第32卷 第23期 计 算 机 工 程 2006年12月
Vol.32 No.23 Computer Engineering December 2006
·开发研究与设计技术· 文章编号：1000—3428(2006)23—0268—03 文献标识码：A 中图分类号：TN911.73

基于嵌入式处理器的H .264编码器的存储优化

徐 宁，史 册，陈梅丽
（浙江大学信息与电子工程学系，杭州 310027）

摘 要：由于H.264/AVC新标准采用了很多新技术，在可编程处理器的应用领域中，如果不进行优化将会需要非常大的存储空间。该文对编码器的存储复杂度进行了分析，在此基础上提出了基于宏块级的滤波和插值算法。为了便于嵌入式处理器的实现，提出了一种高效的内存管理调度策略。实验结果表明，优化方法在极大地降低存储复杂度(cycle:64.9%)的同时得到了更高的编码速率(76.6%)，而只有很小的编码效率损失。

关键词：H.264；嵌入式处理器；存储优化

Memory Optimization Scheme of H.264/AVC Encoder Based on Embedded Processor

XU Ning, SHI Ce, CHENG Meili
(Dept. of Information & Electronic Engineering, Zhejiang Univ., Hangzhou 310027)

【Abstract】The newly introduced techniques adopted by H.264/AVC encoder requires large memory space for a programmable processor implementation without optimization. An improved algorithm of mb-level (Macroblock-level) interpolation and deblocking filter are proposed for H.264/AVC encoder based on memory usage analysis of the reference software (JM8.4). In order to implement the encoder on an embedded processor, an efficient memory management scheme is also presented. Experimental results show the approach can greatly improve encoding speed (74.9%) and reduce memory complexity (cycles: 68.2%) with negligible coding efficiency loss.

【Key words】H.264/AVC; Embedded processor; Memory optimization

随着一系列的数字视频压缩技术在数字电视、DVD电影和网络流媒体等方面的广泛应用，MPEG(Moving Picture Experts Group)和ITU-T视频编码组(Video Coding Expert Group)联合开发了新一代的视频压缩标准H.264/AVC。该标准提供了更强的压缩性能和友好的网络视频接口，主要应用于实时双向交互和非交互应用领域。同以往视频标准类似，AVC仍然采用了基于运动补偿的DCT变换的结构框架，并在此基础上增加了更多的新技术(例如，亚像素预测、可变块大小、基于拉格朗日的编码控制以及多帧参考等)，能够节省大约50%的码率，但同时复杂性也是MPEG-2的4倍、MPEG-4的Visual Simple Profile的2倍[1]。这样在H.264/AVC硬件实时应用过程中，巨大的数据需求和计算复杂度对硬件环境提出了更高的要求。如何解决这二者之间的矛盾，成为的硬件应用的关键。

通常，硬件应用采取两种方法：ASIC (Application Specific Integrated Circuit) 和嵌入式处理器。其中基于ASIC的应用具有成本高、设计周期长以及不易升级的缺点。相对于传统的集成电路设计，嵌入式处理器的设计更加灵活、成本较低、周期较短，已经越来越多地被采用到实时应用领域。

文献[2]从算法和结构上对编码器进行了优化，在插值过程中利用TIS320c6414DSP特有的多媒体指令一次读入多个数据，降低了数据调用次数，但仍需在内存中维持整个参考帧的大量数据。文献[3～5]对编码器的帧间预测、环内滤波器和熵编码的硬件设计进行了研究。本文针对普遍的硬件特点，提出了基于宏块的插值和滤波算法及一种高效的内存管理调度策略。可被广泛应用于不同硬件环境下的编码器实现。

1 基于嵌入式处理器H.264/AVC编码器优化

1.1 H.264/AVC编码器复杂性分析

在H.264标准中没有明确定义编码器的构成，而只定义了编解码端码流的语法结构。除了滤波器(去块效应滤波器)以外，绝大多数基本函数组成(预测、变换、量化、熵编码等)，都与以往视频编码标准类似，新标准对其中一些功能模块进行了许多重要改变。

H.264标准是面向广泛应用领域的新一代视频标准，为了便于应用实现，标准[5]中定义了profiles和levels两个参数用来限制编解码器。其中，profiles (baseline profile、main profile、extended profile)用来限制码流的语法结构。同时在每一级profile中定义了不同level等级，通过level对码流中语法结构的数值进行设置，对应不同应用的实际要求。

表1给出了baseline profile条件下，不同level值对应的存储需求。其中参数MaxFS指的是每一帧中宏块(MB)的数目。本文以baseline profile的编码器为讨论对象，level等于1，图像格式为QCIF。由于采用7种可变块大小运动补偿和多帧参考等技术，可以在超过一个参考帧中搜索与当前编码更加匹配的宏块或块，更大限度地去除空间和时间上的图像冗余信息，以达到更高的压缩性能。但同时，也带来了更大

