浙江大学

硕士学位论文

H.264/AVC视频编码标准研究及其编码器的优化

姓名：王嵩

申请学位级别：硕士

专业：通信与信息系统

指导教师：刘济林

20040201

Y 581503

摘 要

随着 21 世纪的到来，人类进入了一个全新的多媒体时代。作为多媒体中最重要、最具表现力和最复杂的数字视频处理，也随着时代的发展取得了长足的进步。ITU－T 和 ISO/IEC 制定的 H.26x 和 MPEG 系列视频编码标准极大的改变了人们的生活。在新世纪，ITU－T 和 ISO/IEC 又一次联合制定和发布了新一代的视频编码国际标准：H.264/AVC。本文深入研究了 H.264/AVC 的编码原理，并在此基础上对其参考软件编码器进行了优化。

绪论部分简要介绍了视频压缩编码的基本方法，基于块匹配的混合编码框架的构成以及主要视频编码标准的特点。

第二章介绍了 H.264/AVC 的编码原理，对其中的主要功能模块：帧内预测、多种块模式的帧间预测、1/4 和 1/8 像素的运动估计、多参考帧、整数变换、CABAC 和环路去块效应滤波器及其包含的新技术进行了详细论述。

第三章首先分析了 H.264/AVC 参考软件编码器 JM61e 的性能，仿真实验显示由于采用了一系列新技术，使得 H.264/AVC 的压缩性能明显超过了现有其它标准。在相同的主观视频质量下，码率减少 50%左右。然后分析了基于 Baseline Profile 编码器的计算瓶颈和热点，确定了下一步编码器优化的重点。

最后根据 H.264/AVC 的特点，采用基于边缘方向信息的帧内预测模式选择快速算法、不对称十字和多六角形网格混合运动搜索快速算法、基于全零块检测的整数变换和运动估计快速算法以及帧间模式选择快速算法对 Baseline 的编码器进行了优化。有效的提高了编码器的编码效率。

关键词：视频编码、H.264/AVC、JVT、编码器、优化

未经作者、导师同意
勿全文公布

Abstract

With the coming of the 21st century, man is entering into a brand-new multimedia age. Digital Video Processing, as the most important, expressive and complex one in multimedia, has progressed quite a lot. H.26x and MPEG-series video coding standards made by ITU-T and ISO/IEC have changed the people's lives significantly. In this new age, ITU-T and ISO/IEC join again to establish and release a new generation international video coding standard: H.264/AVC. This dissertation tries to probe the coding principle of H.264/AVC, and optimize the reference codec on its basis.

The Introduction briefly presents the basic methods of video coding, the structure of block-based hybrid coding frame and the chief characteristics of video coding standards.

Chapter II introduces the encoding principle of H.264/AVC and elaborates on its main function modules of H.264/AVC and new technologies being adopted: intra prediction, multi-block inter prediction, 1/4 and 1/8 fractional pixel motion estimation, multiple reference frames, integer transform, CABAC and deblock filter.

Chapter III first analyses the performance of reference software encoder JM61e. Simulation results reveal that the adoption of a series of new technologies makes H.264/AVC achieve much higher coompression efficiency compared to contemporary video coding stanards. Under the same visual quality, bit rate reduces about 50%. Then analyses the bottle neck and hot spot based on Baseline Profile, and determine which emphasis should be placed on next-step optimization of encoder.

Finally, according to the characteristics of H.264/AVC, apply the Baseline encoder to optimization through fast mode decision for intra prediction based on edge direction information, Unsymmetrical-Cross Multi-Hexagon-Grid motion estimation fast algorithm, integer transform and motion estimation fast algorithm based on all zero coefficient detection, inter prediction mode selection fast algorithm, therefore greatly improve the efficiency of enoder.

Key Words: video coding H.264/AVC JVT encoder optimize

第一章 绪 论

本章简要分析了数字视频编码的背景，发展历史和基本原理。讨论了基于块匹配的混合编码框架下视频编码器的构成以及相关的主要算法和主要标准。最后给出本文的结构安排。

1.1 引言

随着上世纪七十年代后大规模和超大规模集成电路技术、数字信号处理技术、计算机技术、通信技术的跨越式飞速发展，人类对信息的处理也迅速从模拟领域进入数字领域，从本地的单机处理进入网络交互式处理，从简单的文本信息处理进入多媒体信息处理。因此 21 世纪被形象的称为信息时代、数字时代、多媒体时代。

多媒体信息主要包括文字、声音、图像、图形和视频等内容。其中视频又是多媒体信息中最重要的组成部分。这是因为：首先，视频信息极易被人类接受。据统计人类接受的信息大约 70%来自视觉。其次，视频信息具有直观、形象、准确、高效和应用广泛等特点。第三，视频的信息容量大，与音频、数据相比，视频具有无与伦比的信息容量。但与文本、数据和语音相比，数字视频巨大的数据量使得未经压缩的数字视频几乎没有实用价值。例如：对于 CCIR-601 格式的视频材料，当帧率为 25fps，每采样点 8 比特量化，色差格式为 4:2:2 时，每秒数据量为 165.9Mbits。如果直接在容量为 4.7GB 的 DVD 格式光盘中保存，则只能保存不到 4 分钟的内容。对于高清晰度数字电视（ITU-R 709）每秒数据量更高达 884.7Mbits，而地面广播系统的传输带宽仅有 6M 到 8M。因此无论是存储还是传输，数字视频都必须经过极大的压缩才能具有实际意义，这就使得视频压缩技术成为多媒体技术的关键所在。

1.2 视频编码原理

视频编码的目的是实现对视频的压缩，其核心思想是去相关。通过减少视频序列间的相关性，降低视频内容中的冗余，用较少的比特数来表示视频内容，从而实现对视频的压缩。视频序列中的冗余主要有以下几个方面[1-3]。

- 空间冗余

空间冗余是指在同一帧画面中，相邻的像素间存在的相关性，特别是当这些

相邻像素位于同一个视频对象中时，相关性极强。例如在图像的背景区域。

➢ 时间冗余

通常对视频序列而言，除非发生场景切换，否则相继帧在时间上都是连续的。在前后两帧中往往包含与当前帧相同的背景和对象。只是由于镜头的转动或对象的移动使得空间位置发生变化。运动越缓慢，位置的变换越小。因此视频序列在时域存在极强的相关性。

➢ 编码冗余

对于编码符号，其平均码长高于所表示信息的信息熵，这个差值就形成了编码冗余。编码冗余、空间冗余和时间冗余都依赖于图像数据的统计特性，可以统称为统计冗余。

➢ 人眼视觉冗余

由于人眼视觉的非均匀性，使得人眼视觉对某些空间频率感觉迟钝。因此视频中不同频率成分的内容对于人眼系统而言其重要性是不同的。也就是说存在频域冗余。例如人眼视觉系统对亮度信号变化的敏感性高于色度信号变化。因此可以对色度分量进行降采样，同时保持主观视觉质量不变。YUV4:2:0 色差格式就是对色度分量在水平和竖直两个方向进行 2:1 的降采样。另一方面对信号频域的各个分量可以采取不同的量化步距，将人眼视觉不敏感的分量去除，而不会引起主观质量的下降。

➢ 结构冗余和知识冗余

图像的某些区域存在非常强的纹理结构，图像像素值有明显的分布模式，形成结构冗余。或者图像中包含的信息与某些先验知识有关，例如人的五官位置对于人脸而言就是一种先验知识，这种冗余构成知识冗余。

信源编码的方法按照压缩数据能否被准确恢复分为两大类：无损编码和有损编码。虽然无损编码可以无失真的恢复原始数据，但其压缩效率十分有限。因此在视频压缩中都是将无损编码和有损编码结合使用。视频编码中主要压缩技术有以下几种。

● 预测编码

预测编码不是对一个像素直接编码，而是用同一帧（帧内预测编码）或相邻帧（帧间预测编码）中的像素值来进行预测，然后对预测残差进行量化和编码。显然预测编码实际是利用了图像数据中的空间和时间冗余。线性预测编码又称为差分脉冲编码调制 DPCM(Differential Pulse Code Modulation)，由于算法简单，易

于硬件实现，已被各种视频编码标准采纳。

帧间预测编码的主要方法有帧重复法、帧内插法和运动补偿法等。其中运动补偿法在视频编码中使用的最为广泛。运动补偿预测通常可以采用单向预测（一个参考帧），双向预测（两个参考帧）和插值预测（取两个参考帧预测值的平均）来实现。由于运动补偿预测可以有效的减少视频序列的时域冗余，因此成为构成当前主要视频编码标准最基本的技术之一。

- 变换编码

变换编码是构成当前主要视频编码标准的另一项最基本 技术，用来消除图像的频域（变换域）冗余[4]。

正交变换编码通常是将空域相关的像素点映射到另一个正交矢量空间，使得变换后的系数之间相关性降低。常见的正交变换有，K-L (Karhunen-Loeve) 变换、离散傅立叶变换 DFT(Discrete Fourier Transform)、离散余弦变换 DCT(Discrete Cosine Transform)、沃尔什哈达玛(Walsh-Hadamard)变换和哈尔(Harr)变换。K-L 变换是均方误差准则下的最优变换，但实现困难。在现行视频编码标准中几乎都采用了性能最接近 K-L 变换的 DCT。实际上当自相关系数为 1 时，K-L 变换就退化为 DCT 变换[5]。DCT 变换是 1974 年 Ahmed[6]提出的，它具有一组固定的基函数(和图象内容无关)，以及很好的能量压缩和去相关特性。DCT 变换和 DFT 变换密切相关，NxN DCT 可以由它的偶对称扩展 2Nx2N DFT 变换表达出来，这样利用 DFT 变换的可分级特性以及若干 FFT 变换算法中的一个，可以用 $O(2N^2\log_2 N)$操作代替 $O(N^4)$计算出 NxN DCT[1]，除此之外，目前已经有了许多更为实用的 DCT 变换快速算法[7-8]。由于 DCT 变换采用实数计算，加上有效的快速算法出现，使得硬件实现成为可能，因此被广泛的采用。

变换编码除了采用正交变换编码外，还有子带编码[9-10]和小波编码[11-12]。由于正交变换编码使得图像的能量集中在低频区域，表示图象中缓慢变化的内容，而图像的边缘、细微的纹理等细节部分集中在变换域的高频区。为了实现压缩，通常采用同一个量化器进行量化，这样就牺牲了图像的细节部分，造成解码图像模糊。在高压缩比时，基于块的正交变换编码还会产生块效应(block effect)和振铃效应(ring effect)，降低图像质量。而子带编码则是将图像分裂成几个不同频段的子带(Sub-band)，对不同的子带设计不同的编码参数，提高图像质量。小波变换编码充分的利用了小波分析在时域和频域同时具有良好的局部化特性，与人眼视觉特性相符的多分辨率能力，分解系数分布平稳，自然分级的金字塔式数据结构等优点，在视频压缩领域引起广泛的关注。它利用与正交分解完全不同的小波分解，以原始图像（不是原始图像中的块）为初值，不断的将上一级图像分解为 4 个子带：上一级图像中的低频信息、垂直方向、水平方向和对角线方向的边缘

