文章编号:1671-4598(2018)04-0221-04 DOI:10.16526/j.cnki.11—4762/tp.2018.04.058 中图分类号:F832 文献标识码:A

基于 ARM 平台的高清视频信号编解码器优化设计

张大禹
(中国人民解放军 92124 部队,辽宁 大连 116000)

摘要:由于原始编解码器存在编解码速度慢,耗费时间长的问题,无法满足人们需求,提出了基于 ARM 平台的高清视频信号编解码器优化设计;在 ARM 平台下对接口进行分离,从代码和结构两方面对编码器进行精简,进而对相应程序进行优化;针对解码器优化过程,优化 C 级别和熵解码,并移除多余结构体以及冗余函数,重新建立哈夫曼查表,完成高清视频信号快速解码;通过实验验证结果可知,优化后的编解码器速度快,且耗费时间较短。

关键词:ARM 平台;高清;视频信号;编解码器;优化

Optimization Design of High Definition Video Signal Codec Based on ARM Platform

Zhang Dayu
(No. 92124 Troops of PLA, Dalian 116000, China)

Abstract: Because the original codec is slow and time-consuming, it can't meet the needs of people. A design of HDTV codec based on ARM platform is proposed. Separation of the interface in the ARM platform, streamline the encoder from two aspects of code and structure, and the corresponding program is optimized; for the decoder optimization process, the optimization of C level and entropy decoding, and remove the redundant structure and redundant function, re establishment of Huffman table, complete HD video signal fast decoding. The experimental results show that the optimized codec has a fast speed and a shorter time consuming.

Keywords: ARM platform; HD; video signal; codec; optimization

0 引言

高清视频技术是多媒体技术中的关键技术,多用于无线通信领域和计算机系统中[1]。多媒体设备从最初单一分辨率到如今的多分辨率立体视频,从而也对视频存储提出了较高要求,采用立体视频能够形象记录,尽管所需存储空间不断变大,但是相对记录的内容依旧不能满足用户需求,在这种情况下,视频压缩就显得尤为重要[2]。面临有限信道的存储空间,需要尽量压缩数字视频存储空间,使用较低比特率传输高质量视频。多媒体技术在当代人们生活中所占比重越来越高,更多移动终端被广泛应用,随着系统更新架构不断推出,越来越多性能较好的系统平台被采用[3]。基于 ARM 平台是 ARM 公司新推出的平台架构,其特定的处理器内核和微处理器具有较强兼容性,为编解码器的编码和解码提供良好环境。

由于原始编解码器存在编解码速度慢,耗费时间长的问题,无法满足人们需求[4]。为此,提出了基于 ARM 平台的高清视频信号编解码器优化设计。通过对编解码器的研究,以及 ARM 平台分析,可充分了解编解码流程和各个模块功能,同时也对 ARM 平台处理器机制进行了更加深入的了解,由此可对编码和解码各个功能机制进行优化,通过对比实验可知,优化后的编解码器编解码速度快,且耗费时间较短,能够满足人们需求。

1 高清视频信号编解码器在 ARM 平台上的优化

随着我国多媒体技术快速发展,视频技术得到广泛应用,ARM 平台具有性能强大 ARM 微处理器,并带有 NEON 多媒体处理器,针对高清视频信号的编解码器在该平台上进行优化是安全的,在整个优化过程中,需要采用多种技术,将数据对齐,分别对编码器和解码器进行优化,能够在很大程度上地提高编解码器整体性能[3]。

1.1 编码器优化设计

1.1.1 高清视频信号接口分离

ARM 微处理器具有出色的性能和效率,能够适用于各种移动和消费类应用,由于处理器是根据不同系统性能而实现的,为用户苛刻性要求提供了较高编码性能,并在该性能条件下移动设备功率消耗不到 300 mW[5]。在该平台下对编码器进行编码可大幅度提高编码能力,同时可保持移动设备高功率水平。由于原始编码器中保留着现场可编程门列(FRGA)模块,为此,首先需在程序中对主要接口分离,将属于现场可编程门列(FRGA)进行拆分,通过对不同函数的构建可在核心源程序中,构建 7 个不同种类的函数,并对其功能进行描述,如表 1 所示。

