

文章编号:1001-9081(2010)02-0555-05

## 基于 H. 264 的嵌入式网络视频服务器的设计与开发

周 强<sup>1</sup>, 费章君<sup>2</sup>, 王 强<sup>2</sup>, 杨仕友<sup>1</sup>

(1. 浙江大学 电气工程学院, 杭州 310027; 2. 南京南自信息技术有限公司, 南京 210012)

(zhouq231@163.com)

**摘 要:**基于 Freescale 公司的 IMX27 视频处理芯片, 设计、开发了一种网络视频服务器。服务器以裁剪的 Linux 为操作系统, 服务端和客户端的交互通信采用多线程和短连接的方式, 有效地节约了系统资源, 减小了对服务器 CPU 的占用率。为获取高清晰度的视频图像, 应用自适应反交错算法分别对静止、运动图像进行拼接和插值, 实现了反交错图像变换, 保证了数据的完整性, 提高了图像的清晰度。最后给出了所开发服务器的具体应用实例, 服务器的运行效果达到了预期的设计目标。

**关键词:**嵌入式 Linux 操作系统; H. 264; 反交错变换; 短连接; 视频传输

**中图分类号:** TP277; TP919.8 **文献标志码:** A

## Design and development of H. 264-based embedded network video server

ZHOU Qiang<sup>1</sup>, FEI Zhang-jun<sup>2</sup>, WANG Qiang<sup>2</sup>, YANG Shi-you<sup>1</sup>

(1. College of Electronic Engineering, Zhejiang University, Hangzhou Zhejiang 310027, China;

2. Nanjing Nanzi Information Technology Company Limited, Nanjing Jiangsu 210012, China)

**Abstract:** This paper developed a network video server based on Freescale video processing chip IMX27 running on the Linux operating environment, and short connection was used to complete the interaction between the server and clients, which effectively saved system resources and reduced the CPU utilization of the sever. Moreover, in order to obtain high quality video images, the server adopted an adaptive de-interlacing algorithm based on the use of splicing and interpolation of static and moving picture respectively to realize de-interlacing of the video image, therefore the integrity of the video data was guaranteed and the video clarity was improved. In the end, an application case of the server was given, which achieved the desired designed goal.

**Key words:** embedded Linux system; H. 264; de-interlacing transformation; short connection; video transmission

### 0 引言

当前,我国视频监控系统的发展已经进入到全数字化视频监控时代。经历了第一代基于模拟信号技术构建的模拟视频监控系统(Video-Cassette Recording, VCR),第二代半数字化的视频监控系统(DVR/NVR)的理论与技术的积累和沉淀,第三代全数字化视频监控系统已经呈现蓬勃发展的态势,在不久的将来将成为安全防范市场的主力军。同时,随着计算机网络技术<sup>[1]</sup>、视频图像处理技术、传输和存储技术的快速进步,视频监控系统正向着网络化、数字化、智能化方面发展。此外,安全防范市场对视频图像画面的清晰度和稳定性的要求越来越高。有鉴于此,本文设计、开发了一款高性能的嵌入式网络视频服务器。与现有的方法不同,采用丢弃每帧视频图像奇偶两场中的一场,保存另外一场或直接将两场进行简单的拼接完成去交错的方法,本文服务器采用自适应反交错算法<sup>[2]</sup>,对静止和运动图像分别运用拼接和插值实现反交错,既维护了视频数据的完整性,又保证了视频图像的清晰度。此外,本文服务器的服务端与客户端的通信采用短连接的方式,与现有长连接的通信方式相比,这种方法每次与客户端的通信只完成一项任务,完成后立即释放所占资源,服务端只给有需求的客户端分配系统资源,有效地减少了系统资源的消耗和 CPU 的占有时间,提高

了设备的稳定性和处理速度。经过 IMX27 芯片压缩编码后,视频流为 H. 264 的压缩码流,在客户端解码后播放。最后给出了本文开发服务器的应用实例。典型运行结果表明,监控图像的质量达到了预期目标。

