# 基于GPU实时视频处理的多投影融合系统研究

李晓光 刘宏哲 袁家政
(北京联合大学北京市信息服务工程重点实验室 北京 100101)

**摘 要** 介绍了GPU高速并行运算及其对数字图像、视频处理的重要作用。针对多通道环幕投影系统,采用CPU与GPU组合的异构计算结构,提出了一种视频实时处理方案。该方案通过DirectShow的链路模型保证了视频处理的灵活性,设计并采用可用于并行运算的几何校正、边缘融合算法,提升了视频处理的高效性。这一构架可以用于单通道4k格式视频的高质量效果展示,同时能有效降低构建成本,提高系统的经济实用性。

**关键词** 并行运算,多通道投影,实时视频处理,DirectShow,几何校正,边缘融合
**中图法分类号** TP319.9 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.9.056

## Multi-projector Displays System Research Based on GPU Real-time Video Processing

LI Xiao-guang LIU Hong-zhe YUAN Jia-zheng
(Beijing Key Laboratory of Information Service Engineering, Beijing Union University, Beijing 100101, China)

**Abstract** This paper introduced GPU high-speed parallel computing, which plays an important role in digital image and video processing. For multi-channel surround screen projection system, we used a combination of CPU and GPU heterogeneous computing structure, and proposed a real-time video processing solution. By using link model of DirectShow, the program ensures the flexibility of video processing. Geometric correction and edge blending algorithm are designed for parallel computing to enhance the efficiency of video processing. This framework can be used for single-channel, high-quality 4k format video display, effectively reduces building costs, and improves economic practicality of the system.

**Keywords** Parallel computing, Multi-channel projection, Real-time video processing, DirectShow, Geometric correction, Edge blending

## 1 引言

多通道投影系统[1]作为一种虚拟现实系统,利用多台投影机投射出无缝统一、高图像分辨率的完美影像和宽视角视觉场景,配合环绕立体音响系统,打造出一种沉浸式的、身临其境般的虚拟仿真环境。其产品广泛应用于交通指挥监控、虚拟战场仿真、数字城市规划、博物馆、电影院、教育培训、旅游景区等大型场景仿真环境。多通道环幕投影系统的构建成本远低于专业图形工作站的,且灵活易扩展。

GPU(Graphic Processing Unit)是显卡的计算核心,GPU计算性能的提升和GPU辅助运算技术的实现,使得GPU适合处理并行的科学计算。在图形图像处理中,使用GPU进行通用计算已成为一种发展趋势,卢风顺等[2]深入阐述了CPU、GPU协同运算的研究进展。Microsoft DirectShow应用程序界面是微软Windows平台的一种媒体流系,使用Filter Graph模型来管理整个数据流的处理过程,广泛地支持各种媒体格式,提供高品质的多媒体流捕捉和回放。利用DirectShow编程,可以很方便地对视频数据进行处理。

近年来出现了很多几何校正和边缘融合算法,例如,Bhasker等[3]针对投影镜头的畸变给出了基于有理Bezier曲面网格的修正方法。Okatani等[4]提出了显示墙与相机及投影仪对应的透视几何关系的自动几何校正方法。王修晖等[5]提出了投影仪的广义颜色模型和基于搜索技术的视觉无缝方法,使用数码相机实现了高精度的视觉无缝画面校正。曾鸿等[6]使用贝赛尔曲线重构变形屏幕边界,实现了多投影画面的快速几何校正。大多数视频源的处理需要先对各通道的视频分别进行预处理,然后将处理后的视频直接进行播放,谢逸群等[7]提出了一种基于P-F模型的视频实时处理方法,其不需要对视频进行预处理操作,通过P-F模型就能很好地在普通PC上实现视频的实时播放处理。

视频的实时播放处理要求采用高效的算法和较低分辨率的视频文件,以保证视频播放的流畅性,采用多线程处理方式可以充分利用CPU处理能力,但是会影响系统的稳定性和实时响应能力。本文借助于DirectShow的链路模型,将视频链

