doi:103969/j. issn. 0490-6756. 2010. 04. 017

JPEG 压缩编码在 DM642 平台上的实现及优化

许光辉，王正勇，卿粼波，徐 萍
(四川大学电子信息学院，成都 610064)

摘 要：结合 TMS320DM642 处理器 VP 口视频采集原理和 JPEG 压缩编码原理，研究了在 TMS320DM642 处理器平台上实现图像采集和 JPEG 压缩编码的具体方法. 以此为基础，深入探索了 JPEG 压缩编码在 TMS320DM642 处理器平台上的优化，并提出了基于 DCT 系数低通滤波原理的 JPEG 压缩编码优化算法，最后对优化结果做出了数值分析. 分析结果表明，该算法不仅保证了图像的质量，而且提高了 JPEG 压缩编码的速度和效率.

关键词：JPEG 压缩编码；DM642；DCT 系数；优化
中图分类号：TP202.7；TN492；TP29 **文献标识码**：A **文章编号**：0490-6756(2010)04-0768-07

Realization and optimization of JPEG compression based on DM642 platform

XU Guang-Hui, WANG Zheng-Yong, QING Lin-Bo, XU Ping
(School of Electronics and Information Engineering, Sichuan University, Chengdu 610064, China)

Abstract: In combination with the principle of video capture using the VP port of TMS320DM642 processor, as well as the JPEG compression theorem, a concrete method is studied for the realization of video capture and JPEG compression in the TMS320DM642 processor platform. On this basis, the optimization of JPEG compression in the TMS320DM642 processor platform is explored intensively, and a specific optimization algorithm of JPEG compression based on the low-pass filter to DCT coefficients is investigated simultaneously. As an examination of the optimization methods, a numerical analysis is implemented. Analysis results show that not only the image quality is preserved, but also the encoding speed and efficiency are both improved.

Key words: JPEG compression, DM642, DCT coefficients, optimization

1 引 言

随着目标跟踪、机器人导航、自动驾驶、交通监视等领域的发展，在以 DSP 为核心的嵌入式实时图像处理系统上实现各种图像处理算法越来越受到关注. 作为 PC 机上图像处理领域常用的 JPEG 压缩算法以其压缩比高、失真小等特点，也被广泛使用于嵌入式图像处理系统. TI 公司推出的针对多媒体领域的 32 位定点 DSP 处理器 *TMS320DM642*(以下简称 *DM642*)在图像处理领域有着卓越的性能[1]. 目前，国内外的研究大多是基于 TI 的 DM642 评估板上运行 JPEG 压缩编码例程，优化方法只是概括介绍，工程应用可操作性差. 本设计在实验室自主开发的以 DM642 处理器为核心的多媒体处理平台上，实现了对 720×576 大小、8-bit 色深、YUV4:2:0 格式图像的采集和

收稿日期：2009-08-03
作者简介：许光辉(1984—)，男，河南商丘人，硕士研究生，主要研究领域为电路与系统. E-mail：chhrdw@gmail.com
通讯作者：王正勇. E-mail：690728634@sina.com

JPEG 压缩编码,并研究了针对该平台的 JPEG 压缩编码优化,包括针对该平台的图像数据流调度优化和代码优化,使用 C 编译器优化选项进行优化,以及 DCT 系数低通滤波优化. 这对在该平台上高效地实现更复杂的图像处理算法和图像实时处理系统也有指导意义.

2　视频采集原理

DM642 具有针对多媒体应用开发的功能模块和外设接口,如:三个可配置的视频接口(VPx),外部存储单元接口(EMIF),I2C 总线模块,EDMA 模块等,特别适合用于视频图像的实时采集、处理和传输[2].

本设计使用到的硬件原理图如图 1 所示. 模拟摄像机把采集到的视频数据经过 SAA7113 视频解码芯片解码后,通过 VP0 口传送给 DM642 进行 JPEG 压缩编码. 由于 DM642 片上内存只有 288K-Byte,所以本系统通过 DSP 的 EMIF 接口并行连接了 2 片 32 位的 MT48LC4M32 芯片进行内存扩展,大小为 32M-Byte. DSP 的 EMIF 模块能够使 DSP 与 SDRAM 无缝连接,不需要外部译码电路[4]. 具体电路连接如图 1 的 EMIF 接口所示.

