CUDA框架下的视频关键帧互信息熵多级提取算法

郝晓丽，高 永

(太原理工大学信息与计算机学院 山西 晋中 030600)

【摘要】在传统视频关键帧提取过程中，需要对每一帧视频图像进行特征提取、图像匹配、重复检测等大量计算，导致算法运行时间过长。对此，该文提出了CUDA框架下的关键帧互信息熵多级提取算法。在CPU调度及GPU划分线程基础上，依据帧间三通道互信息熵，将视频序列初次划分为静态片段类和动态片段类；运用相邻帧间互信息量极小值法，将动态片段划分成多个关键子类，在关键子类中选取预备关键帧；并运用SUSAN算子分块计算，快速完成帧间的边缘匹配，从预备关键帧中滤除冗余，得到最终的关键帧序列。实验结果表明，与其他算法相比，该算法的查全率和查准率均为91%以上，提取关键帧的数量平均减少约42.82%，降低了视频数据量的存储，与其他CPU串行方法相比，其关键帧提取时间减少约50%，提高了算法运算效率。

关 键 词 CUDA； 关键帧提取； 互信息熵； SUSAN算子； 视频分割

中图分类号 TP39 **文献标志码** A doi:10.3969/j.issn.1001-0548.2018.05.014

Mutual Information Entropy Multi-level Extraction Algorithm of the Video Key Frame with CUDA

HAO Xiao-li and GAO Yong

(College of Information and Computer, Taiyuan University of Technology Jinzhong Shanxi 030600)

Abstract The video key frame extraction involves feature extraction and matching, it easily leads to high computation complexity. The paper proposes mutual information entropy multi-level extraction algorithm with compute unified device architecture (CUDA). Under CPU scheduling and GPU partition thread, three-channel mutual information entropy among the frames is designed to divide the video clips into the static and the dynamic fragment coarsely. By minimum value method for inter-frame mutual information, the dynamic fragments are categorized into multiple subclasses further, from which pre-key frames are selected. Furthermore, in order to filter out the redundancy of the pre-key frames, the SUSAN operator based on block computing is used to complete the edge matching among the inter-frames, and the final key frame sequence can be obtained by the threshold setting. The experiment results show that, compared with the other algorithms, the precision and the recall ratio of the algorithm in the paper are at least 91%, and the amount of the key frames extracted is reduced by 42.82%. It greatly cut down the video data and saves storage space. Besides it, compared with the CPU serial method, the extraction time by CUDA is shorted by about 50% and it improves the efficiency.

Key words CUDA; key frame extraction; mutual information entropy; SUSAN operator; video segmentation

在互联网、多媒体技术快速发展的今天，视频数据的存储、传输及处理已成为人们获取信息的重要途径之一。但是，互联网上的视频数据量呈指数级增长，要从海量视频数据中快速、高效地获取有效信息，已成为亟待解决的问题。因此研究并设计一种在海量视频数据中高效、灵活地提取关键帧技术，显得尤为重要。在视频检索和关键帧提取的研究领域中，其主要算法研究集中在特征提取和特征匹配两个关键步骤上。在特征提取方面，文献[1]提出了新的基元相关性描述子，通过颜色差分特征和基元频率特征分别描述像素间的对比度和空间位置信息，充分考虑这两种特征在图像检索精度上的作用，但其完全独立的特征提取导致计算量过大。文献[2]提出了基于时空特征聚类的视频镜头分割方法。该方法运用颜色和亮度作为单一帧图像的时间域特征，而对相邻两幅帧在空间域上的特征比较作

收稿日期：2018–01–22；修回日期：2018–06–19
基金项目：国家自然科学基金(61572345)
作者简介：郝晓丽(1973–)，女，博士，副教授，主要从事图像处理与模式识别、人工智能、数据挖掘等方面的研究.