信息。从多分辨率分析出发，一般每次只对上一级的低频子图图像进行分解。将整个图像而非其中的块作为整体进行传送，因此不会产生块效应。由于小波变换的金字塔式数据结构的每一层都包含整个图像的信息，只是其中的分辨率不同，因此可以选择传送部分或全部，非常简单，自然的实现可分级视频编码。

- 统计编码

根据香农信息论的观点，信源冗余度来自信源本身的相关性和信源内部事件概率分布的不均匀性。统计编码主要有基于概率分布特性的霍夫曼编码和算术编码，以及基于相关性的游程长度编码三类。

霍夫曼编码(Huffman coding)是一种变长编码 VLC(Variable Length Coding)。霍夫曼编码将信源符号按概率大小重新排序，通过二叉树算法，依次将两个概率最小的节点合并，直至根结点。完成树的构造后，给所有的树枝分配 0 和 1，这样就可以给高概率符号分配短码，而概率小的符号则分配较长的码字，去除符号间的统计冗余。在已知信源符号概率时，可以给出极好的编码性能。但霍夫曼编码严重依赖信源的统计特性，编码前必须有信源概率分布的先验知识。对于复杂的视频来说，只能用对大量数据统计后获得的近似分布来代替，因此实际应用时无法达到最佳性能。另一方面 VLC 提高了编码效率，但不利于硬件实现。

游程长度编码 RLC(Run-Length Coding)是将符号值相同的连续符号串用一个游程长度（符号数）和一个代表值（值）描述。这样可以用更紧密的序列代替原有的相同值符号串。在视频压缩中，量化后的数据常常出现大量的连零系数，利用游程长度编码可以有效的降低表示零码的比特数。

算术编码[11-14](Arithmetic coding)是 20 世纪 80 年代发展起来的，理论上，算术编码和霍夫曼编码都是最佳的，但在信源概率分布未知的情况下，算术编码优于霍夫曼编码。算术编码的基本原理是用[0, 1]之间的一个概率区间来表示数据序列。将信源 X 的一个给定状态 $x=\{x_1, \ldots x_N\}$与[0, 1]间的一个由大概率 P 和小概率 Q 限定的概率子区间相联系，区间的长度等于序列的概率 $p(x)$。编码器从 $N=1$ 开始，逐位的处理输入的符号流。每输入一位，更新当前符号的条件概率，并以此调整 P 和 Q 限定的概率子区间。随着 N 的增加，和输入符号序列相联系的概率子区间就变得越来越小。最后用这个表示概率子区间的小数给符号序列编码。

- 分形编码和模型基编码

本文的关注焦点主要集中基于块匹配的编码框架中，因此分形编码[15-17]和模型基编码[18-19]超出了本文的讨论范围，不再一一详述。其具体的编码原理可以参见相关的参考文献。

1.3 视频编码标准

视频编码技术的标准化给不同的厂商和视频提供者奠定了一个共同工作的基础，也为编码视频的交互和更为广泛的应用创造了必要的条件。开发一种国际标准需要来自不同国家的许多同行的合作，并需要一个能支持标准化过程和实施标准的组织[20]。视频编码国际标准的制定主要由 ITU-T(International Telecommunication Union-Telecommunication Standardization Sector)和 ISO/IEC (International Organization for Standardization/International Electrotechnical Commission)负责。ITU-T 相继发布了 H.26x 系列标准，而 ISO/IEC 则推出了 MPEG 系列标准。而这些标准都是建立在基于块匹配的混合编码框架下的，并且有非常类似的结构。下面对基于块匹配混合编码框架的基本结构，必要算法和相关视频编码标准作一个简要介绍。

1.3.1 混合编码框架下的视频编码器

基于块匹配混合编码框架下的视频编码系统是将编码帧划分为 NxN 的块，每一个块相对独立的进行处理。其核心思想是利用帧间预测编码消除图像序列中的时域冗余，利用变换编码消除频域冗余。其编码器框图如下：

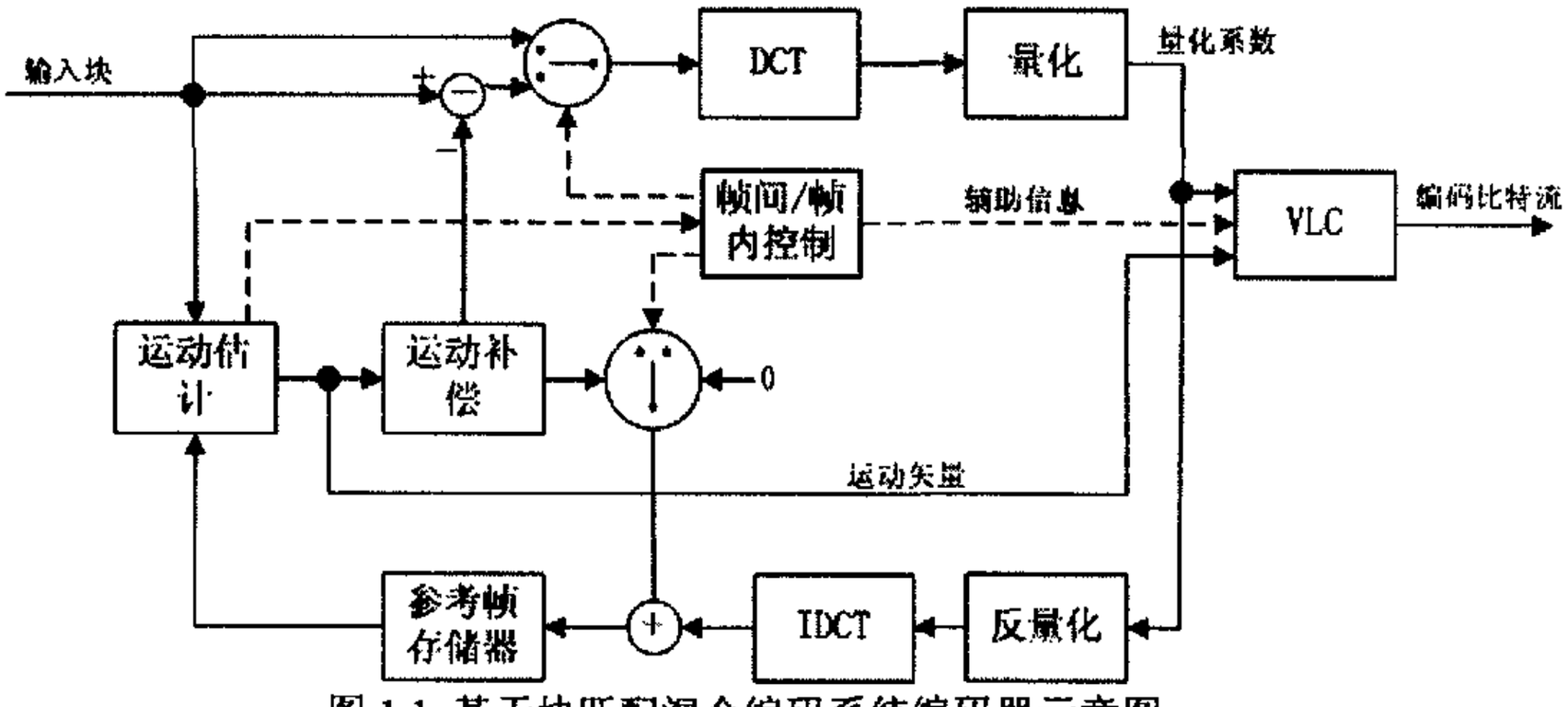

图 1.1 基于块匹配混合编码系统编码器示意图

➢ DCT 变换和量化

在当前的系列标准中采用了 8x8 的 DCT 变换。定义如下：

$$F(u,v)=\frac{1}{4}C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7}f(x,y)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16} \tag{1-1}$$

其中：

$$C(u),C(v)=\begin{cases}\dfrac{1}{\sqrt{2}} & u,v=0\\ 1 & else\end{cases} \tag{1-2}$$

IDCT 定义如下：

$$f(x,u)=\frac{1}{4}\sum_{x=0}^{7}\sum_{y=0}^{7}C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16} \tag{1-3}$$

DCT 变换本身并不能实现数据的压缩，64 个块系数变换后还有 64 个变换系数。但 DCT 变换具有良好的能量聚集能力，可以将能量以较大幅度集中在少数低频系数上。图 1.2 是 DCT 变换能量聚集示意图

183	160	94	153	194	163	132	165
183	153	116	176	187	166	130	169
179	168	171	182	179	170	131	167
177	177	179	177	179	165	131	167
178	178	179	176	182	164	130	171
169	180	180	179	183	169	132	169
179	179	180	182	183	170	129	173
180	179	181	179	181	170	130	169

变换前

313	56	-27	18	78	-60	27	-27
-38	-27	13	44	32	-1	-24	-10
-20	-17	10	33	21	-6	-16	-9
-10	-8	9	17	9	-10	-13	1
-6	1	6	4	-3	-7	-5	5
2	3	0	-3	-7	-4	0	3
4	4	-1	-2	-9	0	2	4
3	1	0	-4	-2	-1	3	1

变换后

图 1.2 DCT 变换的能量聚集示意图

通过随后的量化器量化，高频附近能量幅度较小的系数常常会出现连零串。再经过“之”扫描后，在后续的熵编码阶段实现有效的压缩。

➢ 运动估计

块匹配运动估计 BMME (Block-matching motion estimation) 因其性能良好，易于硬件实现而被系列标准所采纳。图 1.3 为块匹配运动估计示意图[21]。