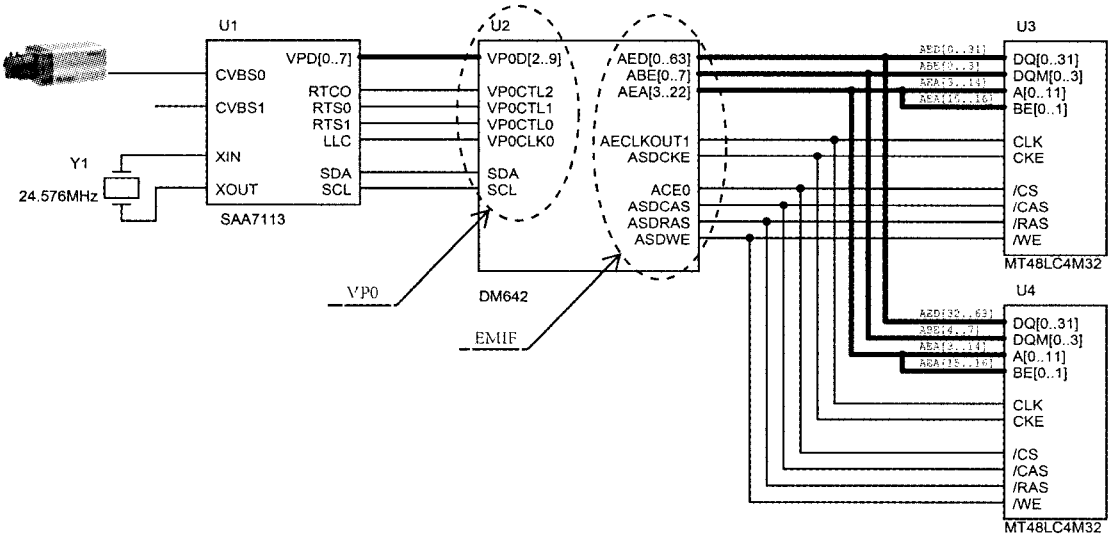

图 1　视频采集硬件原理图

Fig. 1　Principle diagram of video capture hardware

下面结合硬件电路阐述本设计视频采集原理.

(1)视频采集和视频解码. 该部分对应的硬件为模拟摄像机和 SAA7113 视频解码芯片.

模拟摄像机把采集到的 PAL 制式复合视频信号(CVBS)传送给 SAA7113 进行视频解码和 A/D 转换. SAA7113 由 TMS320DM642 通过 I2C 总线进行参数配置和控制,具有多种输入输出模式. 本设计中复合视频信号经 SAA7113 解码后输出为标准的 ITU-R BT. 656 YUV4:2:2 格式(8-bit)的数字视频信号,通过 DM642 VP0 口的 8 位数据总线 VP0[2:9]输入到 VP0 口. 每帧数据都由奇偶两场组成.

(2)视频数据打包. 该部分对应硬件为 TMS320DM642 的视频端口 VP0 口. 专门的视频端口是 TMS320DM642 作为多媒体处理芯片的特色之一,通过设置可自动完成视频数据的缓冲和打包处理.

YUV 数据的包装方式有两种,一种是 Packed Format,以像素点为单位把 Y 和相对应的 UV 包装在一起. 另一种是 Planar Format,把 Y、U、V 拆成三个平面(Planar),分别包装. BT656 格式视频数据就是以 Packed Format 方式包装,当它以 Cb、Y、Cr、Y、Cb、Y……顺序传输给 VP0 口后,VP0 口从中提取 YUV 分量数据,按照 Planar Format 方式包装,并存放于 VP0 口对应的 YUV 分量 FIFO 中. 当 FIFO 中的数据达到一定阈值就会触发 EDMA 传输,各分量被分别搬移到 SDRAM 中专门开辟的 YUV 存储空间,用于 JPEG 编码. 在此过程中,通过对 EDMA 参数的设置,在搬移数据同时完成奇偶场合并.

(3)JPEG 编码. 该部分由 DM642 处理器实现. 存于 SDRAM 中的数据是 YUV4:2:2 格式,在 JPEG 编码前要对 UV 数据进行下采样. 当把视频数据通过 EDMA 搬移给 JPEG 编码模块时, UV 数据隔行搬移,实际上只是读取了奇场或者偶场数据,从而间接完成下采样. 最后送给 JPEG 压缩编码模块进行编码.

