转 镜头分割：像素域方法综述

2013年11月23日 22:41:09　阅读数：7697

本文摘自论文《视频镜头分割方法综述》，列举并比较了几种像素域的镜头分割方法。

1 基于像素的镜头分割算法

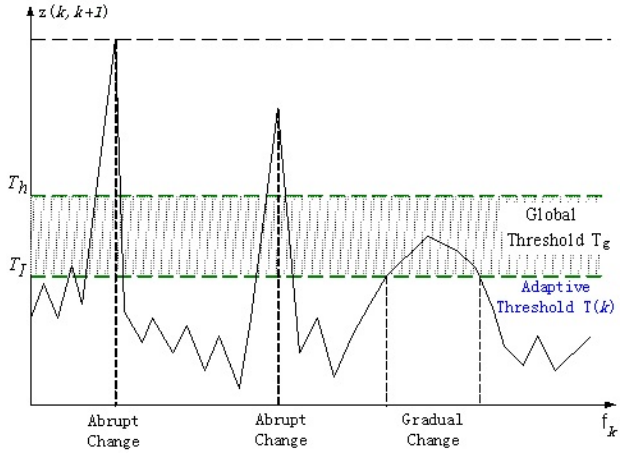
基于像素的镜头分割主要是对视频帧的图像底层处理过程，包括亮度、灰度或者色彩值，其计算简单，原理是计算两帧之间的每个对应的像素的灰度(亮度)的变化。相邻两帧对应点的灰度(亮度)差为

$$f_d(i,j)=\left|f_{n+1}(i,j)-f_n(i,j)\right|$$

式中，$f_n(i,j)$、$f_{n+1}(i,j)$分别代表第n 帧和第n+1 帧像素(i, j)的灰度(亮度)，则相邻两帧之间的总帧差为

$$F_d(f_n,f_{n+1})=\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}f_d(i,j)$$

式中，M、N 为图像的尺寸。若总帧差大于某一设定阈值，则判断镜头内容发生变化。该方法原理简单、便于实现。缺点是对于摄像机及镜头内运动物体、光线条件的剧烈变化敏感，容易误检。



2 基于直方图的镜头分割算法

基于直方图的算法是最普遍的分割方法，它实现简单方便，而且对于大多数视频能得到较好的效果。基于直方图的算法是在基于像素的比较上发展来的，基于直方图的算法通常是将相邻帧之间的各个像素的灰度、亮度等分为N 个等级，再针对每个等级统计像素数做成直方图比较，给出两个图像的直方图，则直方图帧差计算公式如下所示：

$$D=\frac{1}{2N}\sum_i\left|h_m(i)-h_n(i)\right|$$

其中， *N* 为图像帧像素的总数。$h_m(i)$-$h_n(i)$表示的是两个视频帧在 *i* 这个直方图单位上面的距离。基于直方图法不考虑像素的位置信息，而使用其亮度和色彩的统计值，缺点是对结构不同而直方图却很相近的两帧造成漏检而且对于光线变化比较剧烈的情况下，帧差值会受到很大的干扰。

3 基于X2直方图的镜头分割算法

X2　直方图[5]的算法因能放大最大帧差及算法比较稳定而得到广泛应用，为了使直方图帧差更好反映两帧间的差别，可把直方图帧差按下式归一化为：

$$X^2=\begin{cases}\sum_{i=1}^{k}\dfrac{(h_m(i)-h_n(i))^2}{\max(h_m(i),h_n(i))}, & (h_m(i)\neq 0\vee h_n(i)\neq 0)\\ 0, & else\end{cases}$$

式中，k 为图像帧像素的总数，$h_m(i)$-$h_n(i)$表示的是两个视频帧在i 这个直方图单位上面的距离。可见，X2 值越大，则两帧间差异越大; 反之则越小。直方图的归一化也叫直方图均衡化，是通过使用累积函数对灰度值进行“调整”以实现对比度的增强。直方图均衡化处理的“中心思想”是把原始图像的灰度直方图从比较集中的某个灰度区间变成在全部灰度范围内的均匀分布。直方图均衡化就是对图像进行非线性拉伸，重新分配图像像素值，使一定灰度范围内的像素数量大致相同。简单说就是把给定图像的直方图分布改变成“均匀”分布直方图分布。其缺点：①　变换后图像的灰度级减少，某些细节消失;　②某些图像，如直方图有高峰，经处理后对比度不自然的过分增强。算法对于摄像机及镜头内物体运动具有良好的容忍程度，但是实现复杂，计算复杂度高。

4 基于X2直方图分块的镜头分割算法

X2直方图分块法顾名思义是在X2基础上改进的，近几年对X2 直方图分块的研究比较少，为了减少运动、光照等引起的帧差值的变化，本文将各帧分块处理，比较每个块的直方图，将差值最大的块剔除，剩下的块比较帧差异值，其计算公式如下：

$$X^2=\begin{cases}\sum_{i=1}^{k}\sum_{j=1}^{p}\frac{(h_m(i,j)-h_n(i,j))^2}{\max(h_m(i,j),h_n(i,j))}, (h_m(i,j)\neq 0 \vee h_n(i,j)\neq 0)\\ 0, else\end{cases}$$

式中， k 为图像帧像素的总数， p 为块数，$h_m(i)-h_n(i)$表示的是两个视频帧在 i 这个直方图单位上面的距离。缺点是计算时间较X2 直方图时间长，计算复杂度高，优点对于运动的物体有良好的容忍程度。

