基于 H.264 的多传输模式 IP 网络视频监控系统①

赵太飞, 孙孝彬, 娄俊鹏

(西安理工大学 自动化与信息工程学院, 西安 710048)

摘 要: 设计与实现了一种基于 H.264 的多传输模式 IP 网络视频监控系统, 该系统采用 ARM9 S3C2440 作为核心硬件处理器, USB 摄像头采集到的视频数据经 X264 压缩编码后由 RTP 实时传输协议进行网络层的打包, 并通过 ARM-Linux 开发平台与 PC 机之间建立的 UDP Socket 在有线以太网、无线 Wi-Fi 和电力线三种环境中进行网络传输, PC 机在接收到视频流后进行解码与实时显示. 最后, 对系统服务器端的压缩编码效率和三种传输模式下的系统性能进行了测试和分析, 测试数据结果表明该系统具有功耗低、压缩编码效率高、视频画质清晰等特点, 并且能在不同的传输环境中保证视频传输的高实时性.

关键词: 视频监控; X264; RTP; FFmpeg; X-window

Multi-Transport Mode IP Network Video Monitoring System Based on H.264

ZHAO Tai-Fei, SUN Xiao-Bin, LOU Jun-Peng

(Faculty of Automation and Information Engineering, Xi'an University of Technology, Xi'an 710048, China)

Abstract: This paper designs and implements the multi-transport mode IP network video monitoring system based on H.264. The ARM9 S3C2440 is used as the system core hardware processor. Video data is captured by the USB camera and encoded by X264. The encoded video data is packed in the network layer by RTP and transmitted in Wired Ethernet, Wi-Fi and Power Line by UDP Socket which is established between ARM-Linux development platform and PC. After receiving the video stream, PC will decode and real-time display. Finally, compression coding efficiency in the server and system performance in three transmission modes are tested and analyzed. The test data and the results show that the system has the advantages of low power consumption, high compression coding efficiency and clear video quality, etc. And it can ensure the high real-time video transmission in various transmission environments.

Key words: video monitoring; X264; RTP; FFmpeg; X-window

随着社会的进步, 视频监控系统正在向数字化、网络化和智能化时代发展, 信息联动、统一管理已成为一个必然的趋势[1]. IP 网络视频监控系统以嵌入式硬件平台为基础, 综合运用数字视频处理技术、视频压缩编码技术、流媒体技术、IP 网络技术等与实际应用需求相结合, 相对于传统的模拟监控系统具有系统体积小、可靠性高、性价比高等特点[2].

目前, 视频监控系统给人们带来便利的同时也存在一些普遍问题, 例如布线复杂、传输媒介单一等, 不利于后期维护以及对监控过程中的突发状况进行快速应对. 同时, 对于无法使用有线以太网传输的特殊监控环境而言, 视频数据的网络传输也存在一定的局限性. 针对目前视频监控系统所存在的问题, 本文设计并实现了一种基于 H.264 的多传输模式 IP 网络视频监控系统, USB 摄像头采集到的视频数据经过 X264 压缩编码并由 RTP 进行网络层打包后, 通过 ARM-Linux 开发平

① 基金项目:国家自然科学基金委员会-中国民航局民航联合研究基金(U1433110);陕西省科技计划工业公关项目(2014K05-18);陕西省教育厅产业化培育项目(2013JC09);陕西省自然科学基础研究计划 (2013JC2-15);西安市科学计划(CXY1435(4);西安市碑林区科技计划(GX1302)

收稿时间:2015-03-05;收到修改稿时间:2015-04-26

台与 PC 机之间建立的 UDP Socket 在有线以太网、无线 Wi-Fi 和电力线三种环境中进行网络传输，最终在 PC 端经过 FFmpeg 解码后通过 X-Window 技术进行实时显示. 该系统对于无法使用有线以太网传输的特殊监控环境，能够在无线 Wi-Fi 或电力线环境中进行系统的快速搭建，实现移动视频监控，有效的解决了布线复杂、安装不便、传输媒介单一以及监控空时间固定的问题，并且对视频监控过程中的突发状况能够进行快速应对.