图像采集独立自主进行,DSP 除了对采集模式进行设定外,不参与采集过程,以满足系统实时性要求. 测试结果表明,该系统采集图像清晰,处理速度快,运行稳定可靠[1].

3 JPEG 压缩编码在 DM642 上的实现

JPEG(Joint Photographic Experts Group)是适用于彩色和单色多灰度或连续色调图像的压缩标准,主要用于静止图像的压缩. JPEG 标准分为两种基本压缩编码算法:基于 DCT 的有失真算法和基于空间线性预测的无损压缩算法[3]. 本设计实现的即是基于 DCT 的有失真算法,算法流程图如图 2 所示.

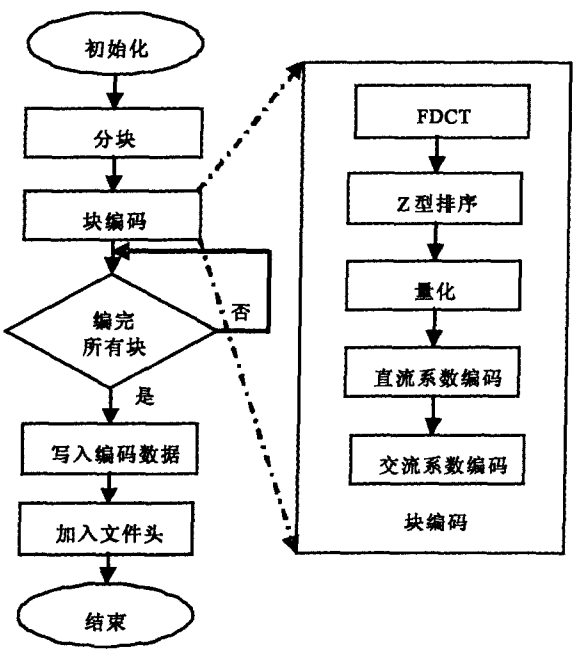

图 2 JPEG 压缩编码流程图

Fig. 2 Flow-chart of JPEG Compression

下面结合硬件平台和 JPEG 压缩编码流程图,具体研究 JPEG 压缩编码的实现过程.

3.1 初始化

JPEG 压缩编码程序的初始化主要包括视频数据空间的开辟,以及量化表和 Huffman 码表的初始化.

量化表有两个,分别对应于 Y 分量和 U、V 分量. Huffman 码表有四个,分别对应于 FDCT 变换得到的四种数据(亮度 Y 和色度 UV 的直流 DC 系数及交流 AC 系数). 四个码表分别存放于对应的四个结构体数组中. 该结构体定义如下:

```
typedef struct tagHUFFCODE
{
    Uint16 code;//码字
    Uint16 length;//码字长度
    Uint16 value;//码值
}HUFFCODE;
```

其中,码值 value 对应于量化后的 DC 或 AC 系数,同时也确定了尾码的长度;码字 code 是对应码值的 Huffman 编码结果;码字长度 length 是对应码字二进制表示的长度.

3.2 数据的分块

对于 YUV4:2:0 格式数据,JPEG 文件写入格式要求以宏块(MCU)为单位写入. 宏块的切分与颜色分量采样率有关,对于 YUV4:2:0 格式图像,宏块由 YYYYUV 序列组成,其中每个 Y、U、V 代表一个 8×8 的数据单元,记作 DU. 因而数据分块时要以此顺序切分,切分顺序示意图如图 3 所示. YUV 三个分量按照从上到下、从左到右的顺序分别切分出 DU,并按顺序组合成 YYYYUV 宏块,送给块编码模块. JPEG 编码是以 DU 为单位进行的. 后面编码数据写入 JPEG 文件也是按照这个顺序进行.

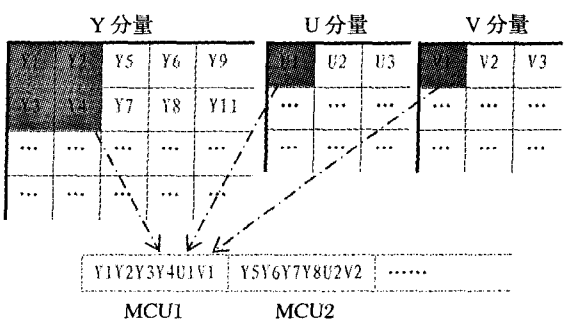

图 3 数据块切分顺序示意图

Fig. 3 The sequence of data block dividing

3.3 FDCT 变换

为使图像在 FDCT 变换后平均功率集中在 0 值附近,在进行 FDCT 变换之前要对每个 8-bit 的 YUV 数据减去 128,使其动态范围由 0～255 偏移至－128～127.