为二次深度特征，将此三维空间上的特征进行融合并聚类，由此得到镜头边界。该算法能够很好地提取各类视频图像特征，并实现平滑镜头的准确检测，但会增加时间复杂度。在特征匹配方面，文献[3]提出了图像序列的多视图匹配算法，为提高匹配精度，先对变形图像进行矫正并重采样，再运用Harris算子进行特征提取，并计算在极线约束下相似图像的相关度系数。文献[4]在处理运动目标时，为避免剧烈运动对其稳定性的干扰，运用SUSAN对图像的角点特征进行提取，取得很好的效果。在关键帧提取方面，文献[5]提出基于显著性特征的关键帧提取策略。文献[6]设计了基于视频分析的辅助驾驶系统，从帧中提取人眼所敏感的空间、光谱等瞬时信息，运用熵驱动基于内容特征的数据融合，以此判断关键帧中静态图像边界及动态目标的出现。文献[7]运用导向图表示两幅预备关键帧的兴趣点矩阵，并运用模块聚类的方法提取关键帧。上述算法都在一定程度上改进了关键帧的提取速度，然而随着海量视频数据量的增加，算法的时间复杂度依然过高。

近年来，利用GPU进行计算加速，已成为图像领域快速计算的首选。文献[8]提出一种CUDA并行计算下基于扩展SURF的多摄像机视频融合方法；文献[9]提出一种基于关键帧的分布式视频分析解耦机制；文献[10]提出一种基于CUDA的多帧图像并行快速处理方法。上述提出的关键帧提取算法，利用CUDA的并行技术，极大地提高了关键帧提取的速度，缩短了整体的时间开销。但是随着视频分辨率的提高，在提取关键帧时，仍有待研究一种更为高效、快速的关键帧提取算法。

针对现有算法中存在的诸多问题，本文提出一种基于CUDA并行处理的关键帧互信息熵提取算法，来解决视频关键帧提取时间过慢的问题。

1　基于CUDA构架的并行计算

1.1　CUDA编程模型

CUDA[11]图形计算设备内部包含多个采用单指令多线程(single instruction multiple threads, SIMT)结构的流处理器。在CUDA编程模型中，以CPU为主机、GPU为设备，通常它们处于一种并行状态。根据其自身的特性，可将任务分别分配给CPU和GPU，前者主要承担逻辑运算和串行计算，而后者则承担大规模的并行数据处理任务。在CUDA并行阶段中，运行在GPU上的kernel函数指定所有执行线程的代码。当CUDA运行且启动一个kernel函数时，一个两级层级结构的网格即被生成。每个网格都是由线程块组成的数组，且其含有大小一样的线程块。在kernel函数启动时，主机代码指定每个线程块的线程数量，其至多包含1 024个线程。基于CUDA模型处理视频时，视频的读取、解码及存储等一系列的操作都是由CUDA的主程序控制，分配不同的任务给GPU处理，最后再返回给CPU执行。

1.2　CUDA并行架构设计

为了实现并行且缩短时间开销，本文基于CUDA框架，将视频数据处理划分为两个独立的阶段：视频处理阶段和关键帧提取阶段，其设计实现如下。

1) 以实现视频文件的解耦合为目的，本文采用OpenCV读取视频数据，通过调用VideoCapture函数，读入的二进制数据被解析为视频流，并进一步对此时的视频流解码，从解码的数据包中，可依次获取视频图片序列。鉴于视频帧中相邻图片序列之间信息差异度很小，本文采用VideoProcessor函数，读取数据流并进行相邻图片序列距离计算，得到足以代表该视频片段内容的视频帧，从而降低帧信息的冗余。

2) 将视频数据转换为图像数据时，采用VideoCapture将视频数据的二进制数据流进行读取，调用外部处理库，解压得到视频帧图片，便于下一阶段的处理。但由于视频数据采用高度压缩的数据形式，通过解码视频帧，得到的数据量将呈几何倍扩大。例如处理每秒25帧的彩色视频，其分辨率为512×512，将其解压为图片后，一秒钟的数据量为512×512×8×3×25 bit，即19.66 MB。巨大的数据量会在后期处理视频帧的数据存储、管理上占用极大空间和时间。

3) 为了解决视频解压后数据量倍增的问题，本文采用CUDA并行处理的方式，优化视频转换为视频帧的过程。本文提出了基于CUDA的CPU+GPU的帧级并行计算架构模型，将当前帧和参考帧拷贝到内存，鉴于视频解压后的图片序列中相邻图片的差异度非常小，本文在读取数据流时先将当前帧和参考帧读入主机端的内存, 并绑定到纹理内存；然后利用GPU多线程计算两帧图像的互信息熵值，从而得到解压图片的差异值；最后把互信息值拷贝到主机的CPU中，根据确定的阈值消除初始冗余帧。基于此过程选出能够代表相邻图片的视频帧，从而大大降低图片的数据量。