到稿日期:2014-04-13 返修日期:2014-06-28 本文受国家自然科学基金(61372148,61271369,41101111),北京市教育委员会科技发展计划面上项目(SQKM201411417004),北京联合大学人才强校计划人才资助项目(BPHR2014A04,BPHR2014E02),北京市属高等学校创新团队建设与教师职业发展计划项目(CIT&TCD20130513,IDHT20140508)资助。

李晓光(1988—),男,硕士生,主要研究方向为数字图像处理、虚拟现实,E-mail:469589940@qq.com;刘宏哲(1971—),女,博士,副教授,主要研究方向为语义计算、数字图像处理、分布式系统、人工智能、数字博物馆等,E-mail:liuhongzhe@tuu.edu.cn(通信作者);袁家政(1971—),男,博士,教授,主要研究方向为图形图像处理、文物遗迹的数字化处理、数字博物馆等。

路分解成若干个模块，每个模块完成特定的功能，设计并使用非线性几何校正和边缘融合的并行算法，结合 GPU 辅助运算，通过模块间的协同工作，实现对视频画面的实时处理。

## 2 基于 GPU 并行运算的链路模型视频处理

### 2.1 DirectShow 的链路模型

DirectShow 设计灵活、应用广泛，胡耀华等[8]基于 DirectShow 的视频直播系统的研究与实现，构建了网络视频系统。DirectShow 技术中最重要的一个概念就是过滤器，过滤器通常在多媒体中执行一个操作。过滤器大致分为 3 类：源过滤器、转换过滤器和渲染过滤器。

源过滤器主要负责数据获取，数据源既可以是文件，也可以是采集卡采集到的数据，对其然后把数据向下传；转换过滤器首先接收其它过滤器传来的数据，加工处理后传递给后面的过滤器；渲染过滤器负责把数据传递给显卡、声卡进行多媒体演示，或输出到文件进行存储。

根据视频数据处理要求，在链路上添加若干个转换过滤器，每个转换过滤器执行特定功能，如图 1 所示。

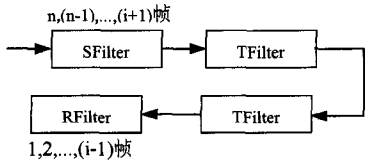


图 1 过滤器链路

假定链路中有 $N$ 个功能模块，耗费的时间分别为 $T(1)$，$T(2)$，…，$T(N)$，延迟计算方法如式(1)所示。

$$\begin{aligned} &\text{帧间延迟：} T_{FDT}=\max(T(1),T(2),\cdots,T(N)) \\ &\text{缓冲延迟：} T_{CDT}=\sum_{i=1}^{N}T(i) \end{aligned} \tag{1}$$

链路工作时，帧间延时 $T_{FDT}$ 为链路中处理速度最慢的 Filter 耗费的时间。缓冲延时 $T_{CDT}$ 反映了视频播放前的缓冲时间，是所有 Filter 处理时间的总和。$T_{FDT}$ 值超过某个限度，视频播放就会出现卡帧现象，可以通过拆分速度慢的 Filter 来降低 $T_{FDT}$ 值，同时也引起 $T_{CDT}$ 值的增加，当 $T_{CDT}$ 值超过它的阈值后，会导致视频播放不同步，并可能会因超过 CPU 的最大处理能力而产生视频卡帧。为了保证视频播放流畅，必须同时兼顾 $T_{FDT}$ 和 $T_{CDT}$，并将其控制在 CPU 的运算能力之内，且在处理高分辨率视频文件时，对算法的效率要求较高。

### 2.2 基于 CPU/GPU 协同运算的功能模块

在多通道环幕投影系统中，CPU 与 GPU 组合的异构计算结构将充分发挥计算机的运算能力。刘凡美、周艳霞等[9,10]均利用 GPU 加速，使提出的图像融合新算法和非线性几何校正算法执行速率明显提升。