### 1 硬件结构

为了获取 H. 264 的标准视频流,本文选取 Freescale 公司的 IMX27 芯片作为主处理核心芯片。该芯片为具有强大视频图像处理能力的高性能视频流压缩编码芯片。芯片中固化了压缩编码算法,支持 H. 264 和 MPEG4 两种视频标准。采集后的视频流输送到此芯片后直接转化为标准的 H. 264 或 MPEG4 格式的视频压缩码流。芯片采用 ASIC 编、解码器配合 ARM9 构架,采用 90 nm 工艺制造,可以达到 400 Mbps 的时钟速度,可同时处理 4 路图像/位流的编、解码,并支持码率控制。设备的硬件结构示于图 1。

由于 IMX27 芯片最多处理一路 D1 效果或两路 CIF 效果的视频图像,而 TVP5150 芯片却能同时采集两路模拟视频,所以本文选用 TVP5150 芯片作为视频采集芯片。该芯片为 Texas Instruments 公司的一款视频采集芯片,主要完成模拟视频信号的数字采样,并转化成 YUV 4:2:0 比例的数字视频信号。其前端输入支持 PAL、NTSC 和 SECAM 制式的视频,在

收稿日期:2009-08-01。

**作者简介:**周强(1985-),男,甘肃西和人,硕士研究生,主要研究方向:电力视频监控、视频图像处理; 杨仕友(1964-),男,辽宁朝阳人,教授,博士生导师,主要研究方向:电磁场分析和综合、电磁兼容分析、计算方法; 费章君(1971-),男,安徽宁国人,工程师,硕士,主要研究方向:视频监控; 王强(1979-),男,江苏南京人,工程师,主要研究方向:视频监控。

IMX27 控制信号下将外接的模拟视频信号采样后通过总线发送到 IMX27 芯片中进行压缩编码,然后通过网口传送出去,完成视频的编码压缩。同时设备中有一个 128 MB 的 NandFlash 和—64 MB 的 DDR SD RAM, RS485 接口用以实现与外接设备的通信,本设备支持外接 SD 卡或硬盘在本地进行录像、本地存储。

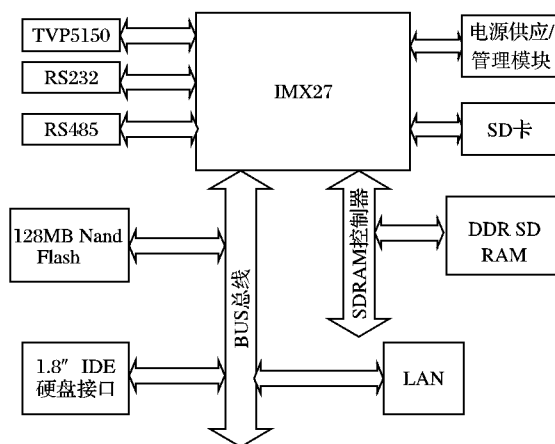


图1 视频服务器硬件结构

## 2 软件设计

由于嵌入式服务器资源非常有限,包括内存资源、CPU 处理速度等,因而软件设计的目的和思想就是在完成任务的同时尽可能地节约系统的资源,减少 CPU 的占用时间,提高系统的性能。有鉴于此,本文在服务器软件开发中采取了以下措施:第一,以裁剪后的 Linux 作为操作系统。一方面, Linux 系统为开源、免费的操作系统,性能稳定、可靠;另一方面,用户可以根据实际需要,对 Linux 操作系统进行相应的裁剪,在提高性能的同时可减少设备系统资源的消耗。第二,采用短连接的通信方式。服务端与客户端的每次通信只完成一项任务,任务结束后立即回收客户端所占的系统资源,以保证资源只分配给真正有需要的客户端。第三,服务端对客户端的管理采用多线程方式。服务端为每一个登录的客户端分配一个线程,独立地完成与此客户端的交互,真正实现了并行处理,一旦两者交互完成,立即关闭通信线程,回收资源。