作者简介：徐 宁(1978－)，男，硕士生，主研方向：视频压缩与图像处理；史 册，副教授；陈梅丽，硕士生

收稿日期：2006-02-17 **E-mail：**xuning0115@tom.com

的存储需求和计算复杂性。为了进一步增强运动估计的搜索精度，H264 在运动补偿中采用了 1/4 和 1/8 像素。

表 1 不同 level 值的内存需求

Level 值	MaxFS (MBs)	Level 值	MaxFS (MBs)	Level 值	MaxFS (MBs)
1	99	2.1	792	4	8 192
1.1	396	2.2	1 620	4.1	8 192
1.2	396	3	1 620	4.2	8 192
1.3	396	3.1	3 600	5	22 080
2	396	3.2	5 120	5.1	36 864

其中采用 6 阶 FIR 滤波器的内插获得 1/2 像素值，并在此基础上通过线性内插得到 1/4 像素值，相应对色度内插得到 1/8 像素值。通过 1/4 像素的搜索，可得到更加精确的运动矢量，但同时编码器需要更多的内存空间用来存储参考图像。

本文以 baseline profile 的编码器为讨论对象，图像格式为 QCIF。在这个条件下，存储一帧原始图像和一帧参考帧各需要 38 016B(99×16×16×1.5＝38 016)的空间，水平、垂直和对角方向的半像素值分别需要 38 016B，运动估计总共需要 152 064B(38 016×4＝152 064)的空间。

1.2 基于嵌入式处理器的 H.264/AVC 编码器的存储优化

通常，嵌入式处理器的内存结构分为两个部分：片内内存和片外内存。片内内存能够以全处理器速度进行存取；片外内存提供额外的容量，能够运行在处理器全速带宽上，但比片内内存的响应时间要稍微长一些。

我们以AD (Analog Devices) 公司的Blackfin533 DSP为例[6]，从片外向片内传送一个 32bits数据平均需要 46.3 个时钟周期，而仅仅在片内的情况下只需要 1 个时钟周期。因此在设计程序时，最重要的是将程序和数据尽量放在片内内存中。但是，不幸的是很难将全部的数据存放在片内，特别是在使用多帧参考时。

在基于嵌入式处理器的实时应用中，如何降低 H.264/AVC 编码器的存储开销是一个非常关键的问题。本文从以下两个方面考虑：

(1)合理分配原始图像和参考帧的存储空间

根据上面的存储复杂度分析，将部分原始数据和参考数据放置在片内，并在原始图像和参考窗的数据传输中采用双缓存机制。这时只需在片内开辟 768(16×16×1.5×2)B 存放原始图像数据，6 912 (48×48×3/2×2) B 用于整像素参考窗。其余暂时不使用的数据放置在片外，需要时再搬运到预先分配好的片内位置。

(2)采用宏块级的插值和滤波方法代替原有的帧级插值和滤波方法

图 1 为帧级插值(左侧)和宏块级插值(右侧)的流程图。

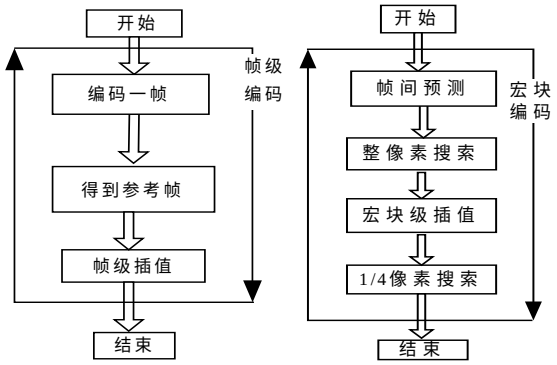

图 1 帧级和宏块级插值算法流程

对于帧级插值过程的调用，水平、垂直和对角线 3 个方向的半像素值，是在得到整个参考帧后进行。这样当我们采用一帧参考时，存储半像素值需要 114 048(176×144×1.5×3)B 的空间，这是很难完全放置在片内的。

为了解决这个问题，我们提出了宏块级的插值算法来代替原来的帧级插值算法。这样就可以在得到每个参考宏块之后就进行插值。同时，为了进一步降低内存开销，可以将 3 个方向的半像素值共用一个子参考窗口，它的大小可以按下面的公式进行计算：

$$size = size_{sw} \times size_{sw} \times 3/2$$

其中，size为子参考窗的大小；$size_{sw}$ 为搜索窗的大小。当搜索窗的大小限制在 48 个像素值时，在片内仅仅需要开辟 10 368(48×48×3/2×3)B的空间。