2 基于CUDA并行的关键帧粗提取

2.1 视频镜头分割

镜头分割是分析视频序列、以及对大规模视频数据进行有效检索和浏览的基础步骤。所以，能否准确定位镜头边界，并将视频分割为镜头集合，对关键帧的提取、减少索引数据量及高效检索，都有重大意义。传统的基于直方图的算法、基于像素的算法，亦或是基于运动特征的算法、基于边缘特征的算法，大都先对视频进行解压缩，再对其视频特征进行分析理解。而本文视频镜头片段分割的方法，采用文献[12]提出的在MPEG域上，直接获得视频信息(如各子块的DCT系数及预测向量等)，以此作为依据来检测镜头边界，此处不再赘述。

2.2 基于互信息熵的关键帧粗提取

在视频镜头分割的基础上，需进一步提取视频镜头中的关键帧。其中，最为关键的设计是要在确保冗余信息较少的前提下，提取出的关键帧序列，能够较为准确地描述视频的主要特征。相比基于颜色直方图、纹理、轮廓等提取方法，本文以文献[13]中提出的信息熵提取算法为启发，采用信息熵特征进行视频图像特征提取，其步骤如下：

假设通过上述镜头边界检测方法进行了视频片段分割，依赖于检测到的镜头边界，视频数据 S 被划分成为了视频片段，记为 $S=\{S_1,S_2,\cdots,S_N\}$，N 为视频分割后总的镜头片段。

步骤1) 从OpenCV中依次读入分割后的视频片段 $S_1,S_2,\cdots,S_N$，CPU调度整个过程，需要将当前帧和参考帧，分别读入主机端的内存，并将它们拷贝至内存, 且绑定到纹理内存。鉴于各个像素点的计算是相互独立的，可以使用GPU加速运算。若GPU线程块的大小为16×16，且一个像素的计算需占用一个线程，则一个线程块可以同时计算1个宏块中4个8×8块的互信息值，由此获得16×16，16×8，8×16块的互信息值。以此方法，可获得片段中相邻两帧的互信息值 $I_N=\{I_{1,2},I_{2,3},\cdots,I_{k,k-1}\}$。

步骤2) 计算两帧之间的互信息值，根据阈值降低片段内相似度较高的帧群的冗余。本文通过R、G、B三通道采集的互信息量求和的方法，可以更准确地衡量相邻两帧之间的相似性，其互信息量越大，表示两帧越相似。分别用 $I_{X,Y}^R,I_{X,Y}^G,I_{X,Y}^B$ 表示 X 帧和 Y 帧之间的三通道互信息值[14]。计算如下：

$$I_{X,Y}^R=\sum_{i=0}^{L-1}\sum_{j=0}^{L-1}P_{X,Y}^R(i,j)\log\frac{P_{X,Y}^R(i,j)}{P_X^R(i)P_Y^R(j)} \tag{1}$$

$$I_{X,Y}^G=\sum_{i=0}^{L-1}\sum_{j=0}^{L-1}P_{X,Y}^G(i,j)\log\frac{P_{X,Y}^G(i,j)}{P_X^G(i)P_Y^G(j)} \tag{2}$$

$$I_{X,Y}^B=\sum_{i=0}^{L-1}\sum_{j=0}^{L-1}P_{X,Y}^B(i,j)\log\frac{P_{X,Y}^B(i,j)}{P_X^B(i)P_Y^B(j)} \tag{3}$$

式中，$L=256$，那么两帧图像的互信息熵表示为：

$$I_{X,Y}=I_{X,Y}^R+I_{X,Y}^G+I_{X,Y}^B \tag{4}$$

式中，$P_X(i)$ 和 $P_X(j)$ 分别为 X 帧和 Y 帧图像灰度的概率分布；$P_{X,Y}(i,j)$ 为 X 帧和 Y 帧的联合概率分布。CPU调度读取视频及计算互信息熵，并将其值拷贝到主机端的内存，其过程都分解为一系列的矩阵运算。在CUDA中基本的矩阵乘法都使用带状划分法划分线程，每一个线程负责读取相乘矩阵中第一个矩阵的一行以及第二个矩阵的一列，且计算出结果矩阵中对应位置的值。

步骤3) 通过计算片段内互信熵的标准差，判断此片段的动态性。互信息熵的标准差计算如下：

$$\sigma=\sqrt{\frac{1}{N}\sum_{i=1}^{N}(I_i-\mu)^2} \tag{5}$$

式中，I_i 为片段中提取的每相邻帧的互信息熵；μ 为片段内的互信息熵的均值；N 为片段内的互信息熵的个数。通过多次实验，设定阈值λ为1.25，当小于λ时，则判定该片段为静态片段，此时只需提取该片段的第一帧作为关键帧；否则，则判定该片段为包含复杂内容的动态片段。

