文章编号:1004-9037(2012)06-0685-06

画卷形式的视频预览在 OMAP 平台上的实现

乔瑞萍　胡宇平　王效鹏　俞　璐

(西安交通大学电子与信息工程学院,西安,710049)

摘要:实现了一个基于 OMAP3530-EVM 平台的具有画卷形式预览功能的视频播放系统。系统首先通过提取关键帧以获取视频的主要信息,再利用图像拼贴算法将关键帧内容以多尺度画卷的形式呈现给用户,方便用户进行视频的预览。与传统的视频预览方式相比,直观的预览方式可以更为有效地实现视频的预览功能,从而提供良好的用户体验。提出了一种更为高效的图像拼贴算法,用以生成预览所需的画卷。实验结果表明,在具有相同拼贴质量的情况下,这种算法的处理速度比起常见的最大流算法有大幅度的提高。

关键词:视频预览;图像拼贴;关键帧

中图分类号:TN919.8　　**文献标识码**:A

Implementation of Video Preview with Tapestries on OMAP Platform

Qiao Ruiping, *Hu Yuping*, *Wang Xiaopeng*, *Yu Lu*

(School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, 710049, China)

Abstract: A video playing system based on OMAP3530-EVM with collage is proposed. The system gains the main information from the video by extracting key frames. Then, the image collage algorithm is adopted to achieve multi-scale video tapestries with continuous temporal zoom as video preview bar, providing convenience for users to preview the videos. Compared with the traditional video preview, preview in the form of video tapestries can realize video preview more directly and effectively, thus providing a better user experience. A more efficient image collage algorithm is presented to obtain the required video tapestries. The experiment result shows that the collage algorithm takes much less processing time than max-flow algorithm with the same collage quality.

Key words: video preview; image collage; key-frame

引　　言

近年来,随着多媒体技术的飞速发展,越来越多的视频文件出现在人们的日常生活中。因此如何高效地管理和使用这些视频文件已成为一个新兴的研究方向。例如,当用户希望能快速、便捷地了解一段或是整段视频的大致内容以完成诸如查询、评价等类似的任务时,一种高效的视频预览方式将显著地缩短用户浏览和查找所花费的时间。视频预览的任务是将视频进行分析、提炼,并将用户所关心的内容尽可能多地展示给用户。毫无疑问,一个好的视频预览方式将提高用户处理视频的效率。

开放式多媒体应用平台(Open multimedia application platform,OMAP)是 TI 公司为无线通信而研制的一个软硬件开发平台。OMAP 处理器采用 DSP 和 ARM 双核结构,除具备性能功耗比高的优势之外还提供了丰富的外围接口,支持 Linux、WinCE 等高级操作系统[1]。本文选择 OMAP3530-EVM 作为平台并在其上实现一个具有多尺度画卷形式的预览功能的视频播放系统,该系统界面包含了一个视频播放窗口和一个预览窗口,界面示意图如图 1 所示,预览窗口内的图像按照时间顺序排列,并以一种平滑、无缝的形式进行拼贴。用户可以通过鼠标的拖拽、滚动等动作来改变画卷内容的时间位置以及分辨率,还可以通过单击某一图像来完成视频播放位置的跳转。

基金项目:德州仪器创新基金(2009W1201)资助项目。
收稿日期:2012-06-09;**修订日期**:2012-10-08

1 核心算法

为实现上述形式的视频预览功能,需要解决如下两个问题:(1)选取哪些帧来概括视频各个镜头的内容,即关键帧的选取;(2)如何将已选取的关键帧以合理、美观的形式呈现给用户。

1.1 视频中关键帧的选取

关键帧,有时也称为代表帧,是用于描述一个镜头的关键图像帧,它通常会反映一个镜头的主要内容[2]。一个视频中所有关键帧组成的集合便能够从整体上反映该视频的主要内容。显然,一个选取合理的关键帧集合能达到更好的视频预览效果。