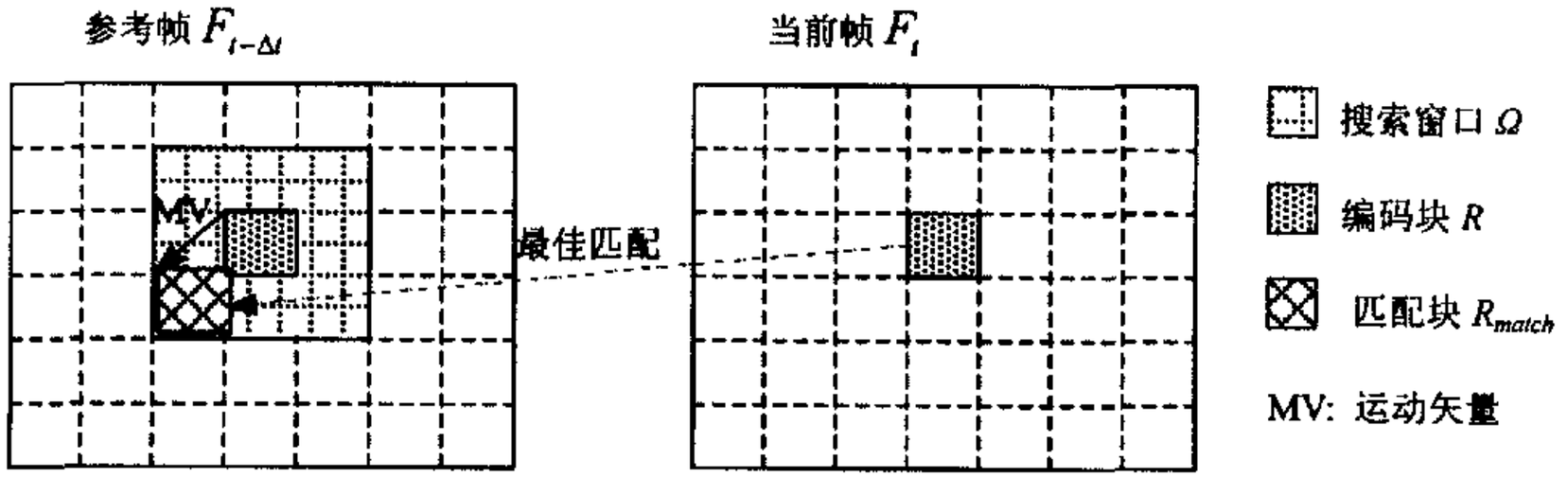

图 1.3 块匹配运动估值示意图

当前编码帧被分割成 MxN 的块，运动估计就是基于一定的匹配代价准则[22]，在参考帧的给定搜索窗口 W 中寻找当前编码块的最佳匹配。常见的匹配代价准则为 *SAD*(Sum of absolute differences)、*SSD*(Sum of squared differences)、*CCF*(Cross-correlation function)和 *PDC*(Pixel Difference Classification)。*SAD* 因其计算简单常被选作运动估计的匹配代价准则。在获得最佳匹配后，就得到了相应的运动矢量 MV。最后预测块和原始块的残差以及运动矢量 MV 被编码传输。

获得最佳匹配的过程叫运动搜索。最基本的算法为全搜索 FS(Full search)，即在搜索窗口中逐点搜寻当前块的最佳匹配。对 CIF 格式的图像，当 M=N=16，搜索半径为 31 时，全搜索每秒钟需完成 1.94×10^{10} 次加法和 9.74×10^{9} 次比较运算[21]。因此运动搜索就成为基于块匹配混合编码系统计算瓶颈所在。目前已有许多性能各不相同的快速算法。通常运动搜索快速算法都是在以下假设条件下建立的：误差曲面存在唯一的全局最小点，误差曲面呈单峰分布，误差曲面不存在零梯度区域。在此基础上形成了三步搜索 TSS (Three-step search)、新三步搜索 NTSS (New three-step search)、二维对数搜索 TDLS(Two-dimensional logarithm search)等算法[22-27]。然而实际的误差曲面不能满足上述假设，对于复杂或高速运动的视频序列更是如此。作为改进，可以利用相邻块（如图 1.4 左、上、右的相邻块）的运动矢量预测初始搜索起点，避免掉入局部最小点。

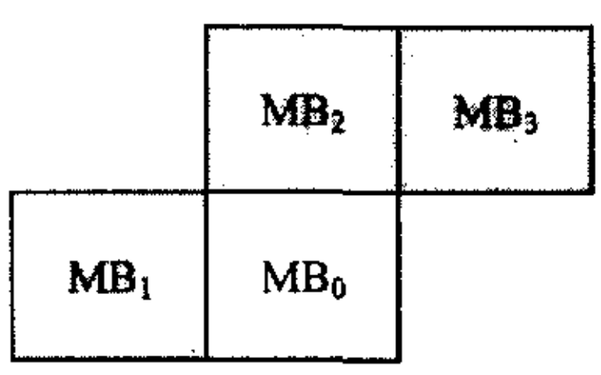

图 1.4 运动搜索起始点预测

在此基础上发展了 MVFAST[28](Motion Vector Field Adaptive Search Technique)、PMVFAST[29](Predictive Motion Vector Field Adaptive Search Technique)和 EPZS[30](Enhanced Predictive Zonal Search)等算法。这些算法利用不同的搜索策略，减少搜索的匹配点数，加速算法收敛。它们可以取得与全搜索相仿的重建图像质量，而运算复杂度有显著的降低。

➢ 熵编码

熵编码的目的是去除编码符号间的冗余。在当前的标准中，主要由游程长度编码和变长编码串接组成。而算术编码在某些标准中也被采纳。算术编码的压缩效率高于变长编码，但增加了计算的复杂度。

1.3.2 主要视频编码标准

➢ H.261[31]

H.261 是第一个获得广泛应用的视频编码标准。它的全称为“Video codec for audiovisual services at p×64kbit/s”。目标是在 ISDN(Integrated Services Digital Network)上以 p×64Kbps，p=1,…,30 的速率开展视频会议和视频电话业务。

H.261 定义了一个完整的视频编码算法，采用了帧内图像编码、帧间误差预测、运动补偿、DCT、变长编码等技术,建立了取得巨大成功的基于块混合编码框架。为后来的 MPEG-1、MPEG-2 等视频压缩标准提供了基础。

➢ MPEG-1[32]

1991 年 11 月活动图像专家组（MPEG）制定的 MPEG-1 标准。MPEG 标准在 H.261 视频编码算法的基础上改进、发展，MPEG-1 改进的主要内容是增加了 B 图像帧（双向预测）和图像组（GOP），这些改进具有更高的压缩比，同时定义了编码算法中各工具层的语法，使视频的可操作性更灵活。

MPEG-1 标准是将数字视频信号和与之相伴的音频信号在一个可以接受的质量下，能被压缩到码率约 1.5Mbit/s 的一个 MPEG 单一流，主要应用于存储应用。MPEG-1 标准只规定了码流语法和解码过程，用户可以很好地利用这个语法的灵活性来设计非常高质量的编码器和非常低成本的解码器。编码器的设计中一些重要参数，如运动估值、自适应量化和码流速率控制等可以由用户自由确定。

速率约为 1.2Mb/s 的用 MPEG-1 算法压缩的视频图像的质量相当于 VHS（家用视频系统）记录质量。空间分辨率限制为每视频帧扫描行 360 个像素，并且在源编码器端的视频信号为 30 帧/秒，非隔行扫描。对大多数原始图像内容，可得到无人工痕迹的图像质量。MPEG-1 标准是 VCD 工业标准的核心，现在已经走入千家万户；利用 MPEG-1 音频的三层的 MP3 音乐格式也倍受青睐。

➢ MPEG-2[33]

ISO/IEC 于 1991 年开始研究新的标准，新标准着力于提高视频质量，提供不亚于 NTSC/PAL 直到 10Mbps 左右的 CCIR601 质量。1994 年公布了 ISO/IEC 13818(MPEG-2)草案，一年后成为国际标准。因此，MPEG-2 标准能广泛应用于卫星广播业务（BBS）、电缆电视（CATV）、数字电视地面广播（DTTB）、点播电视（VOD）、数字声音广播（DAB）、多媒体终端、网络数据库业务、双工通信等众多领域。MPEG-2 是工业标准 DVD 的核心标准。

MPEG-2 是 MPEG-1 的一个超集，它后向兼容 MPEG-1。MPEG－2 又对 MPEG－1 作了重要的改进和扩充。针对隔行扫描的常规电视图像专门设置了“按

帧编码”和“按场编码”两种模式，并对运动补偿作了相应的扩充，使其编码效率显著提高。

档次和等级的划分是MPEG－2为适应不同应用而定义各个子集的结果。“档次”是集成后的完整码流的一个子集，而每个“档次”的“等级”则是对编码参数所作出的进一步的限制。“档次/等级”是通过确定码流中相应的标题信息及附加信息中的有关参数来给定的，这样，为较高“档次”和“等级”码流设计的解码器能够对相同或较低档次的数据解码。

➢ H.263[34]

针对甚低码率（低于64kbps）的视频会议和可视电话的应用，在1995年11月ITU-T推荐的低码率视频编码标准H.263建议草案出台。H.263标准的视频编码算法与H.261相似（运动补偿和DCT算法），但它在性能上有了显著提高。试验表明：在相同的主观质量下，H.263编码码率仅为H.261的一半[35]。与H.261相比，H.263的主要区别如下：

H.263支持更多的图象格式、半像素精度运动估计、宏块（16x16）运动估计和块（8x8）运动估计的自适应变换、3-D(LAST-RUN-LEVEL)而不是2-D(RUN-LEVEL)游程编码、可选的无限制运动矢量、可选的算术编码、可选的重叠运动补偿和四运动矢量/宏块的高级预测模式和可选的双向预测。

在完成H.263标准的制定工作后，为适应在现有的窄带网络环境上传输视频信息，ITU-T在1998年1月通过了H.263标准的第二版H.263+[36]，增加了十二个新的高级模式。2000年11月，又推出了第三版H.263++[37]，新增3个高级模式。新增模式主要包括：参考帧再采样模式、高级帧内编码模式、交替帧间VLC选择模式、分片结构模式、参考帧选择模式、数据分割模式可分级扩展编码等。

➢ MPEG-4[38-39]

1992年，MPEG决定开发新的适用于极低码率的AV编码国际标准。但随着ITU-T的H.263及其第二版，第三版在低码率视频编码领域取了巨大成功，MPEG工作组决定扩大MPEG-4的研究目标。在深入分析了AV领域中电视、计算机、通信以及其交叉融合的发展趋势后，认为MPEG-4应提供用于通信的新方式。其核心是：

基于内容的交互性。支持基于内容的操作和码流编辑，自然与合成数据的混合编码，增强的时间域随机存取。

具有极高的压缩比。在可以比拟的速度下，主观视频质量好于现有的标准，期望在迅速发展的移动通信网中获得应用。具有多个并发流编码能力，实现对景物的多视角编码。

具有通用存储性。提供一种抗误码的鲁棒性，可以实现基于内容的尺度可变性。对重要的对象用较高的时间或空间分辨率表示，具有自适应使用可用资源的能力。作为第一个面向对象的视频编码标准，MPEG-4 的出现具有很强的历史意义。

1.4 全文结构安排

本文以新的视频编码标准 H.264/AVC 为研究对象，在深入研究 H.264/AVC 的文档，性能和计算复杂度的基础上，对其参考软件编码器进行了优化。本文共分五章。

绪论部分简要介绍了视频压缩编码的基本方法，基于块匹配混合编码框架的构成以及主要视频编码标准的特点。

第二章介绍了 H.264/AVC 的编码原理，对其中的主要功能模块：帧内预测、多种块模式的帧间预测、1/4 和 1/8 像素的运动估计、多参考帧、整数变换、CABAC 和环路去块效应滤波器及其包含的新技术进行了详细论述。