FDCT 变换调用库函数 IMG_fdct_8x8()来实现. 该函数包含于 TI 公司提供的 img64x. lib 库中. 调用该函数需要在工程中包含库文件 img64x. lib 和头文件 img_fdct_8x8. h. 函数 IMG_fdct_8x8()使用陈氏 DCT 算法实现 FDCT 变换,并且已经针对 DM642 平台进行了汇编指令级的优化,只需 168 个 CPU 主频周期.

FDCT 变换后得到的 8×8 系数矩阵如图 4 所示. 其左上角是直流 DC 系数,其余 63 个是交流 AC 系数. 从该矩阵左上角到右下角,AC 系数对应频谱逐渐升高. YUV 数据进行 FDCT 变换得到四种数据:亮度 Y 的 DC 和 AC 系数,色度 UV 的 DC 和 AC 系数.

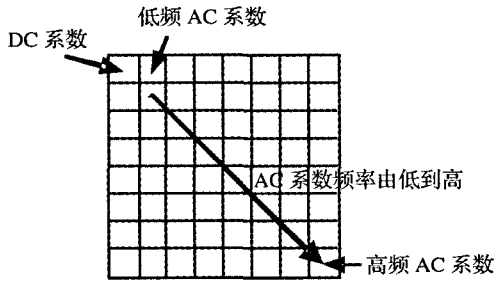

图 4　DCT 系数矩阵示意图

Fig. 4　Schematic diagram of DCT coefficients matrix

3.4 量化和 Z 型排序

对于不同彩色分量的不同频率的 DCT 系数采用不同的加权系数进行量化. JPEG 标准对量化表给出了建议,同时也支持用户自定义的量化表.

本设计使用 TI 的库函数 IMG_quantize()对 DCT 系数进行量化,该函数的使用方法类似于函数 IMG_fdct_8x8(). 该量化过程是不可逆的有损压缩,是造成 JPEG 压缩信息损失的主要原因.

为有利于后面的零游程统计和熵编码,使变换后的 DCT 系数低频分量先出现,高频分量后出现,以增加零游程长度,要对图 4 中的 DCT 系数进行 Z 字形排序.

3.5 Huffman 熵编码

Huffman 编码是一种基于概率统计的无损可变长编码方法,出现概率大的字符用短码表示,出现概率小的字符用长码表示,并用非续长代码编码保证唯一可译性[3]. 直流 DC 系数进行差分编码,交流 AC 系数进行零游程编码. 具体编码过程实现如下.

(1)DC 系数差分编码

差分编码即差分脉冲编码调制技术,即对最相邻两个 DU 的 DC 系数的差值(DIFF)进行编码. 编码结果包括两部分:前缀码(SSSS)＋尾码. 前缀码指明尾码的有效位数,用标准的 Huffman 编码[3]. 程序中,它是对 DIFF 值查找对应的 DC Huffman 码表得到的,即 huffCode[i]. code,同时该 code 所对应的 huffCode[i]. length 也指明了尾码的二进制有效位数. 尾码采用自然编码,如果 DIFF 非负,用对应二进制原码表示;如果 DIFF 为负数,即用反码表示,长度由对应的 huffCode[i]. length 指定.

(2)AC 系数零游程编码

其余 63 个 AC 系数采用零游程编码,可以将其表示为"NNNNSSSS＋尾码"的形式. 其中,4 位 SSSS 和尾码的含义与 DC 系数的类似. 4 位 NNNN 表示相对于前一个非零值的零游程计数,一般在 0～15 之间. 这里将"NNNNSSSS"组合为一个新的"前缀码",对它查找 AC Huffman 码表得到 Huffman 编码结果,然后把该结果与尾码的自然编码组合在一起就是 AC 系数的 Huffman 编码结果[3].