如果视频中某一帧的内容非常的单调,那么将其选取为关键帧显然是不合适的。同样,与相邻帧非常相似的帧也不应该被选取为关键帧。针对这两个特点,本文提出关键性评价函数式(1)来定量描述视频中 t 时刻帧的关键性,并将 $K(t)$ 值较大的帧选定为关键帧

$$K(t) = \alpha_1 \cdot G(\boldsymbol{I}_t) + \alpha_2 \cdot H(\boldsymbol{I}_t) \quad (1)$$

式中:$\boldsymbol{I}_t$ 为 t 时刻的帧图像;$G(\boldsymbol{I}_t)$ 描述 t 时刻帧的内容丰富程度;$H(\boldsymbol{I}_t)$ 描述 t 时刻帧与相邻帧的差异程度;α_1, α_2 为可调整参数。

本文用帧图像的像素均值与方差来反映当前帧内容的丰富程度,用相邻帧像素差的绝对值之和来表示当前帧与相邻帧的差异程度。计算公式分别为式(2,3)。

$$G(\boldsymbol{I}_t) = \frac{\| \boldsymbol{I}_t - \boldsymbol{I}_{t+1} \| + \| \boldsymbol{I}_{t-1} - \boldsymbol{I}_t \|}{2} \quad (2)$$

$$H(\boldsymbol{I}_t) = \begin{cases} D(\boldsymbol{I}_t) & 50 < E(\boldsymbol{I}_t) < 200 \\ 0 & \text{其他} \end{cases} \quad (3)$$

其中

$$E(\boldsymbol{I}_t) = \frac{1}{M * N} \cdot \sum_{x=1}^{M} \sum_{y=1}^{N} \left[\frac{\boldsymbol{R}_t(x,y) + \boldsymbol{G}_t(x,y) + \boldsymbol{B}_t(x,y)}{3} \right] \quad (4)$$

$$D(\boldsymbol{I}_t) = \frac{1}{M * N} \cdot \sum_{x=1}^{M} \sum_{y=1}^{N} \left[\frac{\boldsymbol{R}_t(x,y) + \boldsymbol{G}_t(x,y) + \boldsymbol{B}_t(x,y)}{3} - E(\boldsymbol{I}_t) \right]^2 \quad (5)$$

但是,若直接使用上述的关键帧选取方案对于某些视频来说效果并不理想。举例说,对于诸如人物采访一类具有不变背景的视频片段,只有非常少的帧能够被选定为关键帧。这就导致了相邻的两个关键帧之间存在着比较大的时间间距,用户便不能将视频播放位置跳转至这两帧之间的位置,造成使用上的不便。因此关键帧的选取还应该考虑到时间间隔的影响,即:

(1)在时间间隔较大的顶层导航条中,关键帧的选取应注重其内容的代表性。

(2)在时间间隔较小的底层导航条中,关键帧的选取应注重选取时间分布的均匀性。

即对于某一帧来说,其重要性在不同的时间尺度上是不相同的。因此将式(1)加入乘性的控制因子并改写为

$$K(t;t_s,t_e) = \begin{cases} e^{-\frac{\left(t-\frac{t_s+t_e}{2}\right)^2}{2[\alpha_3(t_e-t_s)]^2}} \cdot K(t) & t_s < t < t_e \\ 0 & \text{其他} \end{cases} \quad (6)$$

式中:t_s, t_e 为包含时间 t 的区间的开始时间和结束时间;α_3 为可调整参数。这样,当 $t_e - t_s$ 较小时,控制因子具有较强的集中性,时间尺度带来的影响增加;当 $t_e - t_s$ 较大时,控制因子比较平坦,时间尺度带来的影响减小。从而实现了"较大时间间隔时注重代表性、较小时间间隔时注重选取的均匀性"这一目标。

将一段长度的视频以不同时间间隔进行等分,选取每一个区间中关键性评价函数值最大的一帧作为该区间的关键帧,进而组成不同尺度下的关键帧集合,即可完成各尺度预览所需要的关键帧的选取。