软件的开发采用主机和目标板相结合的交叉开发模式。主机上运行的 Linux 操作系统为 Fedora 10,并安装了交叉编译器 arm-linux-gcc,在主机中编写、编译设备应用程序,最后将程序烧制到服务器中运行。

本服务器软件采用模块化的设计思想,确保了各个功能模块之间的独立性,保证了系统的稳定性和可扩展性。

### 2.1 服务器的软件结构

因为服务器中运行的 Linux 操作系统管理着整个设备的软件、硬件资源,应用程序运行在 Linux 操作系统上面,因而整个服务器的软件体系可分为四个层次:引导加载程序、Linux 内核、文件系统和视频服务器应用程序,其结构示于图2。

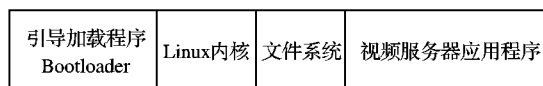


图2 设备的软件结构框图

Bootloader 为一段引导程序<sup>[3]</sup>,是系统上电后首先执行的代码程序,主要完成硬件初始化、设定软件环境并将操作系统内核复制到内存中。Bootloader 种类很多,有 red boot、uboot、

vivi 等,本文服务器选取 uboot,因为它是开源的,支持 ARM 体系,并且资料丰富。

Linux 内核<sup>[4]</sup>是 Linux 操作系统的核心,负责任务的管理和资源的分配。

文件系统是硬盘分区物理组织的用户级视图,该功能模块将 flash 等存储设备划分为若干分区,在不同的分区存放不同类别的文件。

服务器启动后,引导加载程序 Bootloader 首先运行,通过 Bootloader 将内核复制到内存中。内核启动后,挂载根文件系统,启动运行文件系统中的视频服务器应用程序,完成视频服务器的启动过程。

### 2.2 应用层软件结构的组成

设备应用层软件采用模块化的设计模式,整个结构如图3所示。

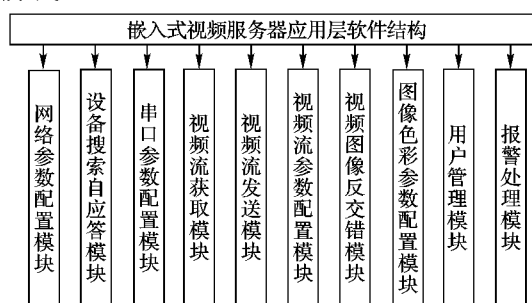


图3 服务器的应用层软件结构框图

下面对核心模块逐一进行分析和说明。

### 2.3 视频流的采集与编码

视频流的采集和编码模块是服务器最重要的核心模块之一,主要完成模拟视频流的采集以及将采集到的数据压缩编码为 H. 264 标准视频流数据,这个过程关系到视频图像的实时性和流畅性。

视频数据的采集是通过 TVP5150 芯片实现的。在服务器启动之后,系统通过激发 TVP5150 的驱动来完成视频数据的采集。在 Linux 操作系统下,V4L 为获取视音频数据的 API 接口,上层应用程序通过 V4L 接口调用 TVP5150 的驱动程序,实现对视频数据的采集。在 Linux 系统中,对所有的设备<sup>[5]</sup>的操作都是通过对设备文件的操作实现的。在本服务器中 TVP5150 设备对应的设备文件是“/dev/v4l/video0”,通过指令 open (“/dev/v4l/video0”, O\_RDWR, 0) 可启动 TVP5150。采集的视频数据流在 IMX27 芯片中经过压缩编码,完成视频流的采集和编码。

在本服务器中,为了保证视频传输的实时性,分别将视频数据的采集和视频数据的编码放到两个独立的线程中,实现采集和编码并行运行。这样可以避免单一线程中由于视频数据的采集比较耗时而造成的数据编码等候现象,提高视频数据的实时性。视频采集和编码部分的程序流程如图4所示。