CPU/GPU 协同并行计算[2]的关键在于如何实现两者的高效“协同”。其主要分为两个层次：(1)CPU 仅负责管理 GPU 的工作，为 GPU 提供数据并接收 GPU 传回的数据，由 GPU 承担整个计算任务；(2)除管理 GPU 外，CPU 还负责一部分计算任务，与 GPU 共同完成计算。

本文采用的 CPU/GPU 协同并行计算采用第二种协同方式，借助于 DirectShow 的链路模型，通过设计转换过滤器，将特定功能的转换过滤器添加到 DirectShow 的链路模型中，如图 2 所示。每一帧的数据处理步骤如下：

(1)在源过滤器中完成视频源数据的解码和音频、视频的分流操作，视频部分经过解码转换为图像矩阵数据，视频矩阵数据通过链路模型传递到转换过滤器的缓冲池，准备进行下一步数据处理。

(2)针对缓冲池中的图像矩阵数据，按照次序进行处理。将图像矩阵按照算法进行 CPU/GPU 协同并行计算，完成运算后将数据送回缓冲池中。由链路模型负责将数据送入下一个链路模块，进行下一阶段的数据处理操作。

(3)每一个转换过滤器执行步骤(2)的操作，直到完成最后一个转换过滤器的数据操作，然后将视频矩阵数据通过链路模型传递到目标过滤器的缓冲池。

(4)目标过滤器负责将数据进行渲染操作，同时通过视频源的参考时钟对音频和视频信息进行同步操作。

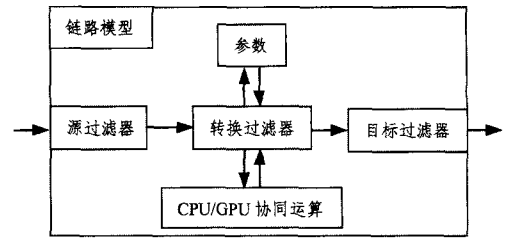


图 2 链路模型

## 3 算法设计与实现

### 3.1 边缘融合算法的设计与实现

边缘融合的目的是调节视频缓存中融合区域图像的亮度，使融合后的图像亮度与正常图像一致。王胜正等[11]采用如式(2)所示的幂函数作为融合函数。它将右通道融合区域的像素乘以 $f(x)$，左通道融合区域的像素乘以 $1-f(x)$。其中 $x$ 为像素在融合带的位置，$x=1$ 表示融合带的左边缘，$x=0$ 表示融合带的右边缘，两个通道重叠像素值之和为 1，即为原始画面的像素值。$a$ 为亮度调节系数，影响融合带中心位置的亮度，$a\in[0,1]$，当 $a>0.5$ 时，混合区域中心变亮；当 $a<0.5$ 时，混合区域中心变暗。$p$ 为渐变指数，控制曲线的弯曲程度。

$$f(x)=\begin{cases} a(2x)^p, & 0\leqslant x<0.5 \\ 1-(1-a)[2(1-x)]^p, & 0.5\leqslant x\leqslant 1 \end{cases} \tag{2}$$

本文通过添加一个专门的视频处理模块，完成边缘融合运算。首先根据需要融合图像的大小，利用融合算法设计可用于并行运算的融合矩阵，再利用 GPU 参与并行运算，达到高质量的融合效果。处理流程如图 3 所示。

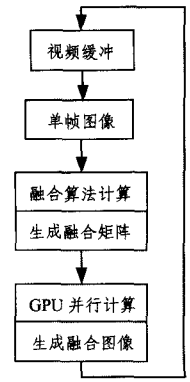


图 3 边缘融合算法处理流程

边缘融合算法具体步骤如下：

(1)在当前边缘融合链路的视频缓冲池中读取视频帧缓存(视频缓存默认最下面一行为初始行)，读取并记录当前图像矩阵 $MatA$ 的窗口大小、高度 $H$、宽度 $W$。

$$MatA=\begin{bmatrix} a_{30} & a_{31} & a_{32} & a_{33} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{00} & a_{01} & a_{02} & a_{03} \end{bmatrix}$$