步骤4) 对于具有复杂内容的动态片段，通常选取多个帧作为片段的关键帧。在本文的方案中，因为互信息量的大小，可以反映图像帧的相似程度。所以对于动态片段，本文首先通过计算动态片段中相邻帧间互信息量极小值的方法，把该动态片段分割成多个子片段。通过子片段的划分，把动态片段中相关性较强的视频帧划分到同一个子片段内，在各个子片段中选择一帧作为关键帧即可。由此，对动态片段中关键帧的提取就转换为子片段内关键帧的提取。

然而并非所有的子片段都要利用起来，实验中只有包含足够帧数的子片段才会被作为关键帧提取的子片段，此类子片段被称为关键类。遵循关键帧只在关键类中选取的原则，本文中划分子片段的帧数要与动态阈值 σ_d 做比较，只有当子片段中的帧数大于动态阈值 σ_d 时，才把其归为关键类。动态阈值 σ_d 的计算公式为：

$$\sigma_d=N_L/(w\times 2) \tag{6}$$

式中，N_L 为片段包含的帧数；w 为片段被划分成为

的子片段数目。

基于互信息熵特征的关键帧提取，得到的只是预备关键帧集合 $F=\{f_1,f_2,\cdots,f_i,\cdots,f_k\}$，其冗余度较高，需进一步运用SUSAN算子协同过滤冗余帧。

3 SUSAN算子协同过滤

3.1 SUSAN算子和边缘匹配

鉴于SUSAN算子[15-16](最小核同值算子)不仅能够检测到图像目标的边界点，能较鲁棒地定位目标的角点，而且不受模板尺寸的影响，可以更加准确地检测到模糊或者平滑的图像边缘，与其他检测算子相比，它更便于计算，且实验参于更易于控制，具有明显的优势。

此处，SUSAN算子运用圆形模板，得到各向同性响应。

设模板函数为 $N(x,y)$，依次将其放在图像中每个点，且将模板内每个像素的灰度值与核的灰度值比较如下：

$$C(x_0,y_0;x,y)=\begin{cases}1 & |f(x_0,y_0)-f(x,y)|\leqslant T\\ 0 & |f(x_0,y_0)-f(x,y)|>T\end{cases} \tag{7}$$

式中，(x_0,y_0) 是核在图像中的位置坐标；(x,y) 是模板 $N(x,y)$ 中其他位置；$f(x_0,y_0)$ 和 $f(x,y)$ 分别是在 (x_0,y_0) 和 (x,y) 处的灰度值；T 是灰度阈值，设定阈值 $T=27$；函数 $C(;,\cdot)$ 代表比较结果。

通过对每个像素进行比较，得到一个输出的游程和：

$$S(x_0,y_0)=\sum_{(x,y)\in N(x,y)}C(x_0,y_0;x,y) \tag{8}$$

运用SUSAN算子时，将游程和 S 与固定的几何阈值 G 比较，并做出判断。设该阈值为 $3S_{\max}/4$，其中 $S_{\max}$ 是 S 的最大值，则图像边缘值 $R(x,y)$ 为：

$$R(x,y)=\begin{cases}G-S(x,y) & S(x,y)<G\\ 0 \quad \text{其他}\end{cases} \tag{9}$$

利用SUSAN算子检测到候选关键帧边缘后，进一步采用边缘匹配率，来甄别相邻帧的边缘是否匹配，用以消除冗余帧。边缘匹配率的公式如下：

$$D(f_i,f_{i+1})=s/n \tag{10}$$

式中，$n=\max(n_{f_i},n_{f_{i+1}})$，$n_{f_i}$ 和 $n_{f_{i+1}}$ 分别为视频帧 f_i 和 f_{i+1} 边缘的像素个数；$s=\sum_{i}^{m}\sum_{j}^{n}R(i,j)$，$R(i,j)=$

$\begin{cases}0 \text{ 其他情况}\\ 1 \ p_{f_k}(i,j)=p_{f_{k+1}} \text{并且} p_{f_{k+1}}(i,j)=p_{f_k}(i,j)=1\end{cases}$，$m$ 和 n 分别为图像的高和宽。$p_{f_k}(i,j)$ 和 $p_{f_{k+1}}(i,j)$ 分别表示帧 f_k 和帧 f_{k+1} 在 (i,j) 处的像素值。$R(i,j)$ 表示相邻帧在 (i,j) 位置处的像素值相同并且值为1；$D(f_i,f_{i+1})$ 是帧 f_i 和帧 f_{i+1} 相似度量值，表示匹配程度，$D(f_i,f_{i+1})$ 的值越大，匹配度越高。