在 V4L 中视频流的采集方式<sup>[6]</sup>有两种:一种是用 read() 函数直接读取;另外一种借助于 mmap() 函数采用内存映射方法截取视频。mmap() 内存映射方法使得进程之间通过映射同一个文件而实现内存共享。文件被映射到进程地址空间后,进程就可以像访问普通内存一样访问文件,并直接读写内存,不需要任何数据的拷贝,因而效率很高。为了保证视频图像的实时性,本文服务器采用第二种方式来获取视频流。

### 2.4 视频图像的反交错变换

由于通过摄像机采集到的视频图像是场交错的,即一帧

视频数据由两场组成:奇数场和偶数场。常用的方法是保存其中的一场,丢掉另外一场。但这样处理丢失了视频数据,图像的清晰度不高。此外,也有用拼接法将奇、偶两场直接拼接为一场。这种处理方法,对于静态图像效果很好;但对于运动图像,因为组成一帧视频图像的奇、偶场存在时间差,奇偶两幅图之间已经发生了变化,因此合成图像将出现锯齿纹、拖影等问题。有鉴于此,本文在服务器的设计中采用自适应去交错算法实现视频图像的反交错,对静止和运动图像分别应用不同的去交错算法。对于静止视频图像画面,直接将奇偶两场拼接为一场;对运动图像采用插值的方法去交错,整个反交错过程如图5所示。

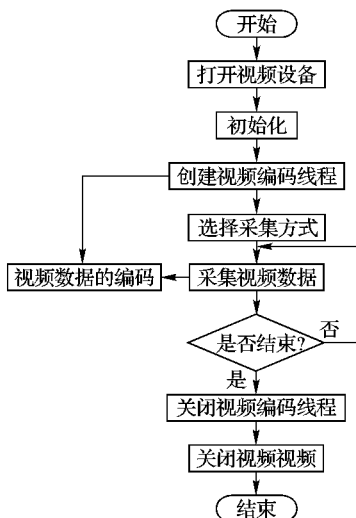


图4 视频流的采集、编码程序流程

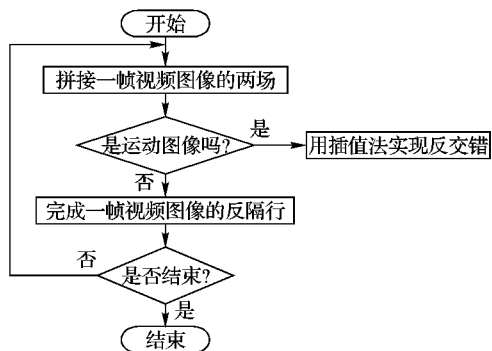


图5 视频图像的反交错流程

该过程分两步实现。

第一步,首先将一帧视频图像的两场拼为一帧图像,合并过程如图6所示,其中图中的每一点表示一个像素。

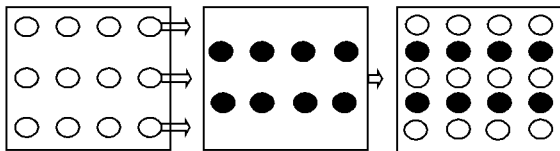


图6 一帧视频图像的奇偶场的拼接

第二步,判断是否为运动图像。若为运动图像,则用插值方法实现反交错;否则不做处理,保持第一步的结果。

运动图像的判断原理:一帧视频图像由奇、偶场组成,每一场都是单独采集的,两场的采集有一定的时间差,每一场都是一幅特定的画面。对于静止图像,由于场景没发生变化,奇偶场两幅图在邻近像素点的像素值不会发生突变,具有连续性,所以,可以将两场拼接为完整的一幅图像;但对于运动图

像,物体在奇场中位于某个位置,在偶场中可能已经运动到另外一个位置,两幅图间邻近像素发生了突变,所以可以根据邻近像素点的像素值是否发生突变判断采集的图像是否为运动图像。应用此原理,本文算法的具体实现过程如下:

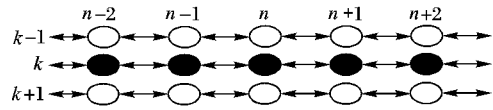


图7 拼接后一帧的区部图

假设  $S(k, n)$  为对应点  $(k, n)$  的像素值,  $X_n$  为点  $(k-1, n)$  与点  $(k, n)$  像素差的绝对值,  $Y_n$  为点  $(k-1, n)$  与点  $(k+1, n)$  像素差的绝对值(其他同理),并令:

$$\begin{cases} X_{n-1} = |S(k-1, n-1) - S(k, n)| \\ X_n = |S(k-1, n) - S(k, n)| \\ X_{n+1} = |S(k-1, n+1) - S(k, n+1)| \end{cases} \quad (1)$$

$$\begin{cases} Y_{n-2} = |S(k-1, n-2) - S(k+1, n-2)| \\ Y_{n-1} = |S(k-1, n-1) - S(k+1, n-1)| \\ Y_n = |S(k-1, n) - S(k+1, n)| \\ Y_{n+1} = |S(k-1, n+1) - S(k+1, n+1)| \\ Y_{n+2} = |S(k-1, n+2) - S(k+1, n+2)| \end{cases} \quad (2)$$

情形一 在式(1)中,若  $X_n > Y_n$ , 同时  $X_{n+1} > Y_{n+1}$  或  $X_{n-1} > Y_{n-1}$ , 说明组成一帧视频图像的奇偶场图像在点  $(k, n)$  附近像素值不再连续,发生了突变,可以判定这是一幅运动图像。因此不能简单地用两场拼接的方法合并两场。在此情况下通过给单场上下行插值的方法实现反隔行。实现的方法如图8所示。在插值点  $(k, n)$  的相邻两行的两侧对称地各取两个对应点对  $(k-1, n-2)$ 、 $(k+1, n-2)$ 、 $(k-1, n-1)$ 、 $(k+1, n-1)$ 、 $(k-1, n+1)$ 、 $(k+1, n+1)$ 、 $(k-1, n+2)$ 、 $(k+1, n+2)$ , 用包含点  $(k-1, n)$ 、 $(k+1, n)$  在内的这五对点的像素差最小的两点的像素平均值作为所要插入的点的像素值。对点  $(k, n)$  像素值的具体插值实现过程如下:

由式(2)求出5对点中像素差值最小的一对:

$$Y_i = \min(Y_{n-2}, Y_{n-1}, Y_n, Y_{n+1}, Y_{n+2}) \quad (3)$$

由此,点  $(k, i)$  处的像素值为:

$$S(k, n) = \frac{S(k-1, i) + S(k+1, i)}{2} \quad (4)$$

通过式(1)~(4)的运算,可对运动图像实现反交错。

情形二 若  $X_n < Y_n$  或  $X_n > Y_n$  且  $X_{n+1} < Y_{n+1}$ ,  $X_{n-1} < Y_{n-1}$ , 则说明视频图像为静态的视频图像,按第一步两场拼接法实现反交错。

## 2.5 视频流的获取和分发

本文服务器视频流的获取采用回调函数的形式实现。在系统中设置一个全局的视频流接收回调函数。当有压缩编码后的视频流产生时,系统会自动调用此回调函数将视频流写入到视频流缓冲区。因为这个过程由系统自动完成,因此效率高,实时性好。

当视频流接收回调函数在视频流接收缓冲区中写入视频流数据后,若有客户端登录到本视频服务器并发出调用视频流请求时,服务器就向此客户端发送视频流。服务器视频流的获取和分发的模式示于图8。

图8中,对视频流缓冲区的读、写都是在不同的线程中进行的。 $W$ 表示缓冲区视频流的写入位置, $R_i$ 表示第*i*个客户端在缓冲区中读取的位置。服务器视频流缓冲区的大小为