(2)初始化边缘融合函数参数，式(2)中参数 $a$、$P$ 的默认取值分别为 0.25、2。

(3)根据步骤(1)记录的窗口大小，设计矩阵 $MatB$ 和融合带宽度 $Bw$，保证矩阵 $MatB$ 和当前图像矩阵保持一致，根据式(3)计算融合矩阵，其中 $j\in[0,H]$。

$$MatB[i,j]=\begin{cases} f(j/BW), & j\leqslant Bw \\ f((W-j)/BW), & j\geqslant W-Bw \\ 1, & Bw<j<W-Bw \end{cases} \tag{3}$$

(4)对于矩阵 MatA 的每一个元素，$i\in[0,W]$，$j\in[0,H]$，根据在步骤(3)中计算出的融合矩阵 $MatB$，将两个矩阵对应元素相乘的结果赋予矩阵 $MatA$，用式(4)进行运算。每一个元素的运算都与其他元素的运算无关，满足并行运算要求，利用 GPU 并行运算资源，将矩阵 $MatA$ 和 $MatB$ 进行并行运算。

$$\begin{aligned} &PV=i+j*W \\ &MatA[PV]=MatA[PV]*MatB[PV] \end{aligned} \tag{4}$$

(5)将处理完后的图像矩阵 $MatA$ 送回视频缓冲池相应的位置。

(6)每次参数调整后，会重新生成融合矩阵，将缓冲池中剩余的图像矩阵依次执行步骤(4)、步骤(5)，直到处理完整个视频或再次进行了参数调整。

### 3.2 几何校正算法的设计与实现

在多通道环幕投影系统中，投影仪需要架在高处向下投影，投影画面产生形变，即非线性几何失真[12]，如图 4 所示。

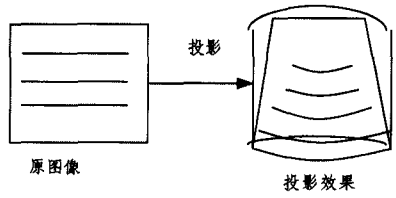


图 4 原始投影效果

使用投影仪自带的光学校正功能，对失真图像进行有限的梯形校正，但是弧面失真需要通过对视频缓存进行几何校正。本文通过添加一个专门的视频处理模块来进行非线性几何失真校正，使得投影屏幕上的图像显示正常，如图 5 所示。

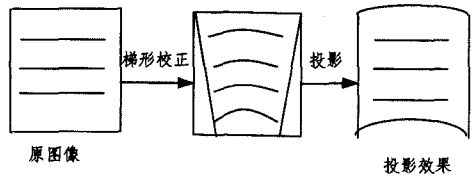


图 5 校正后投影效果

王胜正等[11]使用基于 Bezier 参数的几何变形曲面对投影图像进行非线性几何校正。三次方 Bezier 曲线的参数主要包括起始点 $p_0$、终点 $p_3$、两个曲线控制点 $p_1$ 和 $p_2$ 以及步长 $h$，控制点 $p_1$、$p_2$ 控制曲线的弯曲程度，步长 $h$ 控制图像的光滑程度，步长 $h$ 越小，图像越光滑，如式(5)所示。

$$\begin{aligned} &B(h)=p_0(1-h)^3+3p_1h(1-h)^2+3p_2h^2(1-h)+p_3h^3 \\ &h\in[0,1] \end{aligned} \tag{5}$$

本文采用基于 Bezier 参数的三次曲面并行计算方法，可根据需要实时调整控制点 $p_1$、$p_2$，以便调整曲线的弯曲弧度，使得非线性几何校正更加灵活。首先根据视频缓冲池中图像矩阵的大小，初始化 Bezier 曲线参数，然后设计矩阵 $MatB$，要求 $MatB$ 与视频缓冲池中的单帧图像大小一致，将矩阵的每一行拟合贝赛尔曲线，生成可用于并行运算的 Bezier 参数矩阵，处理流程如图 6 所示。