1.2 关键帧图像的拼贴

对于视频播放类软件来说,预览-导航窗口的尺寸不应该过大,以免影响正常的视频播放。为了在有限大小的窗口下尽可能多地显示视频预览的内容,一种简单的方式是缩小预览图像的尺寸;另一种方式是将预览图像以一定宽度重叠在一起并运用图像拼贴技术处理该重叠区域。较前一种方式相比,后者在不改变图像尺寸的情况下达到了增加显示内容的目的。

图像拼接与图像拼贴是两个相近、容易混淆的概念。图像拼接是将数张有相同或近似内容的图像拼成一幅较大的图像;而在图像拼贴中处理的图像往往没有任何公共部分,其更注重的是如何紧凑地、美观地将若干张图像拼合在一起,并保留各图像的主要内容。在本文所述系统中,用于视频预览的关键帧之间的相同部分较少(尤其在较大时间间

距的情况下),因此图像拼贴技术更适于应用在本系统中。

目前图像拼贴技术主要有基于图像分割的最大流算法和梯度域技术等方法。其中梯度域技术具有较快的处理速度,但拼贴效果较差;而最大流算法有非常好的拼贴效果,但处理速度较慢[3]。

2004 年 Vladimir Kolmogorov[4]提出了一种快速的最大流算法,该快速算法与其他若干最大流算法在解决图像分割问题时的处理时间由表 1 列出。其中,DINIC 为 Dinic 算法,H_PRF 为高度优先的预流压入算法,Q_PRF 为顺序优先的预流压入算法,V.K. 为快速的最大流算法。

表 1 几种最大流算法的处理时间 s

算法	图像尺寸/像素					
	35*35	50*50	70*70	100*100	141 * 141	200 * 200
DINIC	0.39	0.77	3.42	4.19	13.85	43.00
H_PRF	0.17	0.34	1.16	1.68	4.69	12.97
Q_PRF	0.16	0.35	1.24	1.70	5.14	14.09
V.K.	0.16	0.20	0.71	0.74	2.21	4.49

虽然 Kolmogorov 的快速算法在处理速度方面已经有了很大提高,但对于处理性能较低的移动设备或嵌入式设备,其处理速度仍然不能令人满意。因此,本文提出了一种基于最短路径的快速算法,该算法与最大流具有完全相同的拼贴效果,而算法的时间复杂度有显著的降低。

直观地,如果分割线两侧像素的颜色值相差过大,便会在视觉上形成比较明显的接缝;反之,如果分割线两侧像素的颜色值相差较小,其形成的接缝则不易被察觉。这里引入匹配误差函数 $M(s,t,\boldsymbol{A},\boldsymbol{B})$来描述在图像 $\boldsymbol{A},\boldsymbol{B}$ 的重叠区域内相邻的两个像素 s,t 之间颜色值的差异[5]

$$M(s,t,\boldsymbol{A},\boldsymbol{B}) = \|\boldsymbol{A}(s)-\boldsymbol{B}(s)\| + \|\boldsymbol{A}(t)-\boldsymbol{B}(t)\| \quad (7)$$

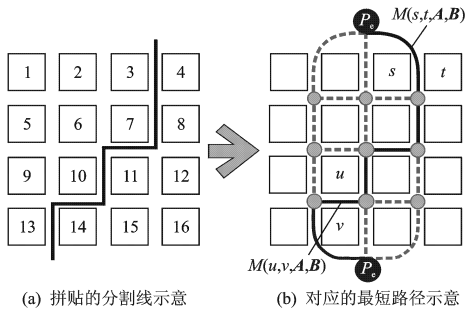
式中:$\boldsymbol{A}(s)$和 $\boldsymbol{B}(s)$分别为像素 s 在左图 $\boldsymbol{A}$ 和右图 $\boldsymbol{B}$ 中的颜色;$\boldsymbol{A}(t)$和 $\boldsymbol{B}(t)$则分别为像素 t 在左图 $\boldsymbol{A}$ 和右图 $\boldsymbol{B}$ 中的颜色。