1　系统总体设计

多传输模式 IP 网络视频监控系统以 ARM9 S3C2440 为核心硬件处理器，以 Linux 实时操作系统为软件开发平台，总体架构以 C/S(客户/服务器)体系结构为基础，主要包括两部分：服务器端和客户端.

服务器端由视频采集模块、视频压缩编码模块、视频 IP 网络传输模块组成. 视频数据通过 USB 接口连接外部 USB 摄像头进行视频数据的实时采集，利用基于 H.264 标准的 X264 视频压缩编码技术将采集到的视频数据压缩编码后由 RTP 实时传输协议进行网络层打包，然后通过 RJ-45 网络接口以 UDP Socket 的形式进行视频流的 IP 网络传输.

客户端主要由视频解码模块和视频显示模块组成. 视频流在 PC 端被成功接收后通过 FFmpeg 进行解码由 X-Window 图形界面进行实时显示，多传输模式 IP 网络视频监控系统的总体架构如图 1 所示.

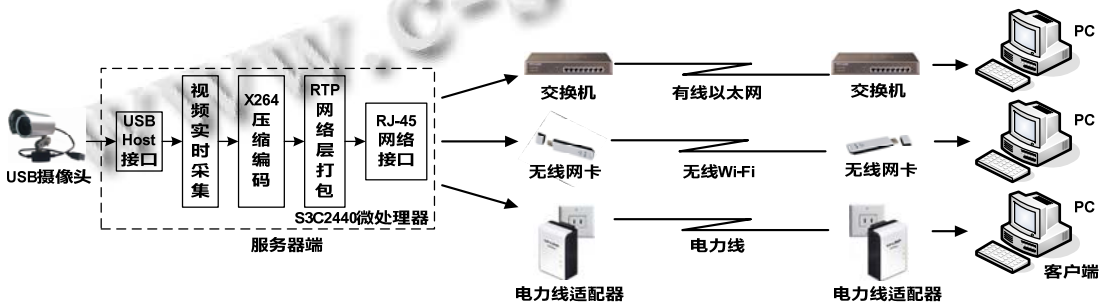


图 1　多传输模式 IP 网络视频监控系统总体架构图

2　系统硬件平台

系统核心处理芯片采用三星公司所开发的 16/32 位 RISC 微处理器 Samsung ARM9 S3C2440，处理器时钟频率在 400MHz，并且最高可以达到 533MHz.

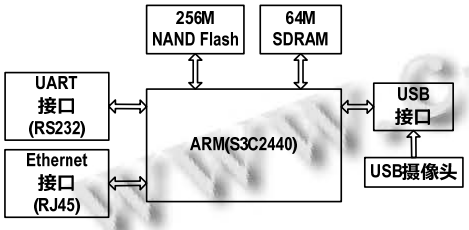


图 2　ARM-Linux 开发平台硬件框架图

ARM-Linux 开发平台硬件框架如图 2 所示. 其中，256MB 的 NAND Flash 用来存放操作系统引导程序、内核代码、根文件系统及其他应用软件，64MB 的 SDRAM 作为动态内存以保证系统的正常工作[3]. USB 接口用于连接 USB 摄像头. RS232 接口用于在系统开发阶段完成 ARM 板与主机之间的通信. 以太网接口采用 10/100 Mbit 自适应以太网卡芯片 DM9000，其中内置有 TCP / IP 协议，用来完成视频数据的实时网络传输.

3　系统软件设计

3.1 Linux 操作系统平台的构建

构建基于 ARM 的 Linux 操作系统平台主要是建立起系统开发的软件环境，主要构建步骤如下[4]:

① 建立 ARM-Linux 交叉编译环境.

② 引导加载程序. 包括固化在内部 ROM 中的启动代码和 BootLoader 两部分.

③ Linux 内核的移植. 包括对内核的裁剪、配置和移植，同时对相关驱动进行加载.

④ 文件系统的制作. 包括根文件系统的制作和其它文件系统的挂载.

⑤ 应用程序的开发. 开发具体的应用程序，并储存在已制作好的文件系统中.