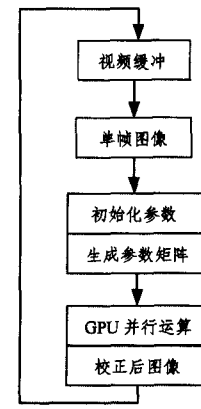


图 6 几何校正算法处理流程

该设计方案能有效提高系统整体运算效率，能保证流畅播放高清视频。几何校正算法具体步骤如下：

(1)在当前链路的视频缓冲池中读取视频帧缓存(视频缓存默认最下面一行为初始行)，读取并记录当前图像矩阵 $MatA$ 的窗口大小、高度 $H$、宽度 $W$。

$$MatA=\begin{bmatrix} a_{30} & a_{31} & a_{32} & a_{33} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{00} & a_{01} & a_{02} & a_{03} \end{bmatrix}$$

(2)根据步骤(1)记录的窗口大小，设计矩阵 $MatB$ 和 $MatC$，与当前图像矩阵保持一致，初始化 Bezier 曲线参数，生成一条如式(5)所示的 Bezier 三次曲线。

(3)对于矩阵 $MatB$ 的每一行，拟合步骤(2)生成的 Bezier 参数曲线，且要求按照矩阵从下边缘到上边缘的次序，依次调整 Bezier 曲线的控制点 $p_1$ 和 $p_2$，使得曲线不断接近直线。式(6)中，$Vertical$、$Vertical\_R$ 为 $p_1$、$p_2$ 初始化时的纵坐标分量，表示 $p_1$、$p_2$ 点到由 $p_0$、$p_3$ 连成直线的垂直距离，$H$ 表示图像矩阵的高度，$i$ 表示矩阵的行索引。运算后，生成可用于并行运算的 Bezier 参数矩阵 $MatB$，如式(7)所示，其中 $B_{ij}=B(h_j,p_1^i,p_2^i)$，$j$ 表示矩阵的列索引。

$$\begin{aligned} &p_1^i=Vertical-Vertical/H*i \\ &p_2^i=Vertical\_R-Vertical\_R/H*i \end{aligned} \tag{6}$$

$$MatB=\begin{bmatrix} B_{30} & B_{31} & B_{32} & B_{33} \\ B_{20} & B_{21} & B_{22} & B_{23} \\ B_{10} & B_{11} & B_{12} & B_{13} \\ B_{00} & B_{01} & B_{02} & B_{03} \end{bmatrix} \tag{7}$$

(4)对于矩阵 $MatA$ 的每一个元素，用式(8)进行运算，其

中 $i\in[0,W]$,$j\in[0,H]$。每一个元素的运算都与其他元素的运算无关,满足并行运算要求,利用 GPU 并行运算资源,将矩阵 *MatA* 和 *MatB* 进行并行运算。

$$PV=i+j*W$$
$$T=PV-MatB[PV]*W \tag{8}$$
$$MatC[PV]=\begin{cases}MatA[T], & T>D\\ 0, & \text{其他}\end{cases}$$

(5)将处理完后的图像矩阵 *MatC* 送回视频缓冲池相应的位置。

(6)每次参数调整后,会重新生成几何校正的参数矩阵,将缓冲池中剩余的图像矩阵依次执行步骤(4)、步骤(5),直到处理完整个视频或再次进行了参数调整。

## 4 系统设计与结构分析

实验系统中 PC 机配置为 I7 2180GHz CPU,4GB 内存,图形加速卡为 NVIDIA GTX770,4GB 显存。投影系统由 3 台分辨率为 1024 * 768 的 Panasonic 投影仪和 3 台普通的计算机组成。3 台投影仪根据环幕特点摆放,确保相邻投影仪的投影有一定的重叠区域,且覆盖整个环幕。系统采用基于局域网的客户端-服务器模型,服务器控制端应用程序可通过局域网向各用户端发送控制指令,进行多投影显示的参数设置。