类似于上面的插值算法，相应地将帧级滤波修改为宏块级的滤波。帧级滤波需要在内存中存储整个重建帧，而宏块级的滤波可以在得到重建宏块之后马上进行。当前宏块的滤波需要用到其左侧和上侧的宏块数据，只需在内存中维持 13 个宏块的空间(图 2), 其中宏块 1～11(QCIF 格式)为重建帧的一行宏块数据。在对第 13 个宏块进行滤波时，不再使用第 1 个宏块的数据，可以将它的数据通过 DMA 搬运到片外参考帧地址。以此类推，在宏块级的滤波过程中，只需要在片内维持 12 个宏块的空间而不是一帧(99 个宏块)的空间 ，节约了 88%的空间开销。很显然，宏块级滤波需要的空间要远远小于帧级滤波。

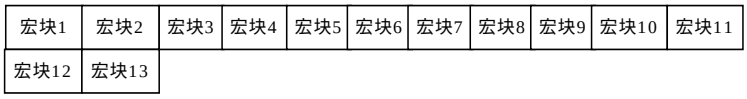

图 2 宏块级滤波

2 一种高效的基于嵌入式处理器的内存调度机制

我们提出了一种高效的内存管理调度机制。从表 3 的统计数据中发现 H.264/AVC 编码器需要在片内和片外间进行大量的图像数据移动。因此，在基于 DMA 的内存调度中，最重要的就是要充分挖掘数据的并行性来实现更高的数据处理能力。图 3 中的 DMA 初始化函数用来初始化 DMA 读通道和写通道。对于编码过程中很多不变的参数，例如数据传输的地址、数量等，可以放在这个函数中进行初始化。

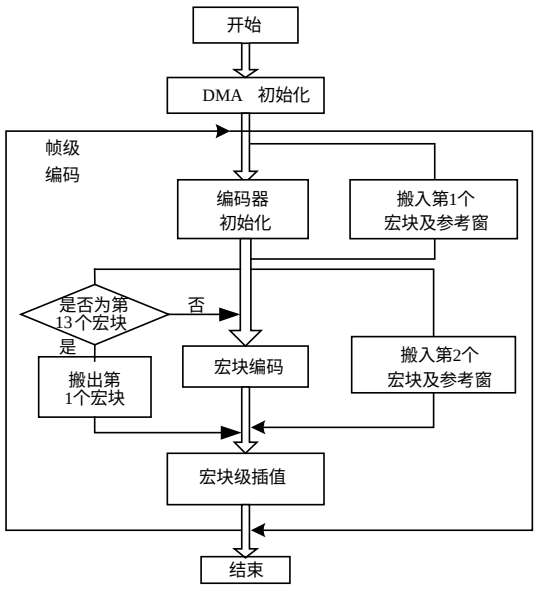

图 3 内存管理策略

在初始化编码器的同时，使能 DMA 读通道，传输第 1 个宏块及其对应的参考窗到片内。在编码当前宏块的同时，传输第 2 个宏块及其对应的参考窗到片内 2st Mb)。在进行第 13 个重构宏块(Mb＝13)的同时，使能 DMA 写通道，将经过滤波的第一个宏块传输到片外参考帧的地址(函数 DMAo 1st Mb)。在片内开辟了双宏块缓存池和双参考窗缓存池，将原始图像和参考窗的传输交替组成一个 DMA 传输链(图 4)。

第1个 宏块数据	第1个 参考窗	第2个 宏块数据	第2个 参考窗	...

图 4 DMA 读通道传输链

具体调度流程如下：

(1)在初始化编码器的同时，使能 DMA 读通道，传输第 1 个宏块及其对应的参考窗到片内。

(2)在编码当前宏块的同时，传输第 2 个宏块及其对应的参考窗到片内。

(3)在进行第 13 个重构宏块的同时，使能 DMA 写通道，将经过滤波的第 1 个宏块传输到片外参考帧的地址。

3 实验结果

我们以 ADSP-BF533 为平台，检验了上文提出的算法和内存调度策略。ADSP-BF533 DSP 是美国 ADI 公司一款 16 位定点的 DSP。Blackfin533 的存储结构由两部分组成：L1(片内内存)和 L2 (片外内存)。其中片内内存又分为片内代码(L1-code)和片内数据(L1-data)两部分。

表 2 给出了基于 BF533，编码器的内存分配方案(一帧参考)。其中程序段完全被放置到 L1-code 中，应用数据被分别放置到 L1-data 和 L2 中。

表 2 编码器存储分配情况(ADSP BF533)