表 1 核心编码函数及其功能

编号	函数名称	功能
1	Y300 * macroblock * cache * load	宏块编码获取
2	Y300 * macroblock * Forward prediction	前向预测
3	Y300 * macroblock * Backward prediction	后向预测
4	Y300 * macroblock * write * cavlc	熵编码获取
5	Y300 * macroblock * cache * save	目前编码信息储存
6	Y300 * reference * update	重构编码
7	Y300 * encode	编码分层

由表 1 可知,现场可编程门列(FRGA)模块主要是对帧间和帧内进行预测,促使残差块掉落,并对帧内部结构进行重新构造,这就包括了帧间和帧内的预测功能。根据其他编码宏块所获取的信息进行预测,在帧间出现的模块信息,为下一个宏块编码提供有效依据,并将信息完整保存,为下一个宏块编

收稿日期:2018-01-18; **修回日期**:2018-02-17。

作者简介:张大禹(1977-),男,黑龙江哈尔滨人,高级工程师,主要从事高清视频信号传输与处理方向的研究。

码提供参考。该部分功能主要是由上述构建函数而实现的，为此，完整帧内部预测模块函数应包括 Y300 * macroblock * cache * load、Y300 * macroblock * Forward prediction 和 Y300 * macroblock * cache * save 三个函数，针对滤波处理，需由 Y300 * reference * update 函数来实现。

针对编码变换、熵编码功能实现，需由 Y300 * encode、Y300 * macroblock * write * cavlc 和 Y300 * nal * encode 三个函数实现，因此，在上述函数中，Y300 * macroblock * cache * load、Y300 * macroblock * Forward prediction 和 Y300 * macroblock * cache * save 以及 Y300 * reference * update 被划分到功能模块当中[6]。在 ARM 平台下的程序结构中，各个函数之间的关系是具有独立性的，不会因此而产生依赖性，因此，在面对宏观模块时，进行预测需将目前宏观模块和预测模块同时当作输入变量，将输入信号作为预测编码，如果当前的宏观模块与预测模块同时进行优化，那么只有残差发生变化时，所得的函数才能包含所有功能，为此需对函数进行拆分，具体拆分过程如下所示：

1）将预测部分从函数中提取出来，采用两个函数和两个全局数组方式，名称和功能如表 2 所示，在函数中需根据当前预测模块所包含的所有高清视频编码中的亮度与色度信号，调用函数，将获取的残差数据全部存入全局数据组中，作为 DCT 变换输入[7]。

表 2 函数和全局数组名称功能描述

名称	功能描述
Y300 * macroblock * predict	宏块预测
Pixel * diff * wxh	像素分辨率差异识别
int15 * ta15 * brightness [15 * 15]	存储亮度信号数据
int15 * ta15 * chroma [8 * 16]	存储色度信号数据

2）在源程序中修改函数，并逐级调用，促使输入的模块为当前宏观模块的预测信息，经过上述一系列高清视频信号接口分离后，输入残差数据，相应函数参数也要进行一定修改。经过调整后的函数被内部程序调用了，那么可降低程序修改量，直接将残差数据输入，为下一步优化提供准确数据。

1.1.2 编码器代码精简

经过上述高清视频信号接口分离后，获得了准确输入数据，将此数据作为基本数据进行编码器档次调整，因此对编码器代码精简是具有必要性的。删除基本档次，将不需要模块去除，如果将编码器内部基本帧内容去除后，那么剩下的档次只有 I 帧和 P 帧，去除 B 帧之后，不能进行实时编码，为此，删除 B 帧可省去后向预测繁琐编码过程[8]。进行熵编码时，需将自适应的二进制算术编码应用到不同档次的实时编码过程当中，由于编码器只支持熵编码，而不支持熵编码，为此删除与熵编码有关的所有程序语句。