为了选择出拼贴效果最好的分割线,引入拼贴误差函数 CE($\boldsymbol{C}$),其定义为

$$\mathrm{CE}(\boldsymbol{C}) = \sum_{c\in\boldsymbol{C}} M(s_c,t_c,\boldsymbol{A},\boldsymbol{B}) \quad (8)$$

式中:$\boldsymbol{C}$ 为与一个拼贴相对应的分割线;s_c 与 t_c 代表分割线上某一线元 c 两侧的像素。即一个拼贴的拼贴误差函数值等于这个拼贴对应的分割线上所有线元两侧的像素的匹配误差之和。本文要寻找的最佳拼贴就是使拼贴误差函数取得最小值所对应的拼贴方案。

图 2(a)为一个大小为 4×4 的重叠区域的示意图,其中实线为一个拼贴对应的分割线。在重叠区域内的每 4 个相邻像素之间加入一个节点,并将相邻的节点相连;再在区域外部加入 P_s 与 P_e 两个节点,并将 P_s,P_e 分别与重叠区域内的第 1 行与最后一行的节点相连。其相连边的权值取为其相邻的两个像素所对应的匹配误差函数值,如图 2(b)所示。这样,一个拼贴所对应的分割线便是从 P_s 到 P_e 的一条无环路径,而使拼贴误差函数 CE 取得最小值的分割线便是从 P_s 到 P_e 的最短路径。



(a) 拼贴的分割线示意 (b) 对应的最短路径示意

图 2 4×4 的重叠区域示意图

至此,图像拼贴问题转化为图论中的最短路径问题,利用基于最小堆的 Dijkstra 算法便能高效地完成最短路径的求解。基于最小堆的 Dijkstra 算法的时间复杂度为 $O(N\times\log_2(N))$,最大流算法的平均时间复杂度为 $O(N^3)$或 $O(NE)$[6],其中 N 为节点数,E 为边数。相比之下,本文算法的时间复杂度显著地降低了。

2 系统在 OMAP3530 平台上的实现

2.1 OMAP3530 简介

美国德州仪器(TI)公司新近推出了采用 DSP 和 ARM 双核架构的 OMAP3530 平台。OMAP3530 采用先进的 CortexA8 结构,可高速运行,具有 TI 最新的 DSP64x + 内核作为强劲的辅助图像处理,还包含了优化的视频硬件加速器和 2D/3D 图像加速器,另外还在同一个封装下提供了许多通用型硬件接口[7]。其在 DSP 端用 DSP/

BIOS 来支持音/视频算法的运行,在 ARM 端用 Linux 来支持其对外设的管理。对于 ARM 和 DSP 之间的数据交互,则用 Codec Engine 和 Codec Server 来加以管理。由于 OMAP 先进独特的结构,加之芯片运算处理能力强、功耗低,在移动通信和多媒体信号处理方面具有明显优势。

ICETEK-OMAP3530-EVM 是一款智能系统板,具有较为完备的通用硬件接口,能连接市场上通用的计算机设备,可用于软硬件系统评估、嵌入式应用教学、软硬件开发等工作;软件方面,支持以开源为宗旨的 Linux 操作系统。这款产品具备开源、先进的软硬件设计,目的是提供给用户一个高性能的评估平台,用来评估以 OMAP3x 系列器件为核心的嵌入式、便携式或手持式系统能达到的性能和指标。

2.2 系统设计框架

为了将上述的关键帧提取算法和图像拼贴算法应用在视频预览中,本文需要实现一个具有预览、播放以及可视化用户界面等功能的视频播放系统。这里按照面向对象的程序设计思想对系统进行了分析、划分、抽象,最终得到系统的模块框图如图 3 所示,各模块功能简述如下。

(1)导航器主模块

该模块是整个系统的核心模块。首先,其负责显示导航所需的画卷、管理关键帧的提取模块和图像拼贴模块的运作;其次,其接收来自控制模块的控制信号和来自操作系统的鼠标事件,并做出相应的动作;最后,其负责与视频播放模块进行视频定位与同步的通信。