有两种特殊情况:如果零游程计数达到 16,则用"NNNNSSSS"＝"11110000"表示,再继续重新对零游程计数;如果后面 AC 系数全为 0,则以 EOB 结束该 DU 的编码[3]. 因此零游程计数从最后一个 AC 系数开始计数比较方便.

3.6 写入编码数据和加入 JPEG 文件头

JPEG 标准规定,块编码数据的写入需按宏块(MCU)的顺序依次写入,其写入顺序与数据块的切分顺序一致,参见图 3.

JPEG 文件头主要用于标记 JPEG 图像数据的相关信息. 它使用 Motorola 格式,即文件中的字节正序排列,高位字节在前,低位字节在后. JPEG 文件头由段(Segments)构成,每个段都是从一个标记字开始. 每个标记字都由固定值 0xFF 加上一个代表该标记字实际意义的字节组成. 例如,0xFFD8 是 JPEG 文件的开始标记字(SOI). JPEG 文件头后是以 0xFF00 开头的各扫描行数据,最后

以0xFFD9(EOI)结束,就构成了一个完整的JPEG文件.

4 JPEG压缩编码的优化

针对DM642平台的程序优化是一个繁琐反复的过程,不仅要提高速度,而且要验证程序运行结果的正确性.优化方法归纳总结如下.其中,程序运行时间通过DM642的Timer模块测量,单位是CPU的主频周期Cycle.

4.1 图像数据流调度优化

DM642上程序运行的实质就是处于对图像数据的搬运和处理的循环往复中,因而其程序结构调整的最主要工作就是结合系统硬件条件调整数据流,充分发挥DSP的并行性和软件流水性能,使DSP用最少的时间处理最多的图像数据.实验证明,合理的数据分块和调度处理会使编码速度提高很多.

原程序是以一个8×8 DU为单位进行FDCT、量化、Z排序和熵编码等,每次都要等待上一个DU编码完成才对下一个DU开始处理,而且每次都要从外部扩展内存读取数据,因而耗费时间比较长.实际测试程序运行时间为:完成一帧720×576大小YUV4:2:0格式图像的JPEG压缩编码所需时钟周期为26 Million Cycles,除以CPU主频600MHz换算成时间约为0.434秒,编码速度很低.因此我们需要对图像数据的处理流程进行调整优化.

原程序中所有的代码和数据都放于外部扩展内存中,但是DM642的CPU对外部存储器SDRAM的实际访问速度只有大约100MB/s,远远低于对二级缓存L2的访问速度(9.6GB/s),对SDRAM的频繁访问阻碍了编码速度的提高.优化时,程序从256K-Byte的L2中划分128K-Byte作为缓存(ISRAM)使用,主要用于缓存即将编码的数据块和频繁调用的代码.

库函数IMG_fdct_8x8()和IMG_quantize()可对多个连续存放于内存中的DU数据进行打包处理,为减少函数调用的开销,优化时可以一次读取多个DU送入FDCT和量化模块进行打包处理.具体方法为:把一帧图像的YUV数据按照YYYYUV顺序切分成宏块MCU,缓存多个MCU并连续存放于片上内存中,然后对多个MCU一起依次进行FDCT、量化、Z排序和熵编码.需要注意的是,由于片上内存有限,而且软件流水的形成要求循环体不能过大,因而要求一次处理的图像数据不能太大,本设计以16行YUV数据为单位进行编码处理.

经过以上优化后,同一帧图像完成JPEG压缩编码需要时钟周期为16 Million Cycles(约0.270秒).

4.2 代码优化

代码优化主要是基于TI提供的库函数和DM642处理器运算单元特性进行的,包括以下几种方法.

(1)尽量使用TI提供的经过线性汇编优化的算法库中的函数[4],如IMG_fdct_8x8().调用这些函数可以获得比常规C代码函数优秀得多的性能.

(2)合并小的计算密集型函数或循环体.DM642采用超长指令字(VLIW),即在一个时钟周期内最高可把8条32位指令,总字长为256位的指令包同时分配到8个并行处理单元进行并行计算.软件流水就是要充分发挥DSP的并行处理特性.