1.1.3 编码器逻辑结构精简

在对高清视频帧数进行判定时，由于对代码进行了精简，为此在编码过程中省去了后向预测繁琐编码过程。在代码精简后的编码器中首先对其它帧进行编码，将输入视频数据序列进行多次空间转移以及重新排列，空间存储转移过程如图 1 所示。

由图 1 可知：在读入视频数据时，首先应该先保存 Y300 _ picture 过程，然后将数据拷贝下来，同时向边界方向进行扩展，经过扩展后的视频宽度变为 32 像素，那么主要用于数据预测和视频方向预计[9]。确定预测帧方向后，将读入顺序与

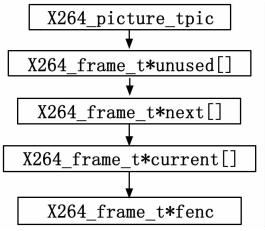

图 1 高清视频帧存储空间转移过程

输入顺序进行对比，如果输入顺序视频帧数一定数值后，可进行重排，对读入视频帧数序列进行类型判断，最后，对 current 序列中的视频进行编码。

针对目前函数中存在的序列，经过种类判定后，序列需再次排列，将 B 帧存在在 P 帧之后，然后对函数进行重新调用，随后按照当前帧顺序完成编码。针对 I 帧种类的判断，可对 P 帧进行编码，当编码完成后，需对 P 帧进行预测，预测后的数据是需要修正的，由此可知，如果编码时含有 B 帧，那么 B 帧种类判定结果比较复杂；如果编码时没有 B 帧，那么实际帧序列为 IPPP…，为此，需对部分逻辑进行简化，去掉 current 序列，需对每次读入的帧数进行直接判定，但需保留 I 帧判定方式，优化后的编码器帧判断类型如图 2 所示。

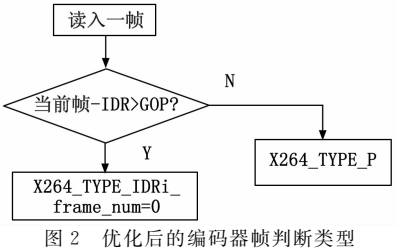

图 2 优化后的编码器帧判断类型

1.2 解码器优化设计

在整个解码器优化的过程中，需利用内联函数将数据对齐，并采用多重优化技术将解码器内部进行优化，该方式能够提高解码器性能，而且针对解码速度也具有一定影响。采用汇编级别优化方式，尤其是在 ARM 微处理器平台下能够并行处理，基于此，提出了针对并行处理机制的优化方法，在较大程度上能够提高整体性能。经过上述内容能够获取不同形式下的函数，针对热点函数需对其进行优化，以此为基础提高高清视频编码器整体性能。

1.2.1 优化 C 级别

在 C 级别优化策略中，需对解码器运算性能进行优化，具体优化过程如下所示：

由于内联函数是函数调用的关键方法，采用该方法不但可以节省函数调用时间，还能使解码整体达到高效率标准，通过对全部函数分析可知，经常被使用的函数是解码器在优化时最好的内联纽带，一般情况下，只要小于等于 5 个信号发出的指令，那么周期函数是需要通过内联纽带才能完成优化的，具体优化方式有：1）保密内联：通过编译器对全部函数进行校验，通过验证的函数进行内联，然后编译选项，利用内联数据进行选择来完成优化；2）显示内联：在函数中使用关键字进行优化，在头文件中的定义添加关键字，防止在连接过程中对定义重复设置；3）人工内联：通过人工方式，对有关性能影响较大关键函数进行定义替换，完成相应部分调用。

1.2.2 移除多余结构体以及冗余函数

在原始解码过程中，难免会存在各种各样的冗余结构，为

此在优化时，需将冗余结构的变量体去除，或者重新设置内存单元来减小内存占用情况，以此为基础提高解码运行效率。在 ARM 平台下进行解码，此时需要调用函数，以此为基础调回更多数据来填满缓存单元。通常情况下，原始缓存数据大小为 128 字节，在小的缓存单元中会出现频繁函数调用情况[10]，此时需对信号解码状态进行读取，如果调回的函数消耗时间较小，那么提高解码器性能的几率就会大大提高。