1 MB, 客户端从缓冲区中读取视频流有两种情况, 分别示于图9中的  $R_1$ 、 $R_2$ 。

图9中,  $W$  表示写入的位置, 缓冲区为循环写入模式, 即: 当写完缓冲区后会从初始位置重新写。  $R_1$ 、 $R_2$  分别表示客户端1和客户端2的读取位置, 缓冲区大小为  $BUFFER\_SIZE$ , 对客户端1下一次能读取的视频流的大小为  $W - R_1$ , 客户端2能读取的视频流大小为  $BUFFER\_SIZE - R_2 + W$ 。

对视频流的发送可以根据客户端的请求用 TCP、UDP、组播方式完成。

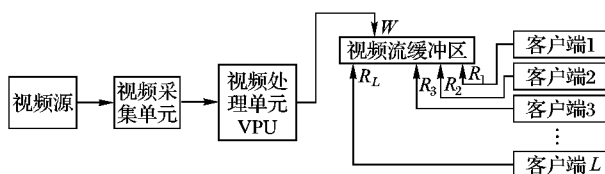
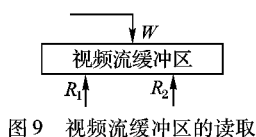


图8 视频流的接收和分发软件设计原理



### 3 设备客户端的管理

为了提高服务器系统资源的利用率, 本文服务器的并发服务端与客户端的交互采取两种有效的方式, 即: 第一, 服务端和客户端的通信用短连接方式; 第二, 服务端与每一个客户端的交互都在独立的线程中完成。

在主程序中 TCP 服务端一直处于监听状态。当有客户端连接时, 创建一个线程以处理此客户端的请求。这种多线程处理模式的优点在于可以可靠地实现客户端和服务端的通信, 避免在单线程中由于其中某一客户端通信发生故障而使整个程序阻塞卡死。所有此客户端与服务端的交互都在这个线程中完成。该部分处理过程<sup>[4]</sup>的部分代码如下:

```
...
//记录登录到设备的客户端数目
int iClientNum = 0;
//循环一直等待客户端连接
while( g_iStopTCPServer! = 1) {
    //给结构指针分配内存并初始化
    struct sockaddr_in * pCltAddr =
        malloc( sizeof( struct sockaddr_in) );
    bzero( pCltAddr, sizeof( struct sockaddr_in) );
    //等待客户端连接
    cltFd = accept( g_svrFd,
        ( struct sockaddr * ) pCltAddr, &iAddrSize );
    //接受客户端连接失败, 跳出本次循环
    if( cltFd == -1) {
        continue;
    }
    iClientNum ++;
    //接受客户端连接成功, 创建此客户端的工作线程
    CreateTask( 110, 8192, ClientService, cltFd,
        ( long) pCltAddr, 0, 0, 0, 0, 0 );
}
...
```

其中: CreateTask 函数封装了线程创建函数 pthread\_create, 110 为线程的优先级, 8192 为线程所占的堆栈大小,

ClinetService 为用于与相应客户端通信的线程函数, 在此线程函数中完成与相关客户端的所有交互通信。

服务器应用层软件的整体运行框图如图10所示。

服务器的每一个视频通道都有一客户端信息链表, 用以保存连接到本视频通道的客户端信息, 主要包括各客户端的 IP 地址、端口号、在本视频通道缓冲区中的读取位置和已经读取的总的字节数。前面的实时查询线程主要用于服务端查询客户端的连接情况, 每隔 50 s 向各个连接到本服务器的客户端发送心跳包, 若服务端发过去的心跳包未得到回应, 则说明该客户端已经断开, 此时应该停止给此客户端发送视频数据, 停止与此客户端的通信, 并将此客户端从视频通道的客户端链表中删去。报警查询线程用来查询是否有报警发生。若有, 即刻做出响应, 并且联动摄像机进行现场录像。设备搜索应答线程用于对网络上发送的网络搜索命令做出响应, 并将设备信息包括设备的 IP 地址、端口号、软件版本等信息反馈给上位机。