3.2 系统服务器终端软件设计

3.2.1 视频数据的实时采集

V4L2(Video for Linux 2)是 Linux 内核中关于视频

设备的内核驱动, 为 Linux 系统下的视频驱动开发提供了统一的 API 接口, 使得应用程序可以使用统一的 API 函数操作不同的视频设备[5].

V4L2 采用流水线的方式实现对视频数据的实时采集过程, 基本遵循打开视频采集设备、设置格式参数、数据采集、数据处理、关闭视频采集设备的流程. V4L2 视频采集流程如图 3 所示.

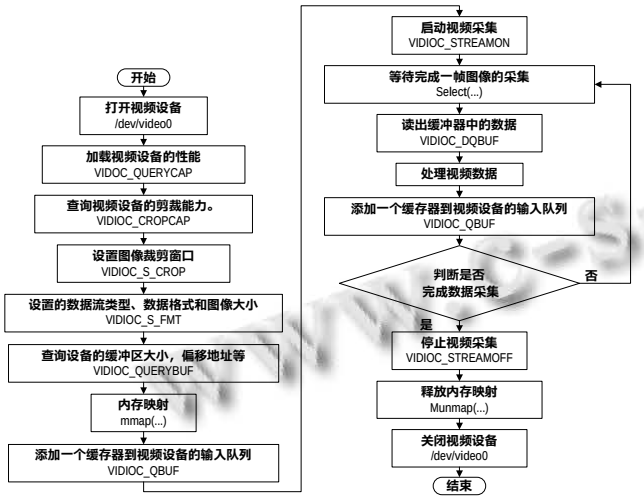


图 3 V4L2 视频采集流程图

视频采集设备主要通过调用 open() 函数以读/写的方式打开并对其进行文件操作, 本系统的视频采集设备名为“/dev/video0”. 通过调用过 iocd()命令对视频捕获格式、视频流参数等进行设置, 同时申请 V4L2 驱动分配视频缓冲区并获取和记录缓存物理空间. V4L2 支持三种数据交换模式: 直接读取设备文件方式、内存映射方式和用户指针方式. 由于内存共享机制的实时性较高, 本系统在视频数据采集过程中通过调用 mmap()函数以内存映射的方式将内核空间申请的多个缓冲区射到用户空间进行数据读取, 并以 FIFO(First In First Out)的方式循环处理缓存数据, 如图 4.

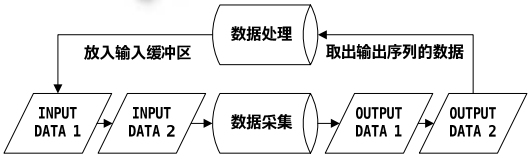


图 4 循环处理缓存数据流程图

3.2.2 基于 H.264 的 X264 编码

由于本系统所采用的基于 H.264 标准的 X264 视频压缩编码只针对 YUV420[6]格式的视频数据, 因此需要对采集的原始 RGB 视频数据进行格式转换以适配 X264 视频压缩编码的硬性输入要求. 视频数据格式由 RGB 向 YUV420 的转换过程, 主要通过调用初始化函数 sws_getContext()、转换函数 sws_scale()以及转换结束函数 sws_freeContext()完成.

X264 编码器包含 libX264 核心编码库和 X264 控制台程序两部分. libX264 核心编码库主要实现对视频数据的 VCL 编码以及 NAL 打包, 而 X264 控制台程序的主要作用是完成编码的输入输出, 解析入口编码参数并将接收到的视频数据传递给 libX264 核心编码库.