第三章首先分析了 H.264/AVC 参考软件编码器 JM61e 的性能，仿真实验显示由于采用了一系列新技术，使得 H.264/AVC 的压缩性能明显超过了现有其它标准。在相同的主观视频质量下，码率减少 50%左右。然后分析了基于 Baseline Profile 编码器的计算瓶颈和热点，确定了下一步编码器优化的重点。

第四章中根据 H.264/AVC 的特点，采用基于边缘方向信息的帧内预测模式选择快速算法、不对称十字和多六角形网格混合运动搜索快速算法、基于全零块检测的整数变换和运动估计快速算法以及帧间模式选择快速算法对 Baseline 的编码器进行了优化。有效的提高了编码器的编码效率。

最后一章为全文总结，并对下一步的工作进行了展望。

第二章 H.264 /AVC 视频编码标准

本章详细分析了 H.264/AVC 视频编码国际标准，并对其中使 H.264/AVC 编码性能得到巨大提升的关键模块进行了详细的分析和研究。

2.1 引言

在1996年初步完成H.263视频编码国际标准的制定工作后，ITU-T确定了近期(Near Term)和长期的(Long Term)两个目标。近期目标是进一步扩展和增加H.263的特色，增强低比特率编码能力，并产生了H.263的增强版，即H.263+、H.263++。长期目标是制定一种新的视频编码标准，以更好的质量、更高的压缩比支持视频会议等低比特率应用，由此产生了H.26L草案[40]。与此同时，ISO/IEC也在继续进行MPEG-4高级视频编码AVC(Advanced Video Coding)的研究。2001年，MPEG对H.26L草案进行了评估并认识到H.26L潜在的优越性，于是由MPEG和VCEG的专家共同组成了联合视频小组JVT(Joint Video Team)，进一步完善H.26L模型，共同发展新的视频编码国际标准。新标准的官方名称分别为：ITU-T Rec. H.264和ISO /IEC MPEG-4 part 10 AVC[41]（或14496-10 AVC）。

新标准的制定工作吸引了学术界和工业界的广泛关注和参与。在JVT制定标准的提案过程中，全球有一百多所大学、公司和科研机构参与了提案，其中不乏HHI(Heinrich Hertz Institute)、微软、英特尔、诺基亚、摩托罗拉、索尼等国际知名的企业和科研机构。而国内包括浙江大学、清华大学、香港科技大学、上广电、华为等高校和企业为避免重蹈DVD的覆辙，也积极参与新标准的发展和制定工作。

新的H.264/MPEG-4 part 10 AVC（以下简称H.264/AVC）视频编码标准在编码质量和压缩比上比原有的视频编码标准都有了明显的提高[42]。在相同的视觉感知质量上，编码效率比H.263、MPEG-2和MPEG-4提高了50%左右[43]，并且有更好的网络友好性[44-45]。虽然ITU-T在发展和制定H.264/AVC的前身H.26L时，主要的目标是为甚低比特率编码提供一种高性能的编码国际标准，但随着MPEG的加入以及更多新编码技术的采纳，H.264/AVC以其卓越的压缩性能在电视、高清晰度电视、卫星电视、存储媒体、无线多媒体应用等方面显示出了巨大的应用潜力。2002年9月，VideoLocus用该公司高度优化的H.264/AVC编解码器在1Mbits/s码率上实现了MPEG-2需5Mbits/s码率的DVD质量视频流端到端的传输[46]。同年10月，UBVideo公司在PIII800MHz的手提电脑上演示了该公司优化的CIF格式实时编解

码器[47]。除此之外，为了加快H.264/AVC的普及和商业化进程，JVT计划将放弃H.264/AVC基本档次(Baseline Profile)的版权[48]，以吸引更多的关注。显然H.264/AVC这个新世纪制定的，面向从高质量到低比特率，从有线到无线各种应用的视频编码国际标准，有望成为新世纪最为成功的国际标准之一。

2.2 H.264/AVC 概述

与早期的视频编码标准(H.261、MPEG-1、MPEG-2、H.263、MPEG-4)类似，H.264/AVC 也是建立在块匹配的混合编码框架上。基本算法依然是通过帧间预测和运动补偿来消除视频序列中的时域冗余，经过变换编码消除频域冗余。因此基本的功能模块：例如预测、变换、量化、熵编码都没有发生根本的变化。图 2.1 为 H.264/AVC 编码器框图[43]。其中深色部分为编码器内嵌的解码器。

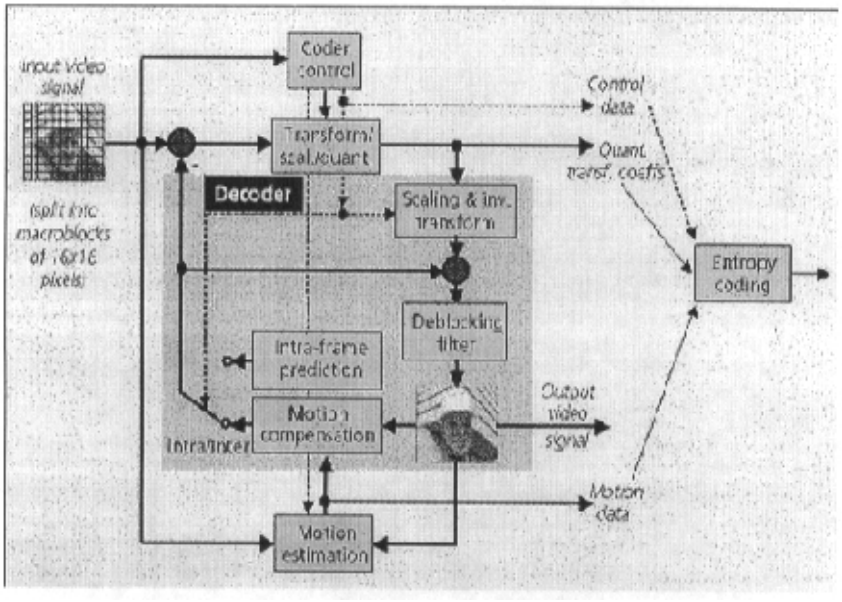

图 2.1 H.264/AVC 编码器框图

基于混合编码框架的视频编码系统在绪论中已做了论述，这里不再详细讨论。H.264/AVC 为了在相同的编码框架下实现更高的视频压缩编码性能和更广泛的适用性，在每一个功能模块中都引入了新的技术，使各功能模块的实现细节都发生了重要的改变。例如 1/4、1/8 像素精度的运动估计、多块模式的帧间预测、多参考帧、帧内预测、环路滤波器、自适应算数编码。下面就对这些变化，也就是 H.264/AVC 的关键技术进行详细的论述。

2.3 H.264/AVC 的关键技术

2.3.1 具有网络友好性的分层结构

随着通信技术和计算机技术的飞速发展，特别是互联网和移动通信网在世界各地的兴起和普及，编码视频的传输网络也将变得越来越复杂。如何增强编码视频的网络适应性，扩展视频编码标准的应用范围日益成为人们关注的焦点。为此，H.264/AVC 在设计上将整个编码系统分成视频编码层 VCL(Video Coding Layer)和网络提取层 NAL(Network Abstraction Layer)两个具有不同概念的层次。图 2.2 为相应的方框图。

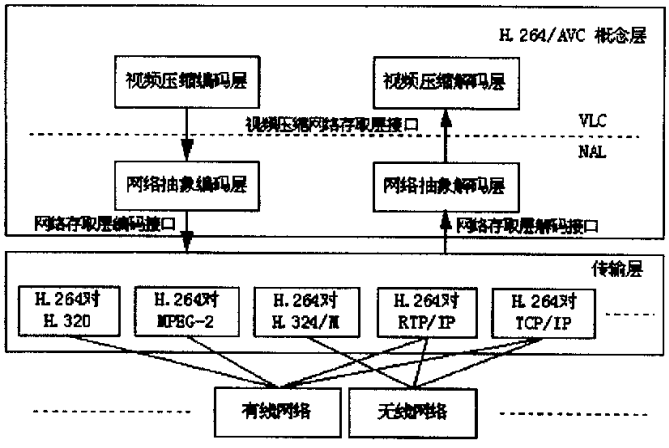

图 2.2 H.264/AVC 标准的分层体系结构

视频编码层 VCL 主要负责对数字视频进行高效编解码，提供具有高质量、高压缩比、健壮性、可分级等特性的视频编码码流。这一部分也是整个 H.264/AVC 视频编码标准的核心部分，同时也是本文研究的重点。但如同前面提到的，编码视频比特流对于不同的传输网络和传输协议并不具有普遍的适应性。为此 H.264/AVC 在视频编码层的外部定义了网络提取层 NAL。

网络提取层 NAL 主要负责将视频编码层 VCL 产生的视频编码数据正确的、恰当的映射到不同的传输网络中去。当 VCL 产生的编码视频比特流将在某种特定网络中传输时，NAL 针对这种网络及其传输协议的特性，对 VCL 的编码码流进行适合该网络及其传输协议的封装。这样 H.264/AVC 就可以在面向不同的传输网络时，灵活的提供不同的封装方式，增强了网络的适应性。NAL 的产生不但使 H.264/AVC 对目前现存的各种不同网络有很强的网络友好性，而且使它对

未来的网络同样具有很强的适应性[49-51]。

2.3.2 帧内预测

视频序列中除了存在已被充分利用的时域和频域冗余外，通常还存在大量的空域冗余，特别是在变化平缓的背景区域。比如图 2.3 为通用视频测试序列 Mather&Daughter 中的第一帧。

图 2.3 Mather&Daughter 第 1 帧

显然，该帧中背景宏块间，特别是墙面部分的变化非常小，存在很强的空间相关性。在 H.264/AVC 以前的视频编码标准中（H.263 系列中，帧内预测只是可选的编码工具之一），对这一类存在强空间相关性的宏块做帧内编码时，只是直接进行变换、量化和熵编码，因此需要用较多的比特编码该宏块。为降低帧内编码的比特数，提高压缩比，H.264/AVC 将帧内预测作为必须得编码工具固定下来。

帧内预测是为了消除视频序列的空间冗余，利用邻近块已解码重构的像素做外推来实现对当前块的预测，预测块和实际块的残差被编码。特别是在变化平坦的背景区域，由于存在大量的空间冗余，利用帧内预测可以取得很好的效果,大大提高编码比特的使用效率，减少帧内编码的比特使用[52]。

但很显然，视频序列的空间相关性远小于其时间相关性[53]。还是以图 2.3 为例，图像中存在活动的主体，静止的背景。即使背景中也存在镜框、椅子背等不同的对象。为了能产生高质量的帧内预测，H.264/AVC 提供了三种帧内预测方式：4x4 亮度块帧内预测、16x16 亮度宏块帧内预测和 8x8 色度宏块帧内预测，并且为每一种预测方式提供多种预测模式。对于相对变化较大、包含多个不同对象的区域，显然需要更小的块分割和更多可选的预测模式，以提供足够的预测精度。