5 基于边缘轮廓变化率的镜头分割

此方法的主要思想是通过计算边界的变化程度来确定镜头的边界。首先利用canny 算子将图像边缘化，然后计算出帧间的总体位移，以此进行配准，然后计算边缘的数量和位置。帧差由边缘变化的比例表示，即边缘从一帧到另一帧移进和移出的比例。但是由于该方法是先边缘化再进行配准最后才进行边缘比较，因此此方法对于运动时稳健的但计算比较复杂。设 Qk 为 k 帧中与 $k+1$ 帧中最近边缘的距离大于给定阈值 T 的边缘像素数目的百分比; 同样设 $Qk+1$ 为 $k+1$ 帧中与k 帧中最近边缘的距离大于给定阈值 T 的边缘像素数目的百分比，则帧差为：

$$D=Max(Q_k,Q_{k+1})$$

6 实验结果与分析

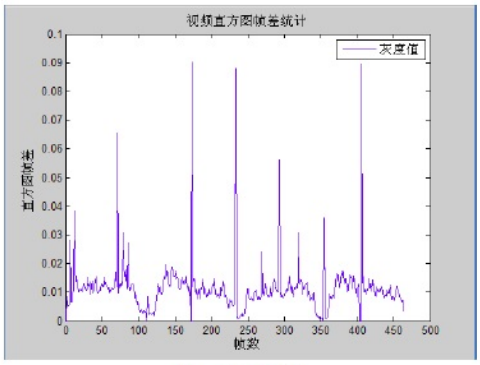
首先，选择2个视频剪辑作为实验数据，它们的格式都是avi 格式。第一个视频时长18 秒，共有465帧; 另一个时长17 秒，共有448 帧，播放速度都是25帧/秒，每个视频都有三个人走动，本文就是将每个人走入镜头和走出镜头的镜头边界检测出来。本文比较了两个算法的运行速度和准确率，实验在Window XP 系统下利用 Matlab 运行五个算法，对上述的视频段进行镜头检测。五种算法的运行速度和准确率如表所示。

表 1　五种算法性能的比较

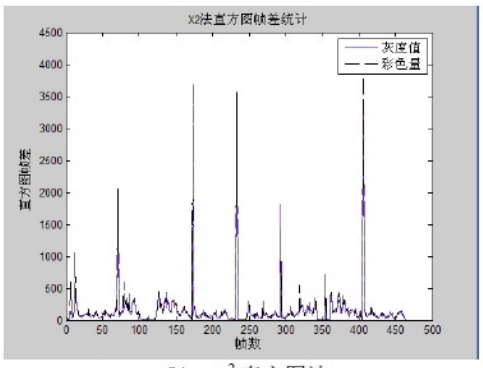
视频	方法	实际检测镜头数	正确检测镜头数	计算时间（秒）	准确率
视频 1	像素法	5	4	13.3	80%
	直方图法	12	6	12.1	50%
	X^2直方图法	5	4	12.5	80%
	X^2直方图分块法	6	5	15.2	83%
	边缘轮廓变化率法	7	4	55.6	57%
视频 2	像素法	5	3	12.4	60%
	直方图法	8	4	12.8	50%
	X^2直方图法	6	4	12.4	67%
	X^2直方图分块法	7	5	14.6	71%
	边缘轮廓变化率法	6	3	55.0	50%

由表1 可知像素法、X2 直方图法和X2 直方图分块法准确率都比较高，其中X2 直方图分块法准确率最高但是时间比像素法和X2 直方图长。而直方图和边缘轮廓变化率法准确率相对较低，但是边缘轮廓变化率法时间比直方图法时间长4 倍多。由于光照变化影响，以上五种算法性能都没有很达到最优。如图2 展示了第一段视频的其中四种镜头检测的

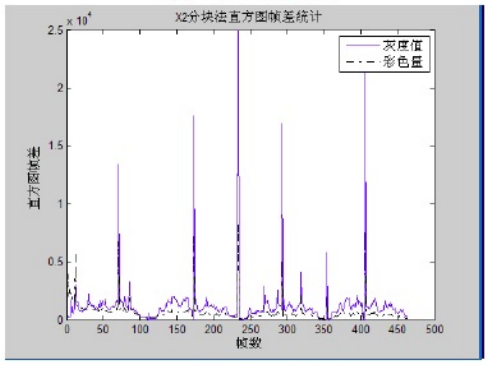
结果比较：



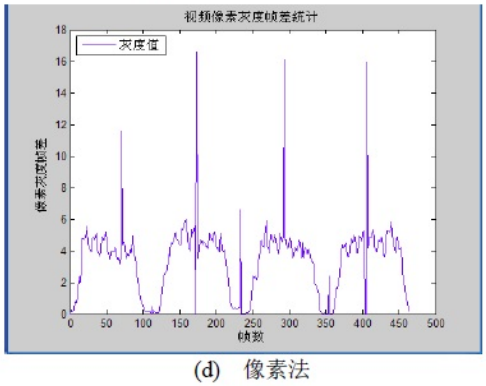
(a) 直方图法



(b) X^2 直方图法



(c) X^2 直方图分块法



(d) 像素法

图 2 四种镜头检测比较

由图 2 曲线图可知镜头内帧差异值变化不大，但是由于光照和快速运动也会发生很大变化，镜头和镜头的边界也发生了很大的变化，怎么样区分是否镜头边界这几种算法都没有得到很好地诠释。而且选取的阈值也不是唯一的，如直方图选择的阈值是0。025 渐变的阈值是0。013，X2 直方图法选择的是800 渐变的阈值是175，这些值都是根据经验得到的，没有一种通用的算法来计算阈值，这样得到的结果也不是最完美的。

文章标签： 镜头分割 视频 算法 matlab

个人分类： MPEG7/图像检索