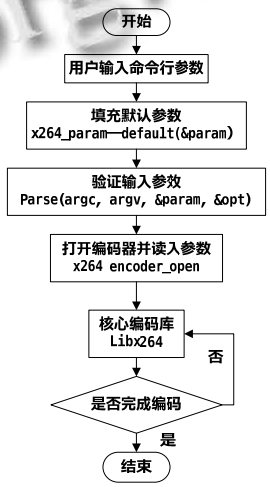


图 5 X264 编码流程图

X264 视频压缩编码流程如图 5. X264 视频压缩编码过程主要包括初始化, 压缩编码和后续内存处理三个阶段. 初始化阶段通过调用 X264_param_default()函数设置相关参数, 包括 CPU、视频参数、编码参数、码率控制参数、分析参数和量化参数等. 压缩编码阶段一般会维持着三个队列: frame_next 队列(临时缓存, 存放帧类型没确定并待压缩编码的帧队列)、frame_current 队列(按压缩编码顺序排放存放已确定帧类型并正在压缩编码的帧队列)和 frame_unused 队列(空白队列, 存放将要压缩编码的帧). 后续内存处理阶段主要释放视频数据所占内存并删除部分关键数据, 包括帧的相关信息、量化矩阵和码率控制信息等.

3.2.3 基于 RTP 协议的实时网络传输

对于 H.264 视频流的实时网络传输而言, 基于 UDP 的高层实时传输协议 RTP / RTCP 是目前解决流媒体实时传输问题的最理想方法[7], 本系统中的数据打包流程如图 6 所示.

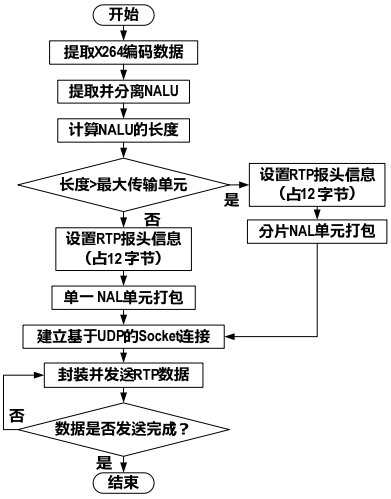


图 6　数据打包流程图

RTP 协议是基于 UDP 协议之上的, 因此需要首先创建 UDP Socket 连接, 将视频流封装上 UDP 报头和 IP 报头, 并以 IP 数据包的形式进行实施网络传输, 网络传输中 H.264 的封装格式如图 7 所示. 客户端成功接受到的数据包, 最终经过 IP / UDP / RTP 层解析后进行解码显示.

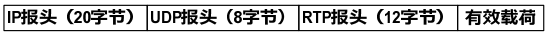


图 7　网络传输中 H.264 的封装格式

3.3 系统服务器终端软件设计

3.3.1 基于 FFmpeg 的 X264 解码

FFmpeg 是基于 Linux 操作系统所开发的一套集录制、转换、音/视频编解码功能为一体的完整的解决方案, 支持 40 种以上的编码, 例如 FLV、MPEG、MTS、AC3、MPEG4 等, 以及 90 余种的解码, 例如 MPEG、AVI、OGG、MP3、ASF 等[8]. FFmpeg 采用“主程序+核心库”的编程模式, 包括 Libavcodec、libavformat、libavutil、libswscale 四个常用的库文件, 具有友好的程序开发接口, 可以缩短程序开发的周期, 同时具有高编解码效率和可移植性, 在视音频编解码方面处于领先地位.

FFmpeg 解码流程如图 8 所示. 开始解码前, 首先需要对 FFMPEG 库进行初始化并对库中的解码器和文件格式进行注册并设置各成员参数. 其次, 打开文件并从文件中提取视频流信息, 同时在库中以穷举的方式查找 CODEC_TYPE_VIDEO 类型的流. 最终打开与待解码视频流相对应的解码器(CODEC_ID_H264)进行实时解码, avcodec_decode_video2()为其核心解码函数为.

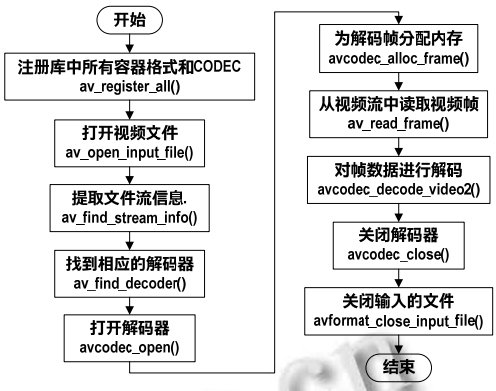


图 8　FFmpeg 解码流程图

3.3.2 视频流的 X-Window 显示

视频显示采用 X-Window 技术以内存共享的方式实现视频流的实时显示, 其中包括 X-Server、X-Client 和 X-Protocol 三部分[9].