内存	片内		片外
	L1-code	L1-data	L2
占用 (bytes)	81 908	42 244	114 080
总容量 (bytes)	80k	64k	256M
Rate (%)	99.98%	64.46%	0.04%

表 3 给出了基于 BF533 不同序列数据传输消耗的时钟数以及比例。

表 3 数据传输复杂性分析表比较(ADSP BF533)

测试序列	数据读入		数据写出	
	时钟周期	%	时钟周期	%
carphone	509 096 946	28.84%	26 872 105	1.52%
foreman	509 120 967	30.57%	26 569 079	1.61%

表 4 给出了基于 PC 平台，帧级和宏块级插值和滤波算法的性能比较。

从表 4 中的数据发现，宏块级插值和滤波算法的引入，使编码器的性能有了一定的下降。表 4 显示的是基于 ADSP BF533 平台，传输数据的时钟数统计，说明数据的传输在整个编码过程中占用很多的时钟周期。

表 4 帧级与宏块级算法的性能比(pc)

测试序列	帧级		宏块级	
	峰值信噪比 psnr（db）	帧率 fps	峰值信噪比 psnr (db)	帧率 fps
Foreman	35.25	6.48	34.00	6.22
Carphone	35.30	6.32	35.22	6.33
Sales-man	34.91	7.71	34.87	7.87
claire	42.40	7.50	41.94	7.70

表 5 给出了基于 ADSP BF533 平台，采用宏块级插值、滤波算法和 DMA 调度策略前后的性能比较。

表 5 优化前后性能比较(ADSP BF533)

测试序列	优化前		优化后	
	帧率 fps	时钟周期 cycles	帧率 fps	时钟周期 cycles
Carphone	4.36	1 765 399 929	7.62	1 229 428 925
Foreman	3.95	1 665 521 121	6.84	1 129 531 105
Sales-man	4.29	1 745 895 510	7.58	1 211 589 645
Chaire	5.39	2 085 462 934	9.43	1 521 961 852

由上面图表的数据分析中发现，尽管采用宏块级的插值和滤波算法会对编码器的性能下降平均 0.46dB(表 3)，但是基于 ADSP BF533 平台的总时钟周期数，平均下降了 31.8%，同时编码帧数提高了 74.89%。以上实验基于以下条件：

(1)总帧数(total frame number) 30 帧，其中 2 个 I 帧，28 个 P 帧。

(2)ADSP BF533: 时钟周期为 270MHz。

(3)代码：JM8.4。

4 结论

本文给出了基于嵌入式处理器平台 H.264/AVC 编码器的存储复杂度分析。根据实时应用的要求和硬件限制，提出了宏块级的插值和滤波算法。并且在此基础上提出了基于硬件实现的一种高效的内存管理调度机制。实验结果表明在基于嵌入式实时应用条件下，新算法和调度机制的采用极大提高了编码器的性能。同时本文中编码器的算法分析和优化方法也适用于多核处理器的应用，并且能够得到更好的编码性能。

参考文献

1 Borman J. Video Coding with H.264/AVC: Tools, Performance, and Complexity[J]. IEEE Circuits and System, 2004, First Quarter: 7-28.

2 刘根林. 基于 TMS320C346416 芯片的 H.264 编码器的设计和实现[J]. 电视技术, 2004, (10): 1-3.

3 Wang Yuehyi, Peng Yantsung, Tsai Chunjen. VISL Architecture Design of Motion Estimation and IN-loop Filter for MPEG-4 AVC/H.264 Encoders[C]. Proc. of IEEE ISCAD'04, 2004-05-23: 49-52.

4 Huang Yuwen, Hsieh Bingyu, Chen Tungchien, et al. Hardware Architectrure Design for H.264/AVC Intra Frame Coder[C]. Proc. of IEEE ISCAD'04, 2004-05-23: 269-272.

5 ADSP-BF533 Blackfin Processor Hardware Reference(Revision 1.0)[Z]. Analog Devices Inc., 2003-12: 58.

6 Joint Video Team (JVT). ITU-T Rec. H.264[S]. ISO/IEC 14496-10 AVC, Draft Text of Final Draft International Standard for Advanced Video Coding, 2003-03.

（上接第 261 页）

由该方案实现的软件系统表明了该方法基本上达到了远程医疗的要求，具有较好的推广应用价值。

参考文献

1 张旭东，卢国栋，冯　健. 图像编码基础和小波压缩技术——原理、算法和标准[M]. 北京：清华大学出版社, 2004.

2 成礼智，郭汉伟. 小波与离散变换理论及工程实践[M]. 北京：清华大学出版社, 2005.

3 蒋东兴，林鄂华，陈棋德等. Windows Sockets 网络程序设计大全[M]. 北京：清华大学出版社, 1999.