### 4.1 系统设计流程

链路模型中,包括 DirectShow 系统自带的源过滤器、解码过滤器、渲染过滤器,它们分别完成视频文件的采集、音视频解码分流、时钟同步渲染操作。插入的转换过滤器包括裁剪过滤器、非线性几何校正过滤器、边缘亮度融合过滤器,以完成显示区域的调整、环幕几何校正、场景拼接的亮度融合操作。系统流程如图 7 所示。

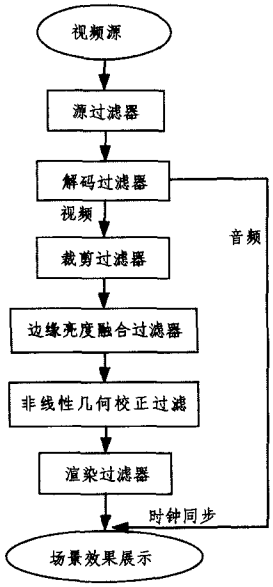


图 7 系统流程

### 4.2 实验结果分析

引入衡量 Filter 优劣的两个时间指标即 $T_{FDT}$ 和 $T_{CDT}$ 分别进行了实验[10],实验采用 4k 格式视频源、分辨率为 3840 * 2160、帧速率为 29.970fps、数据速率为 32.3mbps 的视频文件,测试算法为几何校正算法,测试时间单位为 ms,测试结果如表 1 所列。

表 1 几何校正算法测试结果

| 分辨率 | 1280 * 720 | 1920 * 1080 | 4096 * 2160 | 8192 * 4320 |
|---|---|---|---|---|
| $T_{trans}$ | 1.31 | 2.72 | 9.76 | 37.41 |
| $T_{gpu}$ | 0.59 | 0.88 | 2.75 | 9.23 |
| $T_{trans\_gpu}$ | 1.90 | 3.60 | 12.51 | 46.64 |
| $T_{cpu}$ | 5.01 | 14.61 | 47.48 | 189.47 |

非线性几何校正算法具有较高的时间复杂度。在进行算法测试的过程中,采用非线性几何校正算法进行测试,在使用 GPU 和不使用 GPU 的情况下分别记录算法的执行时间。在使用 GPU 资源计算时,首先应将图像数据从主存传输到显存,再利用 GPU 进行计算,因此,将传输时间 $T_{trans}$ 和 GPU 计算时间 $T_{gpu}$ 的和作为 GPU 计算所消耗的总时间 $T_{trans\_gpu}$,即 $T_{trans\_gpu}=T_{trans}+T_{gpu}$。将 CPU 计算消耗时间 $T_{cpu}$ 与 GPU 计算所消耗 $T_{trans\_gpu}$ 作对比,结果如图 8 所示。

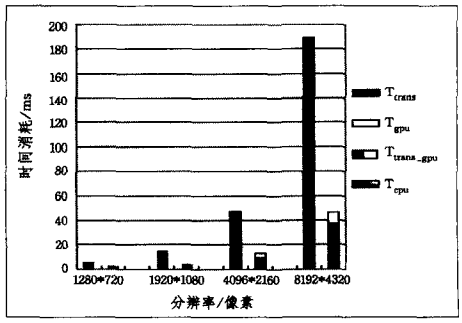


图 8 CPU、GPU 消耗时间对比

从图 8 中可以看出,图像数据从主存到显存的传输时间要远大于 GPU 的计算时间,且随着视频文件分辨率的提升,传输消耗时间呈现出超过指数阶的增长速度,当视频文件分辨率到达 8192 * 4320 时,仅传输消耗时间就不能保证视频的正常播放。