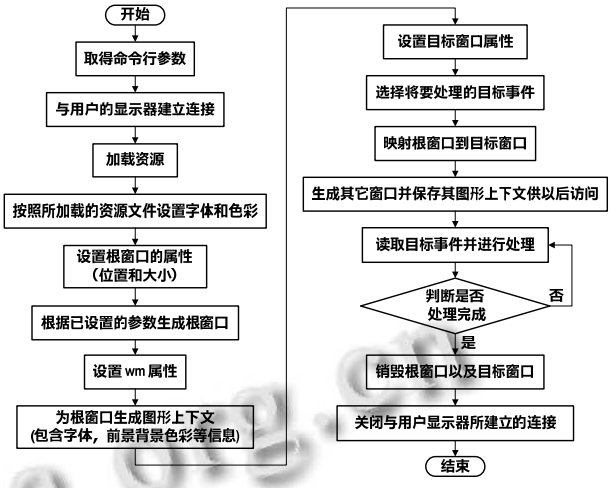


图 9　X-Window 视频显示流程

X-Window 视频显示流程如图 9 所示. 在成功创建 X-Server 和 X-Client 并建立连接后, 首先通过所申请的图形上下文结构(GC)对窗口参数进行设置并传给绘图函数, 例如屏幕类型, 屏幕尺寸, 颜色格式支持等. 其次, 建立根窗口以及目标窗口并通过创建共享内存段来申请共享内存, 通过调用 XMapWindow()函数在根窗口和目标窗口建立映射并读取目标事. 最终, 通过调用函数 XShmPutImage()将解码后视频数据实时的显示在目标窗口中.

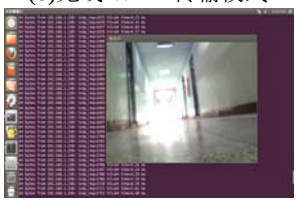
4　系统测试与结果分析

在搭建好系统的测试环境, 同时将 ARM-Linux 开

发平台、PC 机和虚拟机的 IP 地址分别设置为“192.168.1.230”、“192.168.1.23”和“192.168.1.108”进行三方 ping 通后，在 ARM-Linux 开发平台的终端下启动服务器并等待连接，在 PC 机中的 Ubuntu12.10 终端下启动客户端去连接服务器，此时系统就处于运行状态了．系统测试实物和运行效果如图 10，其中图 10(a)、10(b)分别为电力线传输模式和无线 Wi-Fi 传输模式下系统实测图、图 10(c)为系统实际运行效果图.

(a)电力线传输模式

(b)无线 Wi-Fi 传输模式



(c)系统实际运行效果

图 10　系统测试实物和运行效果图

首先，在系统服务器端对基于 H.264 标准的 X264 视频压缩编码效果进行了测试，表 1 是在编码过程中随机对相同原始视频帧数据的视频压缩编码信息进行的连续十组采样值．由表 1 中所测数据可知，该系统服务器端中视频数据的 X264 压缩编码性能较高，满足网络传输带宽的要求.

表 1　服务器端视频压缩信息

参数 / 组数	原始视频帧 (bytes)	压缩编码后的视频帧(bytes)	编码效率 (%)	平均编码效率(%)
1	24582	298	98.788	98.641
2	24582	376	98.470	
3	24582	279	98.865	
4	24582	486	98.023	
5	24582	204	99.170	
6	24582	411	98.238	
7	24582	384	98.438	
8	24582	281	98.857	
9	24582	243	99.011	
10	24582	357	98.548	

① 有线以太网传输模式

表 2 为系统在有线以太网传输模式下的性能测试数据．测试结果表明在有线以太网传输模式下系统平均帧速率可达到 21fps、平均时延 0.6s、平均丢包率小于 0.1%.