3.2 SUSAN过滤冗余帧

由于每帧图像都是高维特征描述，若对两帧进行比对，将占用大量的时间和空间。因此，本文基于分块的思想，运用SUSAN算子计算相邻帧的边缘轨迹并进行快速匹配，进一步消除冗余。其算法描述如下：

步骤1) 将预备关键帧分割成3×3个图像区域，运用SUSAN算子，在GUP分别分配单独线程的前提下，计算得到各个区域的边缘矩阵；

步骤2) 设j=2，由GPU分配9个单独的线程，分别计算相邻两帧 f_j、f_{j-1} 的各区域边缘匹配率 $D_i(f_{j-1},f_j)(i=1,2,\cdots,9)$，并把其值传给CPU，计算平均匹配率 $\overline{D(f_{j-1},f_j)}$，若 $\overline{D(f_{j-1},f_j)}$ 的值大于等于50%，则此帧 f_j 标记为冗余帧；

步骤3) $j=j+1$，若 $j>k$，转到步骤2)；否则转到步骤4)；当GPU分配的单线程处理完当前任务时，发送指令给CPU，并且将其保存到缓存区，返回继续处理下一帧数据；

步骤4) 为了减少I/O操作，CPU每次将检测到的冗余帧进行标记，当全部任务执行完毕后，从预备关键帧序列中，把全部标记的视频帧删除，获取最终的关键帧序列。

本文算法的执行过程，即关键帧提取算法的流程图，如图1所示。

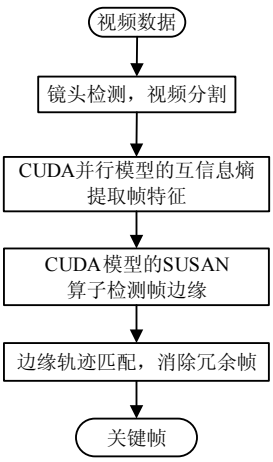

图1　关键帧提取算法流程

4 实验与结果分析

4.1 实验数据和评价标准

本文实验搭建在VS2013环境，使用配置如下：

CPU类型Intel Core i7-7700，CPU主频3.0 GHz，内存容量8 GB，显存容量4 GB，GPU型号为M2000的工作站上运行，使用了NVIDIA CUDA Toolkit5.5和OpenCV2.4.9等软件开发包。为了验证所提算法的性能，本文选取了6种具有不同特征的典型视频流，并选用开源视频数据库(https://open-video.org/)作为测试数据，分别运用本文方法和文献[17]和文献[18]方法对测试数据的检测时间和相关参数进行了对比。

4.2 检索关键帧数量比较

针对表1中的6种实验视频数据，通过比较本文算法、文献[17]算法和文献[18]算法提取出的关键帧数量，从而分析不同算法在提取关键帧时节省的存储容量和减少的数据量，如表1所示。

表1 不同算法提取关键帧的数量

视频类型	视频总帧数	参考关键帧的数量	提取关键帧数量/帧		
			文献[17]方法	文献[18]方法	本文方法
新闻	2 586	23	35	47	22
体育	3 427	34	40	56	32
故事	4 528	45	58	68	45
动画	5 833	48	67	86	48
演讲	4 227	39	51	64	40
会议	5 964	45	70	90	48

从实验结果看出，利用文献[18]的方法提取出的关键帧数量最多。本文算法提取出的关键帧数量相比文献[17]算法提取出的关键帧数量平均减少约26.79%；相比文献[18]算法提取出的关键帧数量平均减少约42.82%。因此，与文献中的两种算法相比，本文提出的关键帧提取算法可以大幅度地减少提取关键帧的数量，从而达到节省存储空间并且降低视频数据量的目的。

4.3 检索时间对比

为了检验关键帧提取算法在CPU与CUDA上实现性能的不同，本文在特征提取过程中分别对CPU和CUDA的处理速度及加速比进行了比较，如表2所示。

表2 特征提取过程中CPU与CUDA上的性能比较

视频种类	特征点的个数/帧	CPU平均用时(帧/ms)	CUDA平均用时(帧/ms)	CPU/CUDA的加速比
新闻	512	132.136 0	1.642 2	80.462 8
体育	1 024	310.455 0	2.285 6	135.830 9
故事	1 024	299.894 1	2.094 7	143.165 0
动画	1 024	329.564 0	2.354 1	139.995 8
演讲	512	129.133 0	1.514 8	85.247 6
会议	512	142.413 2	1.798 4	79.188 8