因此 4x4 亮度块帧内预测共有 9 种预测模式。对于 16x16 亮度宏块帧内预测，更适合用在变化很小而面积较大的区域。同时人眼视觉系统对色度变化的敏感性低于亮度变化。因此对 16x16 亮度宏块帧内预测和 8x8 色度宏块帧内预测，所需预测模式应少于 4x4 亮度块帧内预测，共有 4 种预测模式。各预测模式定义如下：

表 2.1 4x4 亮度块帧内预测模式

4x4预测模式编号	4x4预测模式名称
0	Intra_4x4_Vertical (垂直预测)
1	Intra_4x4_Horizontal (水平预测)
2	Intra_4x4_DC (DC预测)
3	Intra_4x4_Diagonal_Down_Left (下-左对角线预测)
4	Intra_4x4_Diagonal_Down_Right (下-右对角线预测)
5	Intra_4x4_Vertical_Right (垂直-右斜线预测)
6	Intra_4x4_Horizontal_Down (水平-下斜线预测)
7	Intra_4x4_Vertical_Left (垂直-左斜线预测)
8	Intra_4x4_Horizontal_Up (水平-上斜线预测)

表 2.2 16x16 亮度宏块帧内预测模式

16x16预测模式编号	16x16预测模式名称
0	Intra_16x16_Vertical(垂直预测)
1	Intra_16x16_Horizontal(水平预测)
2	Intra_16x16_DC (DC预测)
3	Intra_16x16_Plane(平面预测)

表 2.3 8x8 色度宏块帧内预测模式

色度预测模式编号	色度预测模式名称
0	Intra_Chroma_DC(DC预测)
1	Intra_Chroma_Horizontal(水平预测)
2	Intra_Chroma_Vertical(垂直预测)
3	Intra_Chroma_Plane(平面预测)

下面对不同的预测方式以及相应的模式进行详细讨论。

2.3.2.1 4x4 亮度块帧内预测：

图 2.4 为 4x4 亮度块的预测像素及预测方向图。左图中的大写字母 A-Q 表示来自邻近块，已解码重构但尚未进行解块滤波的像素（当这些像素在图象外部或编码次序上滞后于被预测像素时称为不可得），小写字母 a-p 表示将要被预测的 16 个 4x4 亮度块像素。右图为 4x4 块 9 种预测模式的预测方向图。其中模式 2 为 DC 预测，不包括在预测方向图中。下面给出 4x4 亮度块帧内预测各预测模式的具体计算方法。在这里以像素 a 的位置作为坐标原点，用 $pred4x4_L[x, y]$，x, y = 0..3 表示被预测的像素 a-p。用 p[x, -1]，x= 0..7 表示 A-H，用 p[-1, y]，y= 0..7 表示 I-P，用 p[-1, -1]表示 Q。

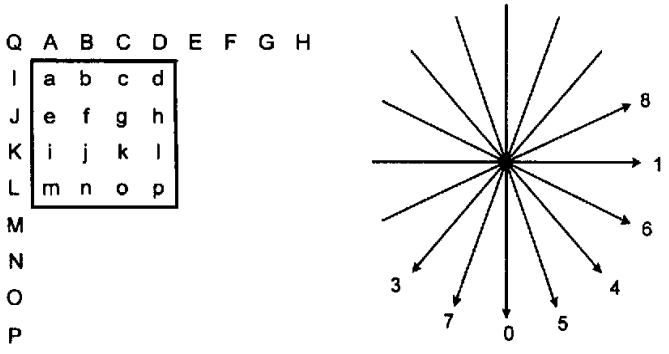

图 2.4 4x4 亮度块的预测及预测方向图

- **模式 0：垂直预测。**

  当像素 A、B、C、D 可得时可以使用。

$$pred4x4_L[x, y] = p[x, -1] \tag{2-1}$$

- **模式 1：水平预测。**

  当像素 I、 J、 K、L 可得时可以使用。

$$pred4x4L[x, y] = p[-1, y] \tag{2-2}$$

- **模式 2：DC 预测：**

  当 A、B、C、D、I、J、K、L 都是可得：

$$pred4x4L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + p[-1, 0] + p[-1, 1] + p[-1, 2] + p[-1, 3] + 4) >> 3 \tag{2-3}$$

  当仅 A、B、C、D 可得：

$$pred4x4L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + 2) >> 2 \tag{2-4}$$

  当仅 I、J、K、L 可得：

$$pred4x4L[x,y] = (p[-1,0] + p[-1,1] + p[-1,2] + p[-1,3] + 2) >> 2 \quad (2\text{-}5)$$

当都不可得：

$$pred4x4L[x,y] = 128 \quad (2\text{-}6)$$

- 模式 **3**：下-左对角线预测，与模式 **0** 成 **45°** 角。

当 A、B、C、D、E、F、G、H 都可得时可以使用。

当 x 等于 3 且 y 也等于 3 时：

$$pred4x4L[x,y] = (p[6,-1] + 3 * p[7,-1] + 2) >> 2 \quad (2\text{-}7)$$

否则：

$$pred4x4L[x,y] = (p[x+y,-1] + 2*p[x+y+1,-1] + p[x+y+2,-1] + 2) >> 2 \quad (2\text{-}8)$$

- 模式 **4**：下-右对角线预测，与模式 **1** 成 **45°** 角。

只有 A、B、C、D、I、J、K、L、Q 都可得时可以使用。

当 x 小于 y 时：

$$pred4x4L[x,y] = (p[-1,y-x-2] + 2*p[-1,y-x-1] + p[-1,y-x] + 2) >> 2 \quad (2\text{-}9)$$

当 x 大于 y 时：

$$pred4x4L[x,y] = (p[x-y-2,-1] + 2*p[x-y-1,-1] + p[x-y,-1] + 2) >> 2 \quad (2\text{-}10)$$

否则：

$$pred4x4L[x,y] = (p[0,-1] + 2 * p[-1,-1] + p[-1,0] + 2) >> 2 \quad (2\text{-}11)$$

- 模式 **5**：垂直-右斜线预测，与模式 **0** 成 **22.6°** 角。

只有 A、B、C、D、I、J、K、L、Q 都可得时可以使用。

令变量 zVR＝2 * x－y。

当 zVR 等于 0，2，4，6 时：

$$pred4x4L[x,y] = (p[x-(y>>1)-1,-1] + p[x-(y>>1),-1] + 1) >> 1 \quad (2\text{-}12)$$

当 zVR 等于 1，3，5 时：

$$pred4x4L[x,y] = (p[x-(y>>1)-2,-1] + 2 * p[x-(y>>1)-1,-1] + p[x-(y>>1),-1] + 2) >> 2 \quad (2\text{-}13)$$

当 zVR 等于-1 时：

$$pred4x4L[x,y] = (p[-1,0] + 2 * p[-1,-1] + p[0,-1] + 2) >> 2 \quad (2\text{-}14)$$

否则（zVR 等于-2，-3）：

$$pred4x4L[x,y] = (p[-1,y-1] + 2 * p[-1,y-2] + p[-1,y-3] + 2) >> 2 \quad (2\text{-}15)$$

- 模式 **6**：水平-下斜线预测，与模式 **1** 成 **22.6°** 角。

只有 A、B、C、D、I、J、K、L、Q 都可得时可以使用。

令变量 zVR＝2 * y– x。

当 zVR 等于 0，2，4，6 时：

$$pred4x4L[x,y]=(p[-1,y-(x>>1)-1]+p[-1,y-(x>>1)]+1)>>1 \quad (2\text{-}16)$$

当 zVR 等于 1，3，5 时：

$$pred4x4L[x,y]=(p[-1,y-(x>>1)-2]+2*p[-1,y-(x>>1)-1]+p[-1,y-(x>>1)]+2)>>2 \quad (2\text{-}17)$$

当 zVR 等于-1 时：

$$pred4x4L[x,y]=(p[-1,0]+2*p[-1,-1]+p[0,-1]+2)>>2 \quad (2\text{-}18)$$

否则（zVR 等于-2，-3）：

$$pred4x4L[x,y]=(p[x-1,-1]+2*p[x-2,-1]+p[x-3,-1]+2)>>2 \quad (2\text{-}19)$$

- **模式 7：垂直-左斜线预测，与模式 0 成 22.6° 角。**

当 A、B、C、D、E、F、G、H 都可得时可以使用。

当 y 等于 0，2 时：

$$pred4x4L[x,y]=(p[x+(y>>1),-1]+p[x+(y>>1)+1,-1]+1)>>1 \quad (2\text{-}20)$$

否则：

$$pred4x4L[x,y]=(p[x+(y>>1),-1]+2*p[x+(y>>1)+1,-1]+p[x+(y>>1)+2,-1]+2)>>2 \quad (2\text{-}21)$$

- **模式 8：水平-上斜线预测，与模式 1 成 22.6° 角。**

当像素 I、 J、 K、L 可得时可以使用。

令变量 zHU＝x + 2 * y。

当 zHU 等于 0，2，4 时：

$$pred4x4L[x,y]=(p[-1,y+(x>>1)]+p[-1,y+(x>>1)+1]+1)>>1 \quad (2\text{-}22)$$

当 zHU 等于 1，3 时：

$$pred4x4L[x,y]=(p[-1,y+(x>>1)]+2*p[-1,y+(x>>1)+1]+p[-1,y+(x>>1)+2]+2)>>2 \quad (2\text{-}23)$$

当 zHU 等于 5 时：

$$pred4x4L[x,y]=(p[-1,2]+3*p[-1,3]+2)>>2 \quad (2\text{-}24)$$

否则：

$$pred4x4L[x,y]=p[-1,3] \quad (2\text{-}25)$$

2.3.2.2 16x16 亮度宏块帧内预测：

对16x16亮度宏块帧内预测，共有256个被预测像素，用Pred(x, y) x, y = 0…15表示。邻近块已解码重构的像素用 P(x, -1) x= -1….15 和 P(-1,y) y= 0….15 表示。

- 模式 **0**：垂直预测。

  当所有邻近的像素 P(x, -1) ， x= 0….15 可得时可以使用。

  $$\text{Pred}(x, y) = P(x, -1), \quad x, y = 0....15。 \tag{2-26}$$

- 模式 **1**：水平预测。

  当所有邻近的像素 P(-1,y) ， y= 0….15 可得时可以使用。

  $$\text{Pred}(x, y) = P(-1, y), \quad x, y = 0....15。 \tag{2-27}$$

- 模式 **2**：**DC** 预测。

  当所有邻近的像素 P(x, -1)， P(-1,y) ， x, y = 0….15 可得时：

  $$\text{Pred}(x, y) = [\sum_{x'=0}^{15} P(x', -1) + \sum_{y'=0}^{15} P(-1, y') + 16] >> 5 \tag{2-28}$$