(2)关键帧提取模块

该模块负责从关键帧信息文件(该文件可以由

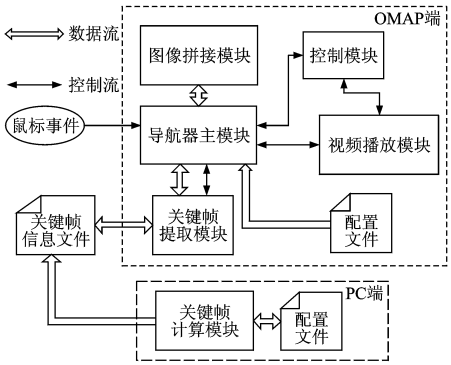


图 3 系统的模块框图

关键帧计算模块自动生成或由人工生成)中读取关键帧信息,并响应来自导航器主模块的控制信号,将关键帧的图像数据发送给导航器主模块。该模块屏蔽了导航器主模块与关键帧信息文件之间的实现细节,为日后的升级与扩展提供了良好的框架。

(3)图像拼贴模块

该模块负责完成图像拼贴任务。由于拼贴算法需要消耗一定的运算资源,因此利用 TI 公司提供的 Codec Engine 工具[8]将其封装成能够在 DSP 端执行的算法包,在需要时由 ARM 端调用、在 DSP 端执行,充分发挥了 OMAP 的双核优势。

(4)视频播放模块

该模块用于完成视频文件的播放任务。其不仅具备播放、暂停、停止等基本的视频播放功能,还可以根据导航器主模块发来的定位消息将播放跳转至相应的位置。

(5)关键帧计算模块

考虑到关键帧提取算法需要大量的计算并且对于同一个视频不需要反复进行关键帧提取等原因,本文只在 PC 端实现了这个模块的功能,然后将选取的关键帧信息保存在关键帧信息文件中,以便视频播放系统使用。

(6)配置文件

该模块负责制定程序运行时所需的信息(例如选择播放文件、修改屏幕尺寸等)。使用配置文件可以在不增加太多代码量的情况下极大地增加程序的灵活性,是系统开发初期常见的手段。

由上述各模块的概述可以看出,该视频播放系统的各模块间通过抽象的接口进行通信,屏蔽了各自的实现细节,降低了各模块间的耦合度,为以后

的升级、改造、移植提供了良好的基础。

3 实验结果与分析

3.1 拼贴效率的比较

在第 1 节中提到的匹配误差函数以及拼贴误差函数与基于最大流的拼贴算法中相应的函数是完全一致的,因此这两种算法具有相同的拼贴结果。严格来说,是最大流算法所求的最大流值与本文算法所求的最短路径长度值相同,若最短路径惟一,那么这两种算法的拼贴结果就是完全一致的。

在 CPU 为 Intel Q8400,4 GB 内存的 PC 机上,针对两幅尺寸都为 512×288、重叠宽度为 180 的图像,分别利用 Kolmogorov 的快速最大流算法(这里采用了 Kolmogorov 在网上公布的 C++源代码)和本文提出的算法进行若干次图像拼贴。处理结果表明,两种算法的拼贴效果完全相同,只是处理时间有所差异,两种算法的处理时间如表 2 所示。由表 2 中的数据可以看出,本文所述算法在与最大流算法具有相同拼贴结果的情况下减少了近三分之二的运行时间,其处理效率有明显的提高。

表 2 两种算法的处理时间 s

算法	拼贴图像数量		
	100	500	1 000
最小割算法	15	71	140
本文算法	5	24	49

3.2 拼贴效果的展示

在 PC 机上实现基于最短路径的图像拼贴的算法,图 4 为两个图像拼贴效果的示意图,其中用白线显著地标出了拼贴间的分割线。由图中可以看出该算法较好地实现了图像拼贴的功能。