由表2得知，当初始特征点的个数相同且视频种类一致时，基于CUDA的并行算法比基于CPU的串行算法实现速度平均提高了两个数量级。例如在新闻类视频中，视频帧的初始特征点的个数为512时，CPU/CUDA的加速比达到80.462 8倍。随着图像像素的提高和初始特征点的增加，基于CPU提取视频帧算法的用时显著增加，但是基于CUDA的并行算法用时增量较小，加速比有明显的提高。例如，在体育类视频中，每一帧的特征点的个数是新闻类特征点个数的2倍，虽然两者的时间开销均有所增长，但由于CUDA的应用使得加速比依然得到提高。因此基于CUDA的并行算法适合于像素要求高、视频质量好的关键帧提取过程。

为了检验本文算法在提取关键帧时时间性能的优越性，实验中分别对文献[17]和文献[18]的两种算法与本文算法在提取关键帧时所用时间进行了比较，如图2所示。

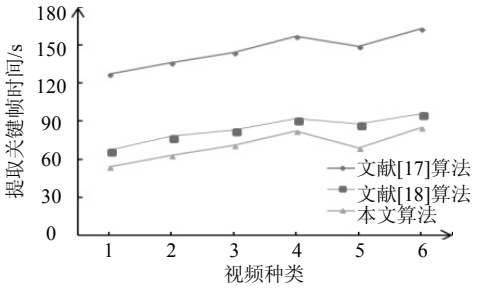

图2 提取关键帧时间对比

从图2中可看出，与文献[17]所用CPU串行方法的关键帧提取时间相比，采用基于CUDA并行模型的检测时间减少约50%。主要原因是本文算法和文献[18]算法都是基于并行计算的模型，在图像特征提取以及消除冗余时采用CPU+GPU并行调度方式，极大地提升了数据处理速度。在基于CUDA并行计算的基础上，本文算法相比文献[18]算法，提取关键帧的检测时间平均减少15.87%。其主要原因是本文算法使用互信息熵作为提取关键帧的特征值，比文献[18]中利用前景和运动对象的局部最大值作为特征值提取关键帧，减少了每帧特征计算和特征对比过程的计算量，从而缩短关键帧检测时间。

4.4 检索参数效果对比

从图2中可以得出，利用本文算法提取关键帧速度最快，时间效率最高。但是得到的关键帧是否可靠、有效，还需要用关键帧提取领域中的查全率和查准率两个重要标准来衡量。

为了更好地显示不同算法的检测效果，表4列出了不同方法的实验结果。该表中包含了不同算法检测出的关键帧数量、正确帧数、漏检帧数、误检帧

数以及冗余帧数等对比参数。

表3　不同算法的检测结果

视频类型	参考关键帧数量	算法	正确检测帧数	漏检帧数	误检帧数	冗余帧数	查全率/%	查准率/%
新闻	23	文献[17]	20	3	2	13	86.97	90.91
		文献[18]	21	2	2	24	91.30	91.67
		本文算法	21	2	1	0	91.30	95.45
体育	34	文献[17]	30	4	3	7	88.24	90.91
		文献[18]	31	3	2	23	91.17	93.94
		本文算法	31	3	1	0	91.43	91.43
故事	45	文献[17]	40	5	3	15	88.89	93.02
		文献[18]	41	4	2	25	91.11	95.34
		本文算法	42	3	3	0	93.33	93.33
动画	48	文献[17]	43	5	5	19	89.58	89.58
		文献[18]	45	3	2	39	93.75	95.74
		本文算法	44	4	3	1	91.67	93.62
演讲	39	文献[17]	35	4	4	12	89.74	89.74
		文献[18]	37	2	3	24	94.87	92.50
		本文算法	36	3	3	1	92.31	92.31
会议	45	文献[17]	40	5	4	26	88.89	90.91
		文献[18]	42	3	2	46	93.33	95.45
		本文算法	41	4	3	4	91.11	93.18