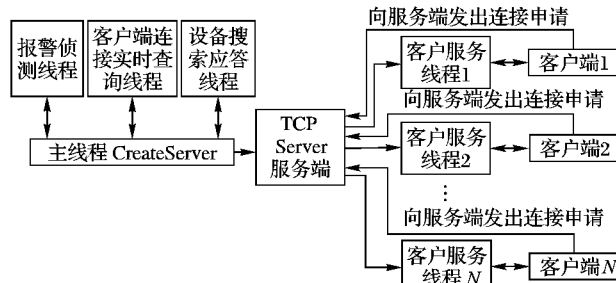


图10 设备服务端对客户端的通信管理框图

### 4 应用实例

基于前述的设计原理和方法, 本文开发了一高性能嵌入式视频服务器, 并在某公路路口搭建了一个小型视频监控系统, 对视频服务器的性能进行实例分析和测试。该监控系统由一台视频服务器(图中记为 A), 一台采用只保存半场视频图像去交错的视频服务器(图中记为 B)和一台采用拼接法去交错的视频服务器(图中记为 C), 三台摄像机, 三条视频传输线, 三台装有 Windows 操作系统的电脑, 以及专供测试用的客户端软件组成, 测试系统结构示于图11。

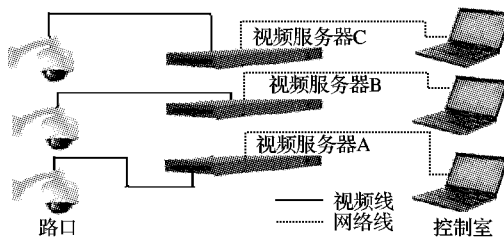


图11 视频服务器测试系统结构

测试客户端软件采用微软 Visual Studio 2005 开发, 播放库采用的 ffmpeg。ffmpeg 是一个开源免费的资源, 可以直接从网上下载其源代码, 它提供了录制、转换以及流化音视频的完整解决方案。

通过控制室安装的测试客户端的电脑登录到视频服务器并发出申请视频流的请求, 在视频服务器响应后, 就可以在控制室电脑上观看路口的场景。采用不同算法去交错的视频服务器的动态视频图像效果如图12所示。

由以上运行结果可知, 本文开发的视频服务器, 由于运用了自适应反交错算法, 可获得分辨率为4CIF (704 × 576) 的高

清晰度视频图像。



(a) 只保存半场视频图像去交错



(b) 用拼接法去交错



(c) 用自适应反交错法

图 12 不同算法去交错的视频服务器的动态视频图像截图

本文服务器图像的平滑度和清晰度远远优于运用其他去交错算法的视频服务器。此外,我们还对视频服务器的并发处理能力进行了测试。测试结果表明,本文设计开发的视频服务器可最多支持 32 路客户端的并发访问,可向最多 12 路客户端并发送视频流(即图 8 中的  $L$  为 12,图 10 中的  $M$  为 32)。因此,本文视频服务器的并发处理能力较强,达到了预期的目的。

## 5 结语

本文设计、开发了一种基于 Freescale IMX27 高性能视频处理芯片的嵌入式视频服务器。该服务器采用模块化的设计思想,各个模块之间具有很强的独立性和可扩展性。对采集到的视频数据,本服务器运用自适应反交错算法实现了视频图像的去交错,自动识别静止、运动图像并应用不同的反交错算法实现视频图像的反交错,确保了视频数据的完整性,有效地提高了图像的清晰度。运行实践表明,本服务器性能稳定,图像的清晰度高、带宽占用率低,在正常的网络运行条件下图像平滑流畅,实时性好,目前已经在变电站进行了试点应用,取得了良好的应用效果。