表 2　有线以太网传输模式下的系统性能测试数据

参数 / 组数	平均帧速率 (fps)	平均时延 (s)	发送包数 (个)	接收包数 (个)	丢包率 (%)	平均丢包率 (%)
1	21.2	0.61	70254	70229	0.036	0.055
2	21.3	0.61	70589	70549	0.057	
3	21.2	0.62	73247	73197	0.068	
4	21.4	0.63	70862	70822	0.056	
5	21.1	0.59	74235	74199	0.048	
6	21.3	0.63	70475	70441	0.048	
7	21.2	0.61	72763	72717	0.063	
8	21.2	0.61	70362	70328	0.048	
9	21.1	0.57	71723	71688	0.049	
10	21.4	0.64	73890	73832	0.078	

② 无线 Wi-Fi 传输模式

无线 Wi-Fi 信号由 54Mbps IP-COM W323G+通过 PC 端发出，发射功率为 18dBm, ARM-Linux 开发平台由 1000VA/500W 的 UPS 电源供电并通过 54Mbps TL-WN321G+无线网卡接入无线网络环境中，表 3 为系统在无线 Wi-Fi 传输模式下的性能测试数据．由于无线 Wi-Fi 信号带宽受限，加上外界电磁场的干扰，系统性能会因此受影响，从测试数据可以看出，相比于有屏蔽的 Wi-Fi 环境而言，无障碍 Wi-Fi 环境中的系统性能要更好，并且由于屏蔽度的增加，系统平均帧速率、平均时延以及平均丢包率会随 Wi-Fi 信号的减弱而不断下降.

表 3　无线 Wi-Fi 传输模式下的系统性能测试数据

参数 / 组数	测试场景	平均帧速率(fps)	平均时延(s)	发送包数(个)	接收包数(个)	丢包率(%)	平均丢包率(%)
1	距信号源直线距离 9m，无障碍	15.4	1.31	54239	54148	0.168	0.168
2		15.2	1.30	54642	54544	0.179	
3		15.5	1.27	53867	53799	0.126	
4		15.2	1.31	54726	54630	0.178	
5		15.3	1.29	53941	53837	0.193	
6	距信号源横向距离 9m，横向穿单层墙	13.7	1.34	48727	48572	0.318	0.336
7		13.9	1.34	49243	49071	0.349	
8		13.7	1.36	48801	48627	0.357	
9		13.8	1.35	49248	49075	0.351	
10		13.5	1.38	48647	48497	0.308	
11	距信号源横向距离 9m，横向穿双层墙	13.2	1.41	46576	46385	0.410	0.403
12		13.1	1.37	46705	46517	0.402	
13		13.1	1.40	46329	46145	0.397	
14		13.3	1.38	46513	46334	0.385	
15		13.3	1.37	46527	46332	0.419	
16	距信号源纵向距离 7m，纵向穿单层墙	13.8	1.35	47945	47763	0.380	0.381
17		13.4	1.37	48413	48235	0.368	
18		13.9	1.34	48317	48137	0.373	
19		13.7	1.35	47976	47792	0.382	
20		13.8	1.36	48124	47931	0.401	

③电力线传输模式

系统通过 TL-PA201 200M 电力线适配器将视频流加载于电力线中进行传输，分别对 3 米到 170 米电力线传输距离下的系统性能进行了测试，表 4 为系统在电力线传输模式下的性能测试数据．测试结果表明，在相同的时段与负载下，随传输距离的增加，电力线中负荷加大、信号衰减变大、噪声干扰变强，因而导致系统平均帧速率、平均时延和平均丢包率逐步下降．