  当仅有 P(-1,y)可得时：

  $$\text{Pred}(x, y) = [\sum_{y'=0}^{15} P(-1, y') + 8] >> 4 \quad x, y = 0..15, \tag{2-29}$$

  当仅有 P(x,-1)可得时：

  $$\text{Pred}(x, y) = [\sum_{x'=0}^{15} P(x', -1) + 8] >> 4 \quad x, y = 0..15, \tag{2-30}$$

  否则：

  $$\text{Pred}(x, y) = 128 \quad x, y = 0..15, \tag{2-31}$$

- 模式 **3**：平面预测：

  当所有邻近的像素 P(x, -1)、P(-1,y)　x,y= -1….15 可得可以使用。

  $$\text{Pred}(x,y) = \text{Clip1}((a + b\cdot(x-7) + c\cdot(y-7) + 16) >> 5 \tag{2-32}$$

  $$a = 16*(P(-1,15) + P(15,-1)) \text{ 、} b = (5*H+32)>>6 \text{ 、} c = (5*V+32)>>6 \tag{2-33}$$

  $$\text{Clip1}(x) = \begin{cases} 0 & ; x < 0 \\ 255 & ; x > 255 \\ x & ; \text{otherwise} \end{cases} \tag{2-34}$$

$$H = \sum_{x'=0}^{7}(x'+1)*(p[8+x', -1]-p[6-x', -1])$$
$$V = \sum_{y'=0}^{7}(y'+1)*(p[-1, 8+y']-p[-1, 6-y']) \quad (2\text{-}35)$$

2.3.2.3 8x8 色度宏块帧内预测：

色度宏块包含 U、V 两个色度分量宏块。在进行 8x8 色度宏块帧内预测时，对两个宏块使用相同的预测模式。与 16x16 亮度宏块帧内预测的各个预测模式相比，除 DC 预测外，其余预测模式十分类似。对于一个色度宏块，共有 64 个被预测像素，用 $Pred_C(x, y)$ x, y = 0…7 表示。邻近块已解码重构的像素用 $P_C(x, -1)$ x= -1….7 和 $P_C(-1,y)$ y= 0….7 表示

- 模式 **0**：**DC** 预测。

对于 $Pred_C(x, y)$ x, y = 0…3，按如下方式计算。

当所有邻近的像素 $P_C(x, -1)$ 、$P_C(-1,y)$，x,y = 0….3 可得时：

$$Pred_C(x, y) = \left(\sum_{x'=0}^{3}p[x', -1]+\sum_{y'=0}^{3}p[-1, y']+4\right)>>3，x,y = 0….3 \quad (2\text{-}36)$$

当所有邻近的像素 $P_C(x, -1)$ x = 0….3 可得时：

$$Pred_C(x, y) = \left(\sum_{x'=0}^{3}p[x', -1]+2\right)>>2，x,y = 0….3 \quad (2\text{-}37)$$

当所有邻近的像素 $P_C(-1,y)$， y = 0….3 可得时：

$$Pred_C(x, y) = \left(\sum_{y'=0}^{3}p[-1, y']+2\right)>>2，x,y = 0….3 \quad (2\text{-}38)$$

否则：

$$Pred_C(x, y) = 128，x,y = 0….3 \quad (2\text{-}39)$$

对于 $Pred_C(x, y)$ x=4…7，y = 0…3，按如下方式计算。

当所有邻近的像素 $P_C(x, -1)$ x = 4….7 可得时：

$$Pred_C(x, y) = \left(\sum_{x'=4}^{7}p[x', -1]+2\right)>>2，x=4…7，y = 0…3 \quad (2\text{-}40)$$

当所有邻近的像素 $P_C(-1,y)$， y = 0….3 可得时：

$$Pred_C(x, y) = \left(\sum_{y'=0}^{3}p[-1, y']+2\right)>>2，x=4…7，y = 0…3 \quad (2\text{-}41)$$

否则：

$$Pred_C(x, y) = 128，\ x=4...7，\ y=0...3 \tag{2-42}$$

对于 $Pred_C(x, y)$ $x=0...3$，$y=4...7$，按如下方式计算。

当所有邻近的像素 $P_C(-1,y)$， $y=4....7$ 可得时：

$$Pred_C(x, y) = \left(\sum_{y'=4}^{7} p[-1,\ y'] + 2\right) >> 2，\ x=0...3，\ y=4...7 \tag{2-43}$$

当所有邻近的像素 $P_C(x, -1)$ $x=0....3$ 可得时：

$$Pred_C(x, y) = \left(\sum_{x'=0}^{3} p[x',\ -1] + 2\right) >> 2，\ x=0...3，\ y=4...7 \tag{2-44}$$

否则：

$$Pred_C(x, y) = 128，\ x=0...3，\ y=4...7 \tag{2-45}$$

对于 $Pred_C(x, y)$ $x, y=4...7$，按如下方式计算。

当所有邻近的像素 $P_C(x, -1)$ 、$P_C(-1,y)$，$x,y=0....3$ 可得时：

$$Pred_C(x, y) = \left(\sum_{x'=4}^{7} p[x',\ -1] + \sum_{y'=4}^{7} p[-1,\ y'] + 4\right) >> 3，\ x, y=4...7 \tag{2-46}$$

当所有邻近的像素 $P_C(x, -1)$ $x=4....7$ 可得时：

$$Pred_C(x, y) = \left(\sum_{x'=4}^{7} p[x',\ -1] + 2\right) >> 2，\ x, y=4...7 \tag{2-47}$$

当所有邻近的像素 $P_C(-1,y)$， $y=4....7$ 可得时：

$$Pred_C(x, y) = \left(\sum_{y'=4}^{7} p[-1,\ y'] + 2\right) >> 2，\ x, y=4...7 \tag{2-48}$$

否则：

$$Pred_C(x, y) = 128，\ x, y=4...7 \tag{2-49}$$

- **模式 1**：水平预测。

  当所有邻近的像素 $P_C(-1,y)$ ，$y=0....7$ 可得时可以使用。

  $$Pred_C(x, y) = P_C(-1,y),\ \ x, y=0....7。 \tag{2-50}$$

- **模式 2**：垂直预测。

  当所有邻近的像素 $P_C(x, -1)$ ，$x=0....7$ 可得时可以使用。

  $$Pred_C(x, y) = P_C(x, -1),\ \ x, y=0....7。 \tag{2-51}$$

- **模式 3**：平面预测：

  当所有邻近的像素 $P_C(x, -1)$、$P_C(-1,y)$ $x,y=-1....7$ 可得可以使用。

$$Pred_C(x, y) = Clip1((a + b\cdot(x-3) + c\cdot(y-3) +16) >> 5, \quad x, y = 0....7。 \tag{2-52}$$

$$a = 16*(P(-1,7) + P(7,-1)) \ 、\ b = (17*H+16)>>5 \ 、\ c = (17*V+16)>>5 \tag{2-53}$$

$$\begin{aligned} H &= \sum_{x'=0}^{3}(x'+1)*\left(p[4+x',-1]-p[2-x',-1]\right) \\ V &= \sum_{y'=0}^{3}(y'+1)*\left(p[-1,4+y']-p[-1,2-y']\right) \end{aligned} \tag{2-54}$$

2.3.3 多种块模式的帧间预测

与以往的视频编码标准类似，H.264/AVC 也使用块匹配的帧间预测以消除视频序列的时域冗余。在过去的标准中定义了两种块的大小，以像素为单位，分别是 16x16 和 8x8 的正方形块（8x8 的块是在 H.263 和 MPEG-4 中定义的）。但由于视频图像的复杂性，在较大的块中可能包含多个具有不同运动状态和不同形状的对象。特别是在运动剧烈的局部区域中，用 1 个（16x16）或 4 个（8x8）运动矢量并不能准确的描述一个宏块全部的运动细节。

在 H.264/AVC 中，为了更准确的描述宏块的运动细节，定义了 7 种不同尺寸和形状的宏块分割和子宏块分割。

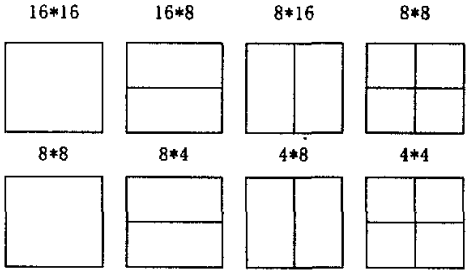

图 2.5 宏块和子宏块分割

在图 2.5 中，上面的是 16x16 的宏块分割，下面的是 8x8 的子宏块分割。这样对于一个宏块，在不同的块模式下，可以用 1 个或最多 16 个运动矢量来描述，并且块的形状可以是正方形或矩形的。这种更小的、更多形状的块模式，可以更好的实现运动隔离。以图 2.6[54]为例。宏块中包含静止的路面和运动的车轮。通过图 2.6 可以形象的看到，由于 H.264/AVC 支持多种块模式，因此比 MPEG-2 和 MPEG-4 更好的实现运动隔离。这样运动估计的残差就会相应减小，提高了帧间预测的效果。

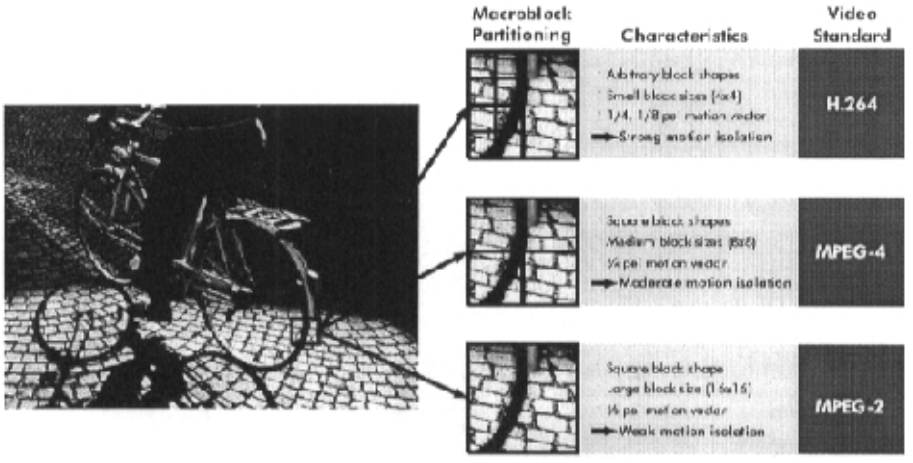

图 2.6 不同块模式中的运动隔离

2.3.4 1/4 和 1/8 像素精度的运动估计

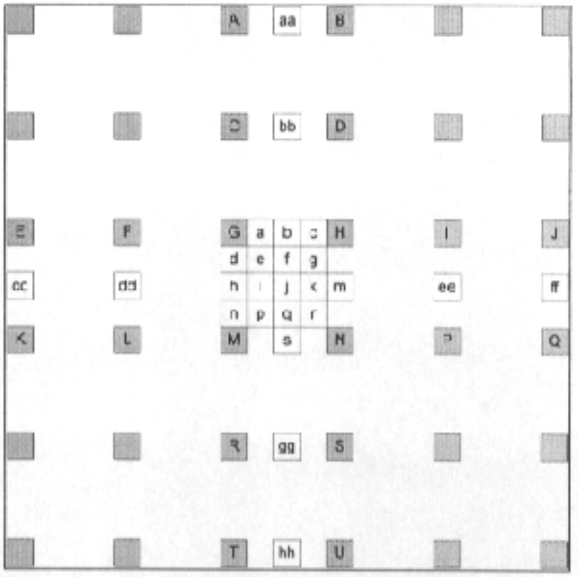

图 2.7 亮度整像素和分数像素位置示意图

运动估计是利用视频图像的时域相关性，产生相应的运动矢量，尽可能准确的描述对象（块或宏块）的时域运动。因此运动矢量的精度越高，运动估计的残差越小，这样在降低编码码率的同时提高重建视频质量。从 H.261 到 MPEG-4，运动矢量的精度也从整像素提高到 1/4 像素。H.264/AVC 支持亮度分量的 1/4 像