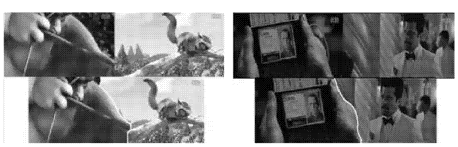


图 4 图像拼贴效果示意图

3.3 系统实现结果

为了检验拼贴算法在 OMAP3530 上的处理速度,这里分别对图片尺寸为 64×36、重叠宽度为 22,图像尺寸为 128×72、重叠宽度为 45,以及图像尺寸为 256×144、重叠宽度为 90 的 3 组图像进行拼贴,得到的图像拼贴算法处理时间数据见表 3。由此可以看出本文的图像拼贴算法具有较高的效率。

表 3 图像拼贴算法的处理时间

测试内容	图像尺寸/像素		
	64 * 64 重叠宽度 22	128 * 72 重叠宽度 45	256 * 144 重叠宽度 90
拼贴图像张数	2 000	2 000	2 000
总拼贴时间/s	7	39	179
平均拼接时间/s	0.003 5	0.019 5	0.089 5

为了向用户提供更多的视频信息,系统采用了多尺度的预览方式,其效果示意图如图 5 所示。该图显示了同一视频中的同一段内容在相邻的 3 个预览尺度下得到的预览效果,图中虚线指出不同尺度中的同一段视频内容。

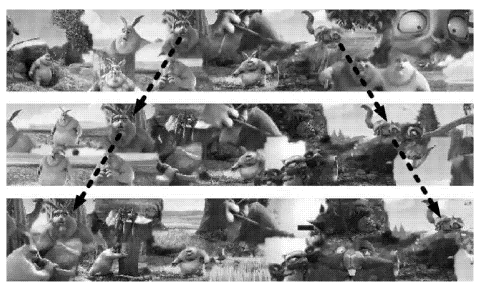


图 5 多尺度视频预览效果示意图

4 结束语

本文提出了一种快速的图像拼贴算法,并将其与关键帧提取等算法相结合,在 OMAP3530-EVM 平台上实现了具有画卷形式的视频预览功能的视频播放系统。与传统的进度条或章节预览等形式相比,这种形式的预览具有更加直观、更加高效、更加美观等特点。另外,该系统还具有良好的模块化结构,各模块之间用抽象的接口进行通信。这样就便于在以后对系统进行升级、改造,增加其所适用的领域。

参考文献:

[1] 彭启琮,杨鍊,潘晔. 开放式多媒体应用平台——OMAP 处理器原理及应用[M]. 北京:电子工业出版社,2005.
Peng Qicong, Yang Lian, Pan Ye. Open multimedia application platform—OMAP processors priciples and applications[M]. Beijing: Publishing House of Elec-

tronics Industry, 2005.

[2] 高新波.模糊聚类分析及其应用[M].西安:西安电子科技大学出版社,2004.
Gao Xinbo, Fuzzy clustering analysis and applications [M]. Xi'an: Xidian University Publishing House, 2004.

[3] Connelly B, Dan B G, Eli S, et al. Video tapestries with continuous temporal zoom[J]. ACM Transactions on Graphics, 2010,29:489.

[4] Vladimir K. Graph based algorithms for scene reconstruction from two or more views[D]. Ithaca, NY, USA: The Graduate School of Cornell University, 2004.

[5] Vivek K, Arno S, Irfan E, et al. Graphcut textures: Image and video synthesis using graph cuts[J]. ACM Transactions on Graphics, 2003,22(3):227-286.

[6] Yuri B, Vladimir K. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004,26(9):1124-1137.

[7] Texas Instruments. OMAP35x applications processors [EB/OL]. [2009-10-15]. http://www.ti.com.

[8] Texas Instruments. Codec engine application developer user's guide[EB/OL]. [2011-04-17]. http://Processors.wiki.ti.com.

作者简介:乔瑞萍(1966-),女,副教授,研究方向:图像与图形处理、音视频编解码实时处理,E-mail:rpqiao@mail.xjtu.edu.cn;胡宇平(1987-),男,硕士研究生,研究方向:图像与图形处理、音视频编解码实时处理;王效鹏(1989-),男,硕士研究生,研究方向:图像与图形处理、音视频编解码实时处理;俞璐(1987-),女,硕士研究生,研究方向:图像与图形处理、音视频编解码实时处理。