表 4　电力线传输模式下的系统性能测试结果

参数 / 组数	距离(m)	平均帧速率(fps)	平均时延(s)	发送包数(个)	接收包数(个)	丢包率(%)	平均丢包率(%)
1	3	15.2	0.81	54638	54295	0.629	0.628
2		15.1	0.80	54259	53927	0.613	
3		15.1	0.78	54645	54316	0.600	
4		15.3	0.77	53879	53528	0.652	
5		15.2	0.81	53954	53606	0.645	
6	30	13.1	1.72	47551	47214	0.708	0.827
7		13.5	1.69	46915	46553	0.772	
8		13.4	1.64	47112	46697	0.882	
9		13.4	1.71	47531	47130	0.843	
10		13.2	1.71	46958	46522	0.928	
11	60	10.3	3.33	36254	35713	1.493	1.517
12		10.2	3.31	36589	36046	1.485	
13		10.6	3.23	36247	35674	1.583	
14		10.3	3.34	35862	35325	1.500	
15		10.4	3.27	36237	35685	1.524	
16	100	7.2	4.60	25124	24712	1.621	1.625
17		7.1	4.62	25416	24991	1.672	
18		7.3	4.57	25195	24798	1.576	
19		7.2	4.61	25476	25058	1.640	
20		7.6	4.59	25218	24811	1.614	
21	130	3.4	6.09	10845	10644	1.853	1.895
22		3.2	6.12	11286	11071	1.905	
23		3.6	6.07	10949	10743	1.881	
24		3.2	6.15	11147	10933	1.920	
25		3.3	6.12	11541	11320	1.915	
26	170	0.4	7.78	1746	1700	2.635	2.671
27		0.5	7.73	1823	1770	2.907	
28		0.4	7.85	1815	1766	2.700	
29		0.4	7.79	1733	1692	2.366	
30		0.5	7.82	1819	1869	2.749	

通过对以上三种不同传输模式的测试结果分析可知, 首先相比于无线 Wi-Fi 传输和电力线传输而言, 系统在有线以太网中的传输性能更加优良; 其次系统的传输性能在无障碍 Wi-Fi 环境中要比有屏蔽的 Wi-Fi 环境中更加良好, 同时随着屏蔽度的增加而不断下降; 最后系统在电力线环境中的传输性能会随着传输距离的增加所带来的电力线负荷加大、信号衰减变大、噪声干扰变强等原因而逐步下降, 当电力线传输距离达到 170 米以上时, 系统传输性能将会变得非常不理想.

5 结语

本文设计并实现了一种基于 H.264 的多传输模式 IP 网络视频监控系统. 通过搭建测试环境对系统服务器端的压缩编码效率以及有线以太网、无线 Wi-Fi、电力线三种传输模式下的系统性能进行了测试和分析, 测试数据结果表明该系统具有功耗低、压缩编码效率高、视频画质清晰等特点, 并且能在不同的传输环境中保证视频传输的高实时性.

参考文献

1 Guan M, Chen Y. Design and implementation of a remote video monitoring system based on embedded Linux. International Conference on Computer Application and System Modeling (ICCASM 2010). Taiyuan, China. 22-24 Oct. 2010. 14. 316–319.

2 Liu S, Zhao JJ, Fan XL. The embedded video surveillance system based on V4L2. Microcomputer Applications, 2011, 32(1): 37–42.

3 Wang WH, Gao MY. Design of embedded media player based on S3C2440 and SDL_FFMPEG. 2011 International Conference on Electrical and Control Engineering (ICECE). Yichang, China. 16-18 Sep, 2011. 2979–2982.

4 孙纪坤,张小全.嵌入式 Linux 系统开发技术详解—基于 ARM.北京:人民邮电出版社,2006.

5 Liang HJ, Wang S. Based on ARM S3C2410 and streaming media technology, network video capture. Computer, 2007, 23: 2–5.

6 刘云鄰,王树东.基于 SSE2 的 YUV 与 RGB 色彩空间转换.中国图像图形学报,2010,1:45–49.

7 彭铁钢,刘国繁,曹少坤等.基于 ARM 的嵌入式视频监控系统设计.计算机工程与设计,2010,31(6):1191–1194.

8 覃艳.基于 ffmpeg 的视频格式转换技术研究.电脑知识与技术,2011,4(12):2912.

9 Kernighan W, Pike R. The Unix Programming Environment. 3rd edition. Polirom Publishing House, 2002, (1): 25–30.