根据上述内容，在代码优化过程中，将调回函数数据缓存大小更改为 1.5 K 个字节，该种优化方式能够有效提高解码整体性能，在解码器使用过程中，利用上层解码数据将解码帧储存到模块中，然后将上层数据通过回调函数方式进行数据收集，进而将整体回调函数去除，该步骤能够加快解码器的解码速度，进而大大提高解码器对高清视频信号解码的效率。

1.2.3　优化熵解码

一般情况下，利用哈夫曼解码方式作为熵解码[11]，可将解码过程变得更加直观化，通常采用二叉树的方式呈现，如图 3 所示。

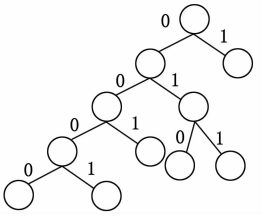

图 3　二叉树解码

查表方式如表 3 所示。

表 3　熵解码符号、码字、码长对应表

符号	码字	码长	概率
a	K	1bit	0.8
b	1k	2bit	0.4
c	11k	3bit	0.2
d	111k	4bit	0.1
e	1111	4bit	0.05

由表 3 可知，解码过程查表可由以下几个表组成，每次进行解码时可读取一个 bit，剩下数据可通过如下方式进行。当读取比特为 1 时，符号为 a。当读取比特为 0 时，需向下跳转：当读取比特为 1 时，符号为 b。当读取比特为 0 时，需向下跳转；当读取比特为 1 时，符号为 c。当读取比特为 0 时，需向下跳转；当读取比特为 1 时，符号为 d。当读取比特为 0 时，符号为 e。该表一共分为 14 个等级，针对每一个等级，进行读取的比特数分别为 1－1－1－1－1－1－1－0－0－0－0－0－0－0。通过完成解码后获得的符号，进行一次性码流最大字节比特数输入，然后将数据进行一一比较进行解码，可获得解码时所耗费时间，如果读入 1 个比特数据，如果比特为 0，那么需要根据上述方式继续查找，如果达到哈夫曼叶节点则说明解码成功。采用这种优化方法，重新建立哈夫曼查表，每一个级别的查找都能快速完成，优化后的查找等级，平均搜索时间较快，能够满足人们快速解码需求。

2　实验

为了验证基于 ARM 平台高清视频信号编解码器优化设计有效性进行了如下实验。该实验选取某高清视频编解码器中的一些典型样本进行测试，在 ARM 平台上利用仿真工具验证优化后结果。

2.1　编码器优化性能验证

将初步移植到 ARM 平台上编码器整体性能进行测试，采用同一视频不同分辨率方法，对复杂画面序列进行测试。其中有关视频名称和内容的描述如下所示：

1）Foreman * cif. yuv：该序列表示视频中的人物与背景纹理复杂程度一致，如果发生剧烈运动，那么视频中的镜头会处于剧烈摇晃状态下；

2）Silent * cif. yuv：该序列表示视频处于静镜头环境下，人物与背景纹理不一致，复杂程度相对高；

3）Container * cif * yuv：该序列表示视频中的人物与背景纹理复杂程度一般，镜头集中且运动相对缓慢。

4）Bigships * 1024 * 768. yuv：该序列表示视频背景与人物视频中运动纹理复杂程度一致均较高，运动相对缓慢；

5）Jet * 1024 * 768. yuv：该序列表示视频背景与人物运动纹理复杂程度一致均一般，此时镜头是可移动的。

利用 CCS 的 profile 工具，分别对序列进行统计，统计结果如表 4 所示。

表 4　编码器序列统计结果

分辨率	序列名称	帧率/fps
442×133	Foreman * cif. yuv	0.70
	Silent * cif. yuv	0.73
	Container * cif * yuv	0.77
520×476	Foreman * 520 * 476. yuv	0.10
	Container * 520 * 476. yuv	0.09
910×768	Bigships * 910 * 768. yuv	0.13
	Jet * 910 * 768. yuv	0.09