通过分析,DSP中的代码可以分为两类:一类是有很多程序判断跳转指令的事务性代码,一类是专门进行数据计算的计算性代码.事务性代码会打断软件流水的形成.为减少事务性代码,增加计算性代码,以利于并行处理和软件流水的形成,程序采取以下优化:把小的函数改为内联函数或者宏来实现,把判断跳转改为查表或者使用一些专门Intrinsics实现,把可以合并的多个计算密集型语句或者循环体合并为一体[5].例如:在读取图像数据时既要进行符号扩展又要进行直流偏移,可以合并为一句:

```
*(pYUVBuf + k) = *(pyl + n + m * width) -128;
```

又例如:计算前缀码SSSS时要根据DCT系数幅度计算尾码的有效位数,需要用一个if-else判断语句块计算.优化时使用自定义宏:

```
#define ComputeVLI(val) (31 - _norm(_abs(val)));
```

其中_abs()和_norm()都是C64X编译器提供的专门内联函数(Intrinsics),优化后该部分代码的运行时间由1173856 Cycles降到177816 Cycles,速度升了6.6倍.

(3)编写代码时注意以下原则也可以提升代码运行性能:尽量用逻辑运算代替乘除运算;用指针

操作代替数组下标寻址;使用 int 型数据作为循环计数器,避免产生符号扩展指令;充分利用 DM642 的具有单指令多数据流(SIMD)的 Intrinsics 函数,例如_sub2()函数可以一次处理两组 16-bit 数据的减法[5].

通过以上步骤的代码优化,完成一帧图像的 JPEG 压缩编码时间为 0.21 秒,减少了约 0.05 秒.

4.3 使用 CCS 的 C 编译器优化选项和软件流水线技术

CCS 是 TI 公司提供的针对 DSP 系统开发的集成开发环境,内部集成了功能强大的 C 编译器.该编译器提供了分为若干种类和等级的自动优化选项,通过 CCS->Project->Build Options->Compiler 选项卡设置.例如:选择"-mv6400"使编译出的代码可以使用 DM642 处理器增加的硬件资源和指令;通过对选项"Opt Level"的选择把程序的优化级别从不优化设置到最高级的-o3 优化.优化选项的设置需要根据编译的具体程序,选择合适的优化选项,在获得速度的同时保证程序的稳定性.

如果把优化级别设置为-o2 或-o3,则编译器将根据程序尽可能地安排软件流水线.软件流水线技术用来对一个循环结构的指令进行调度安排,使之成为多重迭代循环并行执行.在 DSP 算法中存在大量的循环操作,因此充分地运用软件流水线方式,能极大地提高程序的运行速度[6].但使用软件流水线还有下面几点限制,这在编写和优化程序时要注意.

(1)循环结构不能包含代码调用,但可以包含内联函数.

(2)循环计数器应该是递减的.

(3)循环结构不能包含 break,if 语句不能嵌套,条件代码应当尽量的简单.

(4)循环结构中不要包含改变循环计数器的代码.

(5)循环体代码不能过长,因为寄存器(32 个)的数量有限,应该分解为多个循环.

在软件流水线的运用上,应该尽量使复杂的循环分解成简单的小循环,以避免寄存器的数量不够;对于过于简单的循环,应该适当的展开,以增加代码数量,增加流水线中的迭代指令[6].

把优化选项设置为-o3,完成同一帧图像 JPEG 压缩编码的速度可以提升 6 倍左右.

4.4 DCT 系数低通滤波优化

使用 CCS 的 Profile 工具对代码分析后,我们发现量化部分非常耗时,这主要是由于量化部分要对每个 DCT 系数进行除法的浮点运算,运算量很大.因而本文首先做了以下两个设想:(1)能否对量化步长做适当调整,以便用移位操作代替除法浮点运算;(2)针对一些对图像细节要求不是太高的图像处理领域,在保证图像质量的同时,能否对 DU 进行低通滤波,只对低频 DCT 系数量化编码,以减少量化和零游程编码运算量.

基于以上设想,对量化步骤做以下调整:在对每个 DU 的 DCT 系数量化时,只对前 N 个低频 DCT 系数进行向右移位运算,而把后面的 64-N 个高频 DCT 系数的量化结果直接赋值为零.这样在变除法为移位运算的同时也减少了量化的次数,不仅简化了量化的运算步骤,而且由于零行程的增加同时也简化了 AC 系数的零行程编码,极大地提高了编码速度.该算法的实际效果就是对每个 DU 进行了低通滤波,因而称为 DCT 系数低通滤波优化.