### 参考文献:

- [1] 杨水清,张剑,师云飞. ARM 嵌入式 Linux 系统开发技术详解[M]. 北京:电子工业出版社,2008.
- [2] 韦东山. 嵌入式 Linux 应用开发完全手册[M]. 北京:人民邮电出版社,2008.
- [3] WALL K. GNU/Linux 编程指南[M]. 张辉,译. 北京:清华大学出版社,2002.
- [4] STEVENS W R, RAGO S A. UNIX 环境高级编程[M]. 龙晋元,张亚英,戚正伟,译. 北京:人民邮电出版社,2006.
- [5] 赵方鹏,杨建华,赵忠,等. 基于嵌入式 Linux 的网络视频监控系统[J]. 测控技术,2007,26(5):55-57.
- [6] 许卫金,赵建辉,徐中佑,等. 一种具有高精度运动检测的自适应运动控制去隔行算法[J]. 信号处理,2006,22(5):658-662.
- [7] BRAMBERGER M, DOBLANDER A, MAIER A, et al. Distributed embedded smart cameras for surveillance applications[J]. IEEE Computer, 2006,39(2):68-75.

(上接第 542 页)

## 4 结语

本文构造了一种两架或三架位于同一高度层的两条固定航线上的飞行器在航线交叉点附近的冲突解脱的动态调速方案。此方案根据经验选定调速区域。进入调速区域的飞行器则需要实时监测其位置变化和与其他已进入调速区域的飞行器的位置关系,并通过一组偏微分方程和优化控制理论确定飞行器的调速方案。实验证明本方案是安全、可靠的。由于方案能够动态实时监测相关飞行器的状态,按照此方案可以由飞行员确定速度,降低了集中式飞行冲突解脱的运算复杂度,减小了系统的反应时间。多条航线的情况将会是下一步工作的重点和方向。

### 参考文献:

- [1] TOMLIN C, MITCHELL I, GHOSH R. Safety verification of conflict resolution maneuvers[J]. IEEE Transactions on Intelligent Transportation Systems, 2001, 2(2): 110-120.
- [2] TOMLIN C J, LYGEROS J, SASTRY S S. A game theoretic approach to controller design for hybrid systems[J]. Proceedings of the IEEE, 2000, 88(7): 949-970.
- [3] MAO ZHI-HONG, FERON E, BILIMORIA K. Stability and performance of intersecting aircraft flows under decentralized conflict a-

voidance rules[J]. IEEE Transactions on Intelligent Transportation Systems, 2001, 2(2): 101-109.

- [4] 夏怡凡,朱允民,马洪,等. 空中交通冲突调速最优解决方案[J]. 四川大学学报:自然科学版,2006,43(5):955-961.
- [5] 靳学梅,韩松臣,孙樊荣. 自由飞行中冲突解脱的线性规划法[J]. 交通运输工程学报,2003,3(2):75-79.
- [6] 程丽媛,韩松臣,刘星. 采用内点约束的最优冲突解脱方法[J]. 交通运输工程学报,2005,5(2):80-84.
- [7] 朱代武. 低空空域飞行冲突避让算法[J]. 交通运输工程学报,2005,5(3):46-49.
- [8] 魏光兴,杨昌其. 飞行冲突检测与调配的方法研究[J]. 中国民航学院学报,2005,23(6):1-4.
- [9] 刘星,胡明华,董襄宁. 遗传算法在飞行冲突解脱中的应用[J]. 南京航空航天大学学报,2002,34(1):35-39.
- [10] 陈晨,崔德光,程朋. 空中交通管制中改进型冲突探测算法研究与应用[J]. 计算机工程与应用,2002,38(19):250-253.
- [11] 王绍平,崔德光. 空中交通控制的冲突探测算法[J]. 清华大学学报:自然科学版,2004,44(10):1368-1371.
- [12] 邓伟,张军,吴限,等. 一种适用于航路改变情况的冲突概率预测算法[J]. 北京航空航天大学学报,2005,31(6):1327-1331.
- [13] VIETS K J, BALL C G. Validating a future operational concept for en route air traffic control[J]. IEEE Transactions on Intelligent Transportation Systems, 2001, 2(2): 63-71.