：特定人、基元较少的场合，多用于孤立词识别

   问题：

   运算量较大；

   识别性能过分依赖于端点检测；

   太依赖于说话人的原来发音；

   不能对样本作动态训练；

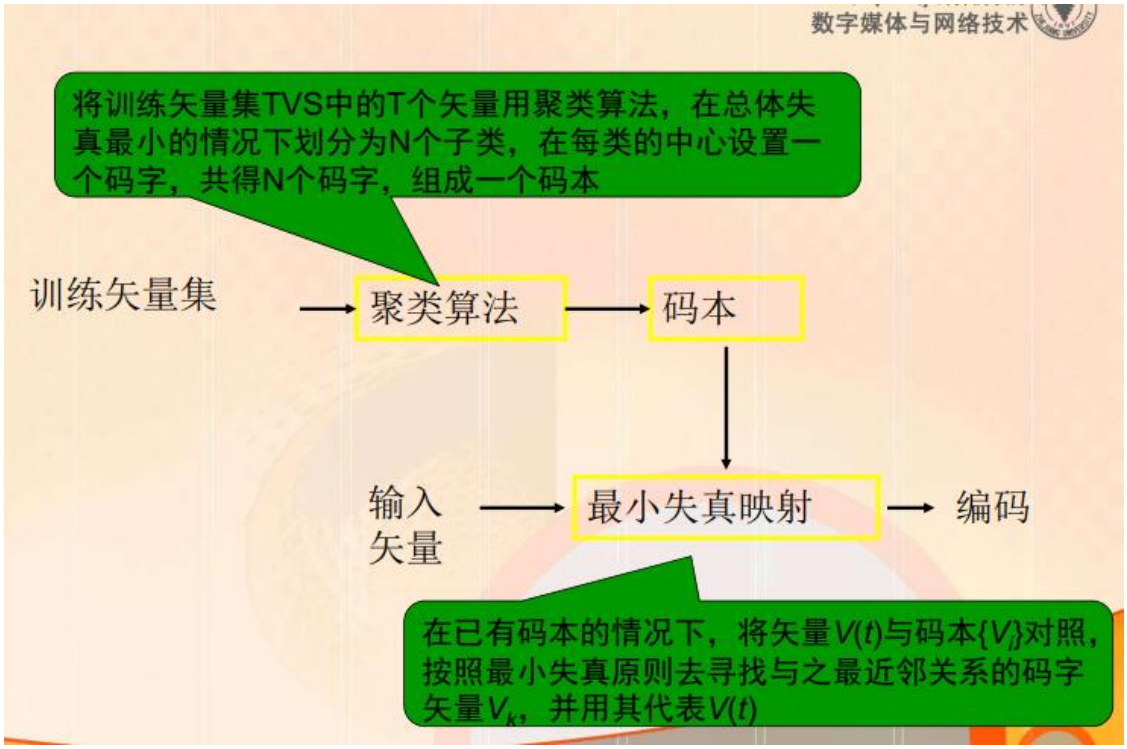
   没有充分利用语音信号的时序动态特性；

3. VQ（Vector Quantization）

   将某一区域（范围）内矢量归为一类。归类主要看靠近哪一矢量。

   两个步骤：

   - 生成码本，这是将语音的特征矢量空间首先进行划分的过程——聚类
   - 将语音参数序列作为矢量，参照码本进行归类的过程 ——量化

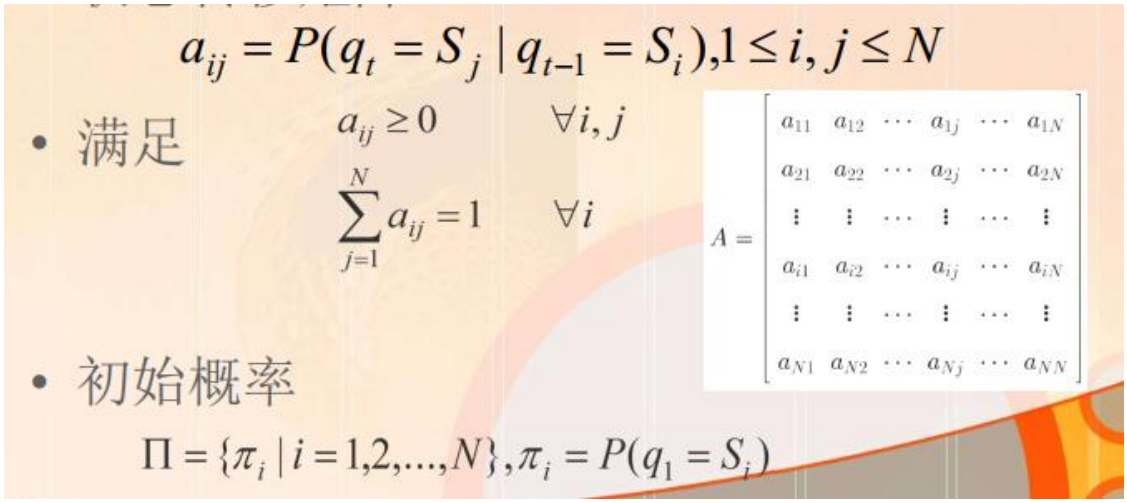


二维矢量空间中，有6类矢量，每一类一个中心（室心），对应一个码字，矢量集合组成码本（codebook）

常用参数：LPCC,,MPCC等的矢量表示。

4. 隐马尔可夫（HMM）

状态转移矩阵A



MM：状态可见，状态即观测结果

HMM：状态不可见，但状态之间的转移仍然是概率的；输出是状态的概率函数