为了使性能验证结果更加具有可靠性，将原始解码器与本文优化后的解码器序列情况进行对比，结果如表 5 所示。

表 5　两种设计方法序列测试结果对比

序列		原始解码器		优化后解码器	
		总时钟数	帧率/fps	总时钟数	帧率/fps
442×133	Foreman	1349304834	8.67	199423586	70.62
	Silent	1443482343	8.71	206848448	68.15
	Container	1324673400	9.52	194447913	72.36
520×476	Foreman	5912216229	1.48	694204481	20.29
	Container	5538519981	1.65	718989170	19.57
910×768	Bigships	10155546792	0.32	1285266061	9.82

由表 5 可知：由于每个测试的序列长度均为 25 帧，相比于原始编码器中的序列初始统计结果，优化后的编码器平均编码速度提高了约 12 倍。

2.2　解码器优化性能验证

不同解码率在优化后的性能统计结果如表 6 所示。

表 6　不同解码率性能优化结果

测试样本	采样率/Hz	平均解码时间/s	平均性能/MHz
高码率	43109	46390	6.53
中码率	43109	108057	6.50
低码率	43109	102635	6.04
普通样本	43109	187564	5.05

根据表 6 中解码器优化性能，将原始解码器与优化后的解码器平均解码时间进行对比，结果如图 4 所示。

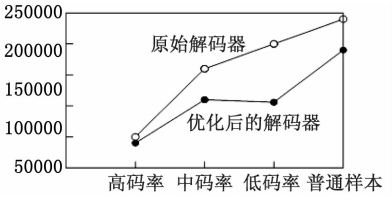

图 4 两种方法解码时间对比结果

由图 4 可知在高码率条件下的原始解码器耗费时间为 50101 s，而优化后的解码器平均解码时间为 46390 s；在中码率条件下的原始解码器耗费时间为 168153 s，而优化后的解码器平均解码时间为108057 s；在低码率条件下的原始解码器耗费时间为 200132 s，而优化后的解码器平均解码时间为 102635 s；在普通样本条件下的原始解码器耗费时间为 240134 s，而优化后的解码器平均解码时间为 187564 s。

2.3 实验结论

根据上述分别对编码器优化性能和解码器优化性能进行验证，可得出实验结论：原始编解码器无论是编码速度还是解码速度都远远小于优化后的编解码器，且时间也是比优化后编解码器耗费较多，由此可知经过优化后的编解码器平均耗费时间较短，编解码速度较快。

3 结束语

通过对编解码器研究，以及 ARM 平台分析，可充分了解编解码流程和各个模块功能，同时也对 ARM 平台处理器机制进行了更加深入了解。分别对编码和解码各个功能机制进行优化，可充分提高编解码器整体性能，在完成针对编解码模块优化后，整体性能有了较大程度提高，并通过实验证明，优化后编解码器无论是编码速度还是解码速度都比原始编码器快，且耗费时间短，可充分保证其整体性能的稳定性，更有针对性地完成优化工作。

参考文献：

[1] 张 晗，段鹏松. 基于通信编码的视频监控无损控制优化设计 [J]. 科技通报，2015，31 (12)：209－211.

[2] 武为江，蒲 涛，朱华涛，等. 基于波长选择开关的 OCDMA 编解码系统实验研究 [J]. 中国激光，2016，20 (1)：116－121.

[3] 陈天恒，杨晓静，王伟力，等. 基于蚁群算法的变电站视频监控联动方案优化设计 [J]. 电力系统保护与控制，2016，44 (2)：134－139.

[4] 孙 文，全惠敏，吴桂清，等. 基于 ARM 和以太网接口的振动信号采集系统设计 [J]. 电源技术，2015，39 (9)：1963－1964.