通过对常用亮度和色度量化系数表的分析,DC 系数和低频 AC 系数的量化系数集中在 8 或者 16 两个值附近,量化时只需进行右移 3 位或 4 位的运算.高频 AC 系数的量化系数一般都比较大,因而直接把其量化结果赋值为零,理论上不会对图像质量产生大的影响.

为准确衡量 DCT 系数低通滤波优化对 JPEG 图像质量和编码时间的影响,本设计使用高、低复杂度两组图片(高复杂度图片以人脸图片为主,低复杂度图片以车牌图片为主)进行了 PSNR 值和编码速度提升值(Promotion Value of Encoding Speed,以下简称 PVES)的测试.对每组图片的测

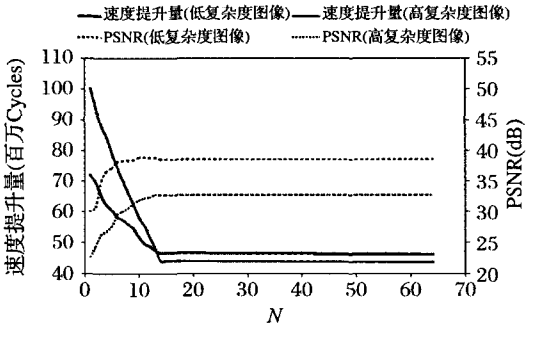

图 5　N 值与 PSNR 和 PVES 关系曲线图

Fig. 5　The relationships between N and PSNR and PVES

试数据统计平均后得到的N值、PSNR值和PVES值三者之间的关系如图5所示.其中,PSNR值是以正常量化编码的JPEG图片为参考图片,把进行DCT系数低通滤波优化后的JPEG图片传输到PC机上,用Matlab程序计算得到的.PVES值由未加入优化模块的编码时间减去优化后的编码时间得到,单位是百万Cycles.程序对前N个DCT系数进行右移3位的量化运算,N的取值范围是1～64.

由图5可知:当N>12时,PSNR和PVES值都趋于一个定值,这是因为高频DCT系数本来就存在很多零值,大的N值对PSNR和PVES值的影响因子已经很小.对于低复杂度的车牌图片,当N取10时,PSNR值达到峰值约为39dB;对于高复杂度的人脸图片,当N取12时,PSNR值达到峰值约为33dB.在没有打开编译器的－o3优化选项的情况下,在PSNR获得峰值时,PVES值平均约为0.085秒,速度提升幅度很大,所以为保证图像质量,N的取值以PSNR达到峰值为第一条件.进行DCT系数低通滤波优化前后的图像质量对比如图6所示.由图可以看出,优化后图像质量损失很小.

优化前 优化后

图6 DCT系数低通滤波优化前后图像质量对比

Fig. 6 Comparison of Image qualities before and after an optimization of LPF of DCT coefficients

5 结 论

经过以上步骤的JPEG编码和优化,在本平台上完成一帧720×576大小的YUV4:2:0格式的彩色图像的JPEG压缩编码需要22 ms左右,压缩后大小约为45 KB,图像质量主观评价良好,如图6所示.由摄像头采集到的原始图像大小为720×576×1.5Bytes=607.5 KB,经本系统的JPEG编码后的压缩比约为13.5:1.测试证明,本系统的编码速度和质量完全满足车牌识别系统的需要.本设计所使用优化方法对其他C6000系列DSP处理器的程序优化,以及更为复杂系统的优化具有一定借鉴意义.

参考文献:

[1] 刘珂含,何培宇,等.基于TMS320VC5509A的图像采集与识别系统[J].四川大学学报:自然科学版,2008,45(1):45.

[2] Texas instruments incorporated. TMS320C6000系列DSP的CPU与外设[M]. 北京:清华大学出版社,2007.

[3] 何小海,等.数字图像通信及其应用[M].成都:四川大学出版社,2006.

[4] 王华斌,丁刚.基于DSP的视频算法系统优化若干策略[J/OL].美国:德州仪器,2006(2006-09-01)[2008-03-26]. http://focus.ti.com.cn.

[5] Texas instruments incorporated. C Implementation of the TMS320C62xx Intrinsic Operators [DB/OL]. USA: TI, 1999. (1999-12-15)[2008-03-26]. http://www.ti.com.

[6] 程凌峰,黄本雄. TMS320C64X DSP的程序设计与优化[J].今日电子,2004,05:91.

[责任编辑:李富河]