素和色度分量的 1/8 像素的运动估计，并详细的定义了相应分数像素的插值实现算法，利用 6 抽头滤波器产生 1/2 分数像素、线性插值产生 1/4 分数像素、4 抽头滤波器产生最高 1/8 分数像素。对亮度分量，以图 2.7 为例。图中，带阴影的大写字母表示整像素位置，不带阴影的小写字母表示分数像素位置。下面给出 G、H、M、N 四个整像素之间的 1/2 和 1/4 像素的插值算法。

- 水平半像素插值。

$$b = Clip1((E - 5*F + 20*G + 20*H - 5*I + J) + 16) >> 5) \quad (2\text{-}55)$$

$$s = Clip1((K - 5*L + 20*M + 20*N - 5*P + Q) + 16) >> 5) \quad (2\text{-}56)$$

- 垂直半像素插值。

$$h = Clip1((A - 5*C + 20*G + 20*M - 5*R + T) + 16) >> 5) \quad (2\text{-}57)$$

$$m = Clip1((B - 5*D + 20*H + 20*N - 5*S + U) + 16) >> 5) \quad (2\text{-}58)$$

- 中心半像素插值。

$$j_1 = cc - 5*dd + 20*h_1 + 20*m_1 - 5*ee + ff \quad (2\text{-}59)$$

或

$$j_1 = aa - 5*bb + 20*b_1 + 20*s_1 - 5*gg + hh \quad (2\text{-}60)$$

其中，aa、bb、cc、等变量为计算相应位置的半像素值时，6抽头滤波器计算的中间结果，以b为例：$bb = (E - 5*F + 20*G + 20*H - 5*I + J)$。其余可以类推。

$$j = Clip1((j_1 + 512) >> 10)$$

- **1/4像素插值**

$$a = (G + b + 1) >> 1、c = (H + b + 1) >> 1、d = (G + h + 1) >> 1 \quad (2\text{-}61)$$

$$n = (M + h + 1) >> 1、f = (b + j + 1) >> 1、i = (h + j + 1) >> 1 \quad (2\text{-}62)$$

$$k = (j + m + 1) >> 1、q = (j + s + 1) >> 1、e = (b + h + 1) >> 1 \quad (2\text{-}63)$$

$$g = (b + m + 1) >> 1、p = (h + s + 1) >> 1、r = (m + s + 1) >> 1 \quad (2\text{-}64)$$

对于色度分量，以图2.8为例。

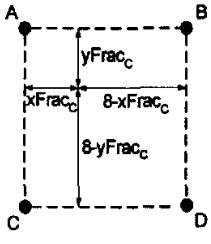

图2.8 色度整像素和分数像素位置示意图

图中，A、B、C、D为整像素位置，$xFrac_C$、$yFrac_C$、$8-xFrac_C$ 、$8-yFrac_C$ 为箭头所指的分数像素位置距整像素边界的距离（以1/8像素为单位）。这样，箭头所指的被预测的分数像素$predPartLX_C[\ x_C, y_C\ ]$计算如下：

$$predPartLX_C[\ x_C, y_C\ ] = ((8-xFrac_C) * (8-yFrac_C) * A + xFrac_C * (8-yFrac_C) * B + (8-xFrac_C) * yFrac_C * C + xFrac_C * yFrac_C * D + 32) >> 6 \quad (2\text{-}65)$$

2.3.5 多参考帧

与过去标准中的单参考帧不同，H.264/AVC 支持多参考帧编码。即通过在多个参考帧中进行运动搜索，寻找出当前编码块或宏块的最佳匹配。在一些特定的情况下，主要是快速的周期运动、快速的场景相互切换、物体存在遮蔽现象等，多参考帧的使用会有非常好的效果。如图 2.9[54]所示

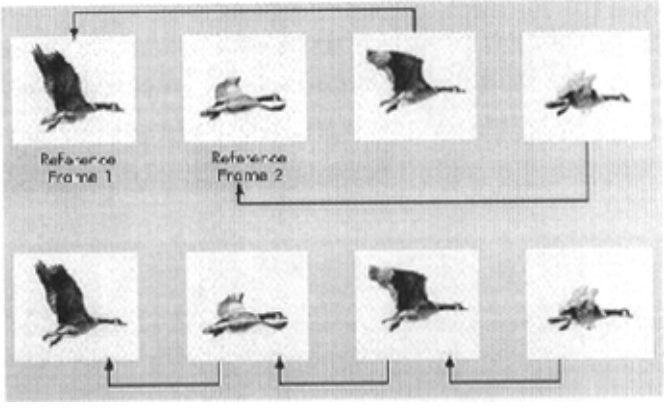

图 2.9 多参考帧示意图

从图中可以明显的发现，如果按下方的单参考帧进行预测，由于连续帧之间鸟的运动状态相差太远，预测效果不佳。如果可以采用多参考帧，由于间隔帧之间的相关性极强，因此预测的效果将大大超过单参考帧。

2.3.6 整数变换及量化

2.3.6.1 整数变换

绪论中已经讨论了变换编码的意义以及在视频压缩编码标准中广泛使用的 8x8DCT 变换。8x8DCT 变换有能量聚集性好，性能接近统计最优的 K-L 变换，计算相对简单，易于软硬件实现等优点。其核心思想是将 N 维矢量 $\boldsymbol{x}$ 通过线性变换 $\boldsymbol{X}=H\boldsymbol{x}$ 映射到新的变换系数矢量 $\boldsymbol{X}$。其中第 k 行 n 列的元素 H_{kn} 定义为：

$$H_{kn}=H(k,n)=c_k\sqrt{\frac{2}{N}}\cos\left[\left(n+\frac{1}{2}\right)\frac{k\pi}{N}\right] \tag{2-66}$$

在 DCT 中变换矩阵是正交的，因此反变换为 $\boldsymbol{x}=H^T\boldsymbol{X}$。由于其采用浮点计算，并且 $H(k,n)$的计算结果是无理数，因此在正变换和反变换之间存在无法避免的舍入误差，即存在变换失配。而在帧间预测时，这种由变换失配引起的舍入误差将被不断的积累、放大。误差积累到一定程度，将引起编码性能的迅速下降。通常可以采用强制帧内刷新来解决这个问题。

在 H.264/AVC 中，不但在帧间编码使用了预测技术，而且帧内编码也使用了预测技术，因此 H.264/AVC 对预测残差变换前后的精度是非常敏感的[55]。例如，在一个 I 帧编码时，每一个 4x4 块都用邻近的已解码重构的像素进行预测，预测块和实际块的残差被变换、量化、编码。而这个 I 帧的重构帧还将成为帧间预测的参考帧。因此如果在变换时存在变换失配，舍入误差在帧内编码和帧间编码阶段都被累积，将迅速被积累和放大。这样将严重破坏 H.264/AVC 的编码性能。此外，对于 8x8 的块变换来说，由于较大的块分割，降低了相继块之间的相关性，在高压缩比时容易出现令人讨厌的块效应。因此 H.264/AVC 中使用了 4x4 的整数变换。这样变换和反变换都在整数上完成，因此变换和反变换将严格匹配，避免了舍入误差的出现。另外，较小的变换块还可以在一定程度上减轻图像的块效应。

由于 DCT 变换在性能上最接近统计最优的 K-L 变换，因此 4x4 整数变换在设计上实质是用整数逼近 DCT 变换。对输入矩阵 X 的 4x4DCT 变换为：

$$Y=AXA^T=\begin{bmatrix}a & a & a & a\\ b & c & -c & -b\\ a & -a & -a & a\\ c & -b & b & -c\end{bmatrix}[x]\begin{bmatrix}a & b & a & c\\ a & c & -a & -b\\ a & -c & -a & b\\ a & -b & a & -c\end{bmatrix} \tag{2-67}$$

其中，$a=1/2$，$b=\sqrt{\frac{1}{2}}\cos(\frac{\pi}{8})$，$c=\sqrt{\frac{1}{2}}\cos(\frac{3\pi}{8})$。逼近的结果是获得如下 4x4 的整数变换：

$$Y=HXH^T=\begin{bmatrix}13 & 13 & 13 & 13\\ 17 & 7 & -7 & -17\\ 13 & -13 & -13 & 13\\ 7 & -17 & 17 & -7\end{bmatrix}[x]\begin{bmatrix}13 & 17 & 13 & 7\\ 13 & 7 & -13 & -17\\ 13 & -7 & -13 & 17\\ 13 & -17 & 13 & -7\end{bmatrix} \tag{2-68}$$

显然，这样的整数变换要求大量的乘法运算，因此 H.264/AVC 中采用的是上面整数变换的一种改进形式，表示如下：

$$Y=HXH^T\otimes E=\begin{bmatrix}1&1&1&1\\2&1&-1&-2\\1&-1&-1&1\\1&-2&2&-1\end{bmatrix}[x]\begin{bmatrix}1&1&1&1\\2&1&-1&-2\\1&-1&-1&1\\1&-2&2&-1\end{bmatrix}\otimes\begin{bmatrix}a^2&ab/2&a^2&ab/2\\ab/2&b^2/4&ab/2&b^2/4\\a^2&ab/2&a^2&ab/2\\ab/2&b^2/4&ab/2&b^2/4\end{bmatrix} \quad (2\text{-}69)$$

其中，$a=1/2$，$b=\sqrt{\frac{5}{2}}$。“$\otimes$”表示 HXH^T 中的每一个元素与 E 中对应位置的元素相乘。这样，在 HXH^T 的变换部分，只有系数 1 和 2，在大多数硬件平台上只有加法操作和平移操作，避免了计算开销大的乘操作。而“$\otimes$E”的部分可以移入量化阶段的常系数矩阵 $QP[qp\_rem][i][j]$，通过查表执行。这样就获得了一种高性能的且计算简单的 4x4 整数变换。