[5] 刘隆华，黄洪全，黄启哲，等. 基于电弧外特性的故障定位信号发生器优化设计 [J]. 电力系统自动化，2016，40 (9)：100－105.

[6] 王风燕，方华丽，刘志刚. 基于信号白噪声处理的玉米精密排种器结构优化设计 [J]. 农机化研究，2016，38 (7)：30－34.

[7] 刘艳飞，郭 军. 基于三值光学计算机解码器信号判定系统的设计研究 [J]. 激光杂志，2016，37 (8)：54－56.

[8] 龚 裕，朱海业，李 楠. 基于正交试验方法的相邻电容传感器优化设计 [J]. 北京工业大学学报，2015，19 (1)：32－36.

[9] 时维元，林正英，陈希信，等. 线性调频信号低旁瓣脉压窗函数的优化设计 [J]. 现代雷达，2015，37 (10)：18－20.

[10] 周 军，李 娟，王庆丰，等. 基于自发辐射抑制的红外光机系统优化设计 [J]. 光学学报，2015，35 (3)：278－285.

[11] 龚杨阳，安军社，朱 岩. CCSDS 标准下低码率 LDPC 码的编码器设计 [J]. 电子设计工程，2017，25 (5)：57－60.

(上接第 220 页)

表 4 模型测试 %

评值 / 数值 / 特征选取－算法	正确率	召回率	$f-score$
Word2vec－SVM	79.61	78.45	78.99
词对向量－LSTM	78.17	76.42	77.42
MHMM－LSTM	79.65	75.23	77.52
随机向量－LSTM	75.32	74.28	74.22
Word2vec－LSTM	78.52	77.46	76.04

量作为 LSTM－RNN 细胞的输入，进行情感分类，通过多轮迭代产生，对其正确率、召回率和 f－score 进行评估，实验结果表明了通过 MHMM 进行预训练产生的词对向量，用于 LSTM－RNN 分类器中，其性能优于空间相邻的词对向量，而词对向量的性能优于 word2vec。

综上观之，运用 MHMM 进行特征抽取，进行情感分类的作用效果较好。

参考文献：

[1] Tsou B K Y, Yuen R W M, Kwong O Y, et al. Polarity classification of celebrity coverage in the Chinese press [A]. Proc. of International Conference on Intelligence Analysis [C]. 2005.

[2] Pang B, Lee L, Vaithyanathan S. Thumbs up?: Sentiment classification using machine learning techniques [A]. Proc of the ACL Conference on Empirical Methods in Natural Language Processing [C]. Association for Computational Linguistics, 2002: 79－86.

[3] Pang B, Lee L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales [A]. Pro. of the 43rd Annual Meeting on Association for Computation Linguistics [C]. Association for Computational Linguistics, 2005: 115－124.

[4] WhtitelawC, GargN, ArgamonS. Using appraisal groups for sentiment analysis [A]. Proc of 14th ACM International Conference on Information and Knowledge Management 2005: 625－631.

[5] Taboada M, Brooke J, Tofiloski M, et al. Lexicon－based methods for sentiment analysis [J]. Computational Linguistics, 2011, 37 (2): 267－307.

[6] Nigam, McCallumA, ThrunS, et al. Learning to classify text from labeled and unlabeled documents [A]. Proceedings of the 15th National /10thConferenceon Artificial Intelligence/Innovative Applications of Artificial Intelligence [C]. Menlo Park, CA, USA: AAAI Press, 1998: 792－799.

[7] Lee Y. Dernoncourt: sequential short－text classification with recurrent and convolutional neural networks (2016). arXiv preprint: 2016.

[8] 李 锐，张 谦，刘嘉勇. 基于加权 word2vec 的微博情感分析 [J]. 通信技术，2017 (3)：502－506.

[9] ChiLu et al, A P－LSTM Neural Networksfor Sentiment Classification [J]. Jinho Kim . KyuseokLongbing Cao Jae－Gil Lee XueminLin. Yang－Sae moon (eds): 105－110.