从表3可以看出，本文算法查全率平均可以达到91.86%，查准率平均可达到93.22%。与文献[17]中的算法相比，本文算法的查全率和查准率都有显著的提高。与文献[18]所用算法相比，本文算法在查准率和查全率两方面损失控制在1%左右。主要原因是本文算法在特征提取方面，为降低计算量，提高运算速率，使用互信息熵特征值作为了衡量标准，忽略了不同环境下背景和运动对象的复杂性，但损失的精准度在一定的置信范围内。

4.5　关键帧检测结果对比

本文中选取了一段1分钟左右的体育类视频，分别应用文献[17]、文献[18]和本文方法进行关键帧提取，结果对比如图3所示。

从图3中可以看出，文献[17]和文献[18]方法提取出的关键帧集合不仅存在较大冗余，而且都有明显的漏检现象。而本文算法简洁有效地反映了运动员投篮的全过程。

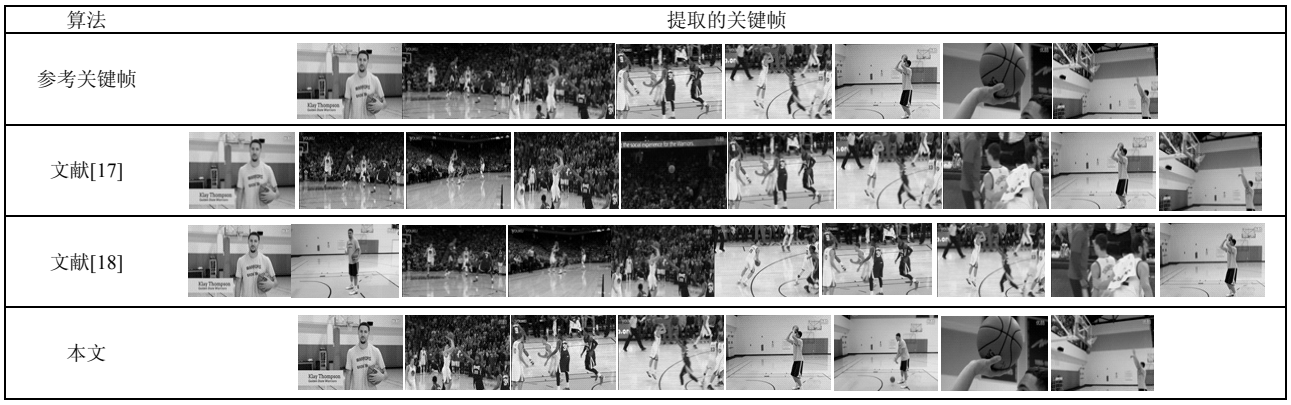

图3　不同算法提取出的关键帧结果

5　结 束 语

本文提出一种基于CUDA并行处理的互信息熵关键帧提取算法，解决了传统的基于CPU串行算法在提取海量视频数据时处理速度过慢的问题，极大地提高了算法的时间效率。另外，本文采用互信息熵提取图像特征，并用SUSAN算子检测边缘特征，相比基于前景和运动特征并行处理的关键帧提取算法，简化了计算过程，进一步地提升了算法的运算效率，并降低了数据量从而节省存储空间。上述6种视频数据集测试结果表明，与基于前景和运动特征的并行关键帧提取算法相比，本文算法具有更高的时间效率和较小的空间开销，对大规模视频数据的检索具有一定的应用价值，特别适合从分辨率较高的视频中提取关键信息，对图像和视频检索领域的研究具有一定意义。

参 考 文 献

[1] 吴俊, 刘胜蓝, 冯林, 等. 基于基元描述子的图像检索[J]. 计算机研究与发展, 2016, 53(12): 2824-2835.
WU Jin, LIU Sheng-lan, FENG Lin, et al. Imag retrieval based on texton correlation descriptor[J]. Journal of Computer Research and Development, 2016, 53(12): 2824-2835.

[2] DUAN Feng-feng. Shot segmentation for binocular stereoscopic video based on spatial-temporal feature clustering[J]. 3D Research, 2016, 7(4): 29-36.

[3] ZOU Xiao-liang, ZHAO Gui-hua, JONATHAN L, et al. Multiview matching algorithm for processing mobile sequence image[J]. Journal of Surveying Engineering, 2017, 143(4): 142-150.

[4] XU Yi-ming, GU Ju-ping, ZHU Hai-rong, et al. An image stabilization algorithm on corner detection and feature block matching[C]//2014 International Conference on Audio: Language and Image Processing. Shanghai: IEEE, 2015: 190-194.