利用 CPU 计算资源,在分辨率为 4096 * 2160 时,算法处理一帧图像矩阵的时间为 47.48ms,视频文件不能正常播放;采用 CPU、GPU 协同编程的方法,处理时间压缩为 12.51ms,满足流畅播放要求。通过实验证明,系统具有较好的边缘拼接和几何校正效果,在单通道视频分辨率为 4096 * 2160 时,仍可以保证多投影系统正常运行。系统采用 CPU、GPU 协同编程,在 CPU 较差的 PC 机中利用 GPU 的计算资源,保证了算法在系统允许的时间范围内完成运算,减少对 CPU 计算资源的消耗,保证视频播放的流畅性,提高了计算机运行的稳定性。

系统效果展示如图 9 所示。

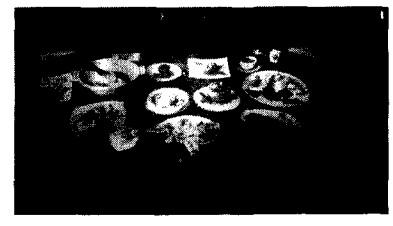

图 9 系统效果展示

**结束语** 本文通过设计并使用边缘融合和非线性几何校正并行算法,采用 CPU、GPU 协同运算的编程方式,可以有效提高算法的运行速度,减少 CPU 资源的消耗,有效降低对 CPU 计算性能的依赖。在提升硬件利用率和计算效率的同

(下转封三)

[15] Chao Yu-wei, Yeh Y R, Chen Yu-wen, et al. Locality-constrained group sparse representation for robust face recognition[C]// Proceedings of the 18th IEEE International Conference on Image Processing. 2011:761-764

[16] Yang Meng, Zhang Lei, Feng Xiang-chu, et al. Fisher discrimination dictionary learning for sparse representation[C]// Proceedings of the IEEE International Conference on Computer Vision. 2011:543-550

[17] Ma Long, Wang Chun-heng, Xiao Bai-hua, et al. Sparse representation for face recognition based on discriminative low-rank dictionary learning[C]//Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2012:2586-2593

[18] Wei Chia-po, Chao Yu-wei, Yeh Y R, et al. Locality-sensitive dictionary learning for sparse representation based classification [J]. Pattern Recognition, 2013, 46(5):1277-1287

[19] 詹永照,汪满容,柯佳. 基于人工免疫有序聚类的视频关键帧提取方法[J]. 江苏大学学报(自然科学版),2012,33(2):199-204
Zhan Yong-zhao, Wang Man-rong, Ke Jia. Video key-frame extraction using ordered samples clustering based on artificial immune[J]. Journal of Jiangsu University (Natural Science Edition), 2012, 33(2):199-204

[20] Wang Yi-ding, Yan Qing-yu, Li Ke-feng. Hand vein recognition based on multi-scale LBP and wavelet[C]// Proceedings of the 2011 International Conference on Wavelet Analysis and Pattern Recognition. 2011:214-218

(上接第 288 页)

时,也为优化算法性能提供更多的空间,使得最终投影显示效果更加自然、流畅。借助于 DirectShow 的链路模型,增强系统对播放环境的适应能力,可以灵活地修改或者添加 Filter 来完善系统功能,用户可以通过修改 Filter 的参数来调整投影面的非线性几何校正和边缘融合效果。系统的整体效果已经达到一些专业设备的水平,随着 PC 机处理速度和 GPU 计算能力的进一步提高,可以通过设计效果更好但复杂度较高的算法来提升系统整体效果。

但是本文也存在着一些不足之处有待进一步改进和优化。由于 CPU 中高速缓存相对较多,而 GPU 中的高速缓存相对较少,数据从主存到显存的传输时间比较长,消耗太多的时间,当视频单帧图片矩阵的数据量比较大时,仅数据的传输时间就导致了系统不能流畅地播放视频文件。可以考虑在系统中添加传输效率更高的显卡,来提高系统对超高清视频文件的处理能力。在设计亮度边缘融合算法时,没有考虑到相邻投影面之间的色差问题;采用非线性几何校正算法,在播放低分辨率文件时,在环幕上会出现锯齿,这些问题都有待进一步完善。