[5] ANTONIO C H, MANUEL C H, FRANCISCO G U, et al. A fast and effective method for static video summarization on compressed domain[J]. IEEE Latin America Transactions, 2016, 14(11): 4554-4559.

[6] MELIH A, MEHMET C. Road scene content analysis for driver assistance and autonomous driving[J]. IEEE Transactions on Intelligent Transportation Systems, 2017, 18(12): 3398-3407.

[7] HANA G, SAHBI B, MOHAMED M, et al. Key frames extraction using graph modularity clustering for efficient video summarization[C]// IEEE Interntaional Conference on Acoustics: Speech and Signal Procssing. New Orleans, LA, United States: IEEE, 2017: 1502-1506.

[8] 崔哲, 孟凡荣, 姚睿, 等. CUDA并行计算下基于扩展SURF的多摄像机视频融合方法[J]. 南京大学学报(自然科学), 2016, 52(4): 627-637.
CUI Zhe, MENG Fan-rong, YAO Rui, et al. Multi-video fusion with extended SURF based on CUDA parallel computing framework[J]. Journal of Nanjing University (Natural Sciences), 2016, 52(4): 627-637.

[9] 蔡晓东, 华娜, 吴迪, 等. 一种基于关键帧的分布式视频分析解耦机制[J]. 电视技术, 2015, 39(14): 1-4.
CAI Xiao-dong, HUA Na, WU Di, et al. Distributed video analysis decoupling mechanism based on keyframe[J]. Video Engineering, 2015, 39(14): 1-4.

[10] AN Yi-yao, GUO Mao-yun, CHAI Yi, et al. The CUDA-based multi-frame images parallel fast processing method[C]//Proceedings of 2016 Chinese Intelligent Systems Conference. Xiamen: Springer, 2016: 593-598.

[11] 邹彬彬, 梁凡. 一种基于CPU+GPU的AVS视频并行编码方法[J]. 上海大学学报(自然科学版), 2013, 19(3): 235-239.
ZOU Bin-bin, LIANG Fan, Parallel implementation of AVS video encoder based on CPU+GPU[J]. Journal of Shanghai University(Natural Science), 2013, 19(3): 235-239.

[12] 章亦葵, 赵晖. 基于预处理的视频镜头边界检测算法[J]. 计算机应用, 2014, 34(11): 3327-3331.
ZHANG Yi-kui, ZHAO Hui. Video shot boundary detection method based on pre-processing[J]. Journal of Computer Applications, 2014, 34(11): 3327-3331.

[13] 岳昕, 尚振宏, 强振平, 等. 基于信息熵与SIFT算法的天文图像配准[J]. 计算机科学, 2015, 42(6): 57-60.
YUE Xin, SHANG Zhen-hong, QIANG Zhen-pin, et al. Astronomical image registration combining information entropy and SIFT algorithm[J]. Computer Science, 2014, 34(11): 3327-3331.

[14] 赵晖. 基于内容的视频镜头边界检测及关键帧提取[D]. 天津: 天津大学, 2014.
ZHAO Hui. Content based video boundary detection and key frame extraction[D]. Tianjin: Tianjin University, 2014.

[15] FAN Xin-feng, CHENG Yuan-zeng, FU Qiang. Moving target detection algorithm based on susan edge detection and frame difference[C]//International Conference on Information Science and Control Engineering. Shanghai: IEEE, 2015: 323-326.

[16] 章毓晋. 图像处理基础教程[M]. 北京: 电子工业出版社, 2012: 173-193.
ZHANG Yu-jin. Fundamental course of image processsing[M]. Beijing: Publishing House of Electronics Industry, 2012: 173-193.

[17] 刘华咏, 李涛. 基于改进分块颜色特征和二次提取的关键帧提取算法[J]. 计算机科学, 2015, 42(12): 307-311.
LIU Hua-yong, LI Tao. Key frame extraction algorithm based on improved block color features and second extraction[J]. Computer Science, 2015, 42(12): 307-311.

[18] ZHENG Ran, YAO Chuan-wei, JIN Hai, et al. Parallel key frame extraction for surveillance video service in a smart city[J]. Plos One, 2015, 10(8): e0135694.

[19] 庞亚俊. 基于先验的动作视频关键帧提取[J]. 河南理工大学学报(自然科学版), 2016, 35(6): 862-868.
PANG Ya-jun. Key frames extraction of motion video based on prior knowledge[J]. Journal of Henan Polytechnic University (Natural Science), 2016, 35(6): 862-868.

编 辑 蒋 晓