## 参 考 文 献

[1] Raskar R, Brown M S, Yang R, et al. Multi-projector displays using camera-based registration[C]// Proceedings of Visualization'99. IEEE, 1999:161-522

[2] 卢凤顺,宋君强,银福康,等. CPU/GPU 协同并行计算研究综述[J]. 计算机科学,2011,38(3):5-9
Lu Feng-shun, Song Jun-qiang, Yin Fu-kang, et al. Survey of CPU/GPU synergetic parallel computing[J]. Computer Science, 2011, 38(3):5-9

[3] Bhasker E, Juang R, Majumder A. Registration techniques for using imperfect and par tially calibrated devices in planar multi-projector displays[J]. IEEE Transactions on Visualization and Computer Graphics, 2007, 13(6):1368-1375

[4] Okatani T, Deguchi K. Autocalibration of a projector-camera system[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005, 27(12):1845-1855

[5] 王修晖,华炜,林海,等. 面向多投影显示墙的画面校正技术[J]. 软件学报,2007,18(11):2955-2964
Wang Xiu-hui, Hua Wei, Lin Hai, et al. Screen Calibration Techniquese for Multi-Projector Tiled Display Wall[J]. Journal of Software, 2007, 18(11):2955-2964

[6] 曾鸿,张均东,马理胜,等. 快速多投影画面几何校正与边缘融合方法[J]. 计算机工程与设计,2013,34(5):1846-1850
Zeng Hong, Zhang Jun-dong, Ma Li-sheng, et al. Fast multi-projection image geometric calibration and edge blending method [J]. Computer Engineering and Design, 2013, 34(5):1846-1850

[7] 谢逸群,王慧雅,许华虎. 基于视频实时处理的多投影曲面拼接系统的研究[J]. 中国图象图形学报,2009,14(2):286-291
Xie Yi-qun, Wang Hui-ya, Xu Hua-hu. Achieving Multi-projector Displays Stitching Based on the Real-time Video Processing [J]. Journal of Image and Graphics, 2009, 14(2):286-291

[8] 胡耀华. 基于 DirectShow 的视频直播系统的研究与实现[D]. 北京:北京交通大学,2008
Hu Yao-hua. Study and Implementation of Video Live System based on Directshow[D]. Beijing: Beijing Jiaotong University, 2008

[9] 刘凡美. 基于 GPU 加速的多投影融合新算法的实现[J]. 电子技术与软件工程,2013(19):204-206
Liu Fan-mei. New Mutli-projection Fusion Algorithm Based on GPU Acceleration[J]. The Application of Computer Technology, 2013(19):204-206

[10] 周艳霞,秦开怀,罗建利. 多投影仪自由立体显示的 GPU 几何及亮度校正技术[J]. 计算机辅助设计与图形学学报,2011,23(4):561-570
Zhou Yan-xia, Qin Kai-huai, Luo Jian-li. GPU-Based Geometric and Photometric Corrections for Multi-projector Autostereoscopic Display[J]. Journal of Computer-Aided Design & Computer Graphics, 2011, 23(4):561-570

[11] 王胜正,杨杰. 自动多投影仪非线性几何校正与图像边缘融合方法[J]. 上海交通大学学报,2008,42(4):574-578
Wang Sheng-zheng, Yang Jie. Auto-Nonlinear Geometry Calibration and Edge Blending of Multi-Projector Display System [J]. Journal of Shanghai Jiaotong University, 2008, 42(4):574-578

[12] 贾庆轩,徐卉,宋荆洲,等. 基于投影仪一相机系统的多投影颜色校正技术[J]. 系统仿真学报,2013,25(5):1005-1013
Jia Qing-xuan, Xu Hui, Song Jing-zhou, et al. Color Correction for Multi-projector Screen Based on Projector-camera System [J]. Journal of System Simulation, 2013, 25(5):1005-1013