基于 4x4 整数变换，H.264/AVC 中对不同的宏块类型定义了 3 种不同的变换操作类型。

- **4x4 残差数据块变换**

对 4x4 的预测残差信号 **P**，变换系数 **C** 为：

$$\mathbf{C}=\mathbf{HPH}^T$$

其中 **H** 为：$\mathbf{H}=\begin{bmatrix}1&1&1&1\\2&1&-1&-2\\1&-1&-1&1\\1&-2&2&-1\end{bmatrix}$ (2-70)

- **16x16 帧内预测宏块变换**

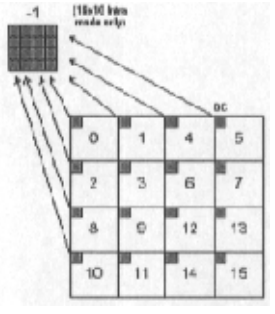

图 2.10 16x16 帧内预测宏块变换示意图

对于 16x16 帧内预测宏块的变换，其变换的次序如图 2.10 中所示编号，从-1 到 15。其中-1 表示 16 个 4x4 块中的 DC 系数所组成的 4x4 变换块，被首先进行变换。其变换矩阵与上述 4x4 残差信号变换略有不同。定义如下：

$$\mathbf{H}=\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\\ 1 & -1 & 1 & -1\end{bmatrix} \tag{2-71}$$

其它 4x4 的残差块变换参照第一种方式进行，只是将 DC 系数设为零。

- 色度宏块变换

对于两个色度宏块，其变换是在完成相应的亮度宏块变换之后进行的。由于在亮度宏块中进行了 0 到 15，共 16 个 4x4 块的变换，因此色度宏块按 16 到 25 的次序完成变换，如图 2.11 所示。其中第 16 和第 17 是两个色度宏块中 DC 系数组成的 2x2 DC 系数块。其变换定义如下：

$$\mathbf{f}=\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}c_{00} & c_{01}\\ c_{10} & c_{11}\end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \tag{2-72}$$

其它 4x4 的残差块变换参照第一种方式进行，只是将 DC 系数设为零。

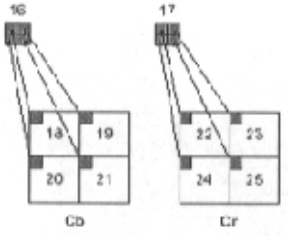

图 2.11 色度宏块变换示意图

2.3.6.2 量化

在 H.264/AVC 的量化部分，没有采用以往标准中的固定量化步距的策略，而是采用了可变的量化步距。量化参数 *QP* 增加 1，量化步距增加 12%，这样量化参数 *QP* 增加 6，量化步距加倍，量化参数 *QP* 共有 52 个可选值。显然，在量化操作上，H.264/AVC 在高量化和低量化上进行了扩展，允许更广泛的量化水平，使得精确的量化成为可能，提高了码率控制能力。令变换系数为 $C(x, y)$，量化后的系数为 $C(x, y)_Q$，则量化计算如下：

- 对 **16x16** 宏块帧内预测的 **4x4DC** 系数：

$$C(x,y)_Q=\left[C(x,y)*QE[qp\_rem][0][0]+2*qp\_const\right]>>(qp\_bits+1) \tag{2-73}$$

- 对色度宏块的 **2x2DC** 系数：

$$C(x,y)_Q=\left[C(x,y)*QE[qp\_rem][0][0]+2*qp\_const\right]>>(qp\_bits+1) \tag{2-74}$$

➢ 对其它 **4x4** 残差块系数：

$$C(x,y)_Q = \left[C(x,y) * QE[qp\_rem][x][y] + qp\_const\right] >> qp\_bits \qquad (2\text{-}75)$$

其中

qp_bits=15+*QP*/6，*qp_rem*=*QP*%6，

帧内预测时：*qp_const*=(1<<*qp_bits*)/3，

帧间预测时：*qp_const*=(1<<*qp_bits*)/6，

常系数矩阵 *QP*[*qp_rem*][*i*][*j*]为：

QP[6][4][4] = {

{{13107, 8066,13107, 8066},{ 8066, 5243, 8066, 5243},{13107, 8066,13107, 8066},{ 8066, 5243, 8066, 5243}},

{{11916, 7490,11916, 7490},{ 7490, 4660, 7490, 4660},{11916, 7490,11916, 7490},{ 7490, 4660, 7490, 4660}},

{{10082, 6554,10082, 6554},{ 6554, 4194, 6554, 4194},{10082, 6554,10082, 6554},{ 6554, 4194, 6554, 4194}},

{{ 9362, 5825, 9362, 5825},{ 5825, 3647, 5825, 3647},{ 9362, 5825, 9362, 5825},{ 5825, 3647, 5825, 3647}},

{{ 8192, 5243, 8192, 5243},{ 5243, 3355, 5243, 3355},{ 8192, 5243, 8192, 5243},{ 5243, 3355, 5243, 3355}},

{{ 7282, 4559, 7282, 4559},{ 4559, 2893, 4559, 2893},{ 7282, 4559, 7282, 4559},{ 4559, 2893, 4559, 2893}}}

2.3.7 CABAC 熵编码

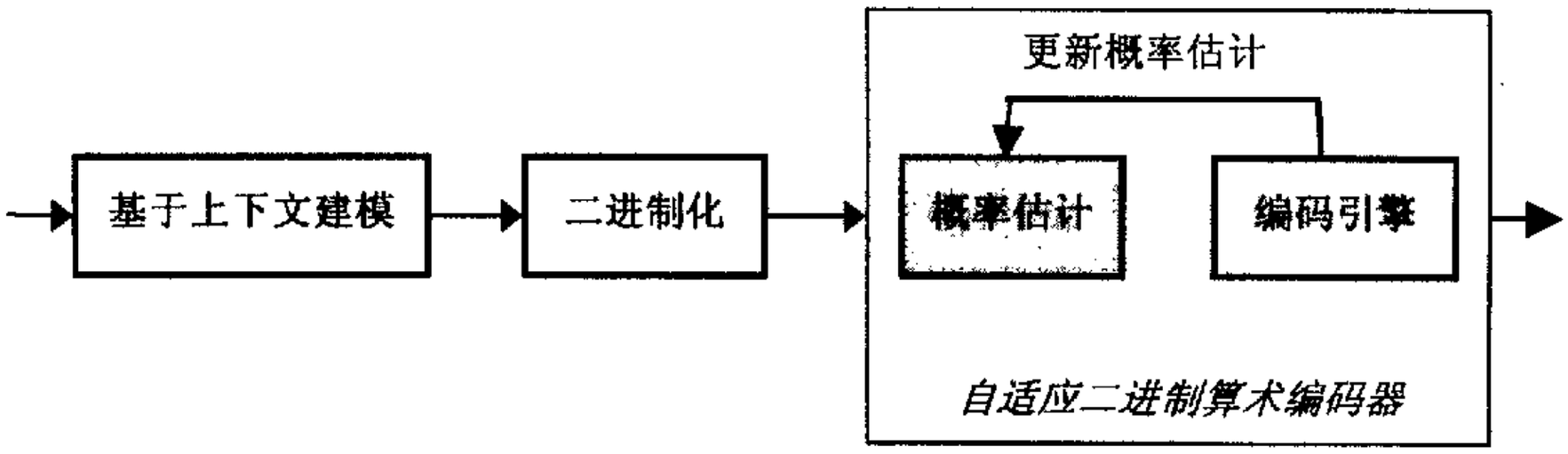

图 2.12 CABAC 编码原理的框图

熵编码是整个视频编码的一个重要环节。它的实质是将最常出现的消息用短码表示，不常出现的消息用长码表示，以使平均码长尽可能的短，达到无损压缩的目的。在绪论中已经提到 UVLC 由于实现相对简单，是最常用的熵编码方法。但算术编码的压缩效率要高于 UVLC。因此 H.264/AVC 中除了支持 UVLC 熵编码外，还支持基于上下文的自适应二进制算术编码 CABAC(Context-based Adaptive Binary Arithmetic Coding)。CABAC 不但充分发挥了算术编码压缩效率高的特点，而且其基于上下文的特点使它可以充分利用不同视频流的统计特性和符号间的相关性，自适应的调整不同符号（消息）出现的概率统计。因此，CABAC

的编码效率十分高，输出码字的信息量可以逼近符号的熵率[56]。图 2.12 是 CABAC 编码原理的方框图。

基于上下文的自适应二进制算术编码 CABAC 的编码过程由以下三步组成：

- 基于上下文建模：

  对于已经过预测、变换和量化，充分利用了其中时域和频域相关性的残差信号以及相关的编码信息来说，符号间冗余的消除是进一步提高压缩效率的关键。而基于上下文建模的实质是为编码符号提供准确的条件概率估计。上下文模型是根据宏块类型，运动矢量，参考帧数，预测模式等不同的编码元素预先定义的概率模型。对于当前待编码符号，首先根据符号的类型（比如是运动矢量还是残差数据）以及左边和上面邻块中相应符号给当前待编码符号选定相应的上下文模型。而这个模型可以为当前待编码符号提供相对更加准确的概率估计（这是由于在第三步，不断根据当前编码情况对上下文模型进行更新）。

- 二进制化：

  由于采用的算术编码器是二进制的，因此所有待编码符号都必须通过这个二进制化过程映射成二进制符号串（Bins）才能被后续的算术编码器编码。对于不同的语法元素可以有不同的二进制映射规则，最常用的是一元二进制转换，见表 2.4

表 2.4 一元二进制转换表

语法元素值	二进制符号串					
0	0					
1	1	0				
2	1	1	0			
3	1	1	1	0		
4	1	1	1	1	0	
5	1	1	1	1	1	0
…						
Bins	0	1	2	3	4	5

- 自适应算术编码

  算术编码的本质就是将编码信息表示为某个概率区间中的一个小数间隔。初始的概率区间是[0，1]，随着编码符号的增加，每个符号出现的条件概率也不断变化，因此概率区间不断缩小，而其自适应性表现在每编码一个符号后，

根据该符号位的信息更新相应的上下文模型(即图 2.12 中的更新概率估计)。这样虽然上下文模型是事先给定的，但在对不同内容编码时，上下文模型是不断动态调整的。

与 UVLC 相比，CABAC 的优点主要有：上下文模型提供编码符号概率分布的估计。利用适当的上下文模型，在编码当前符号时，根据已编码的临近符号的概率统计，在不同的概率模型间转换，充分利用符号间的冗余。算术编码可以给每一个符号的字母分配非整数的比特，因此符号可以接近它的熵率被编码。如果选择了高效的概率模型，符号概率常常大于 0.5，这时分数比特就比 UVLC 的整数（至少 1 比特）比特高效的多。自适应的算术编码可以使编码器自适应的采用动态的符号概率统计。例如，运动矢量的概率统计随空间、时间的不同，或着序列、比特率的不同可以发生巨大的变化。因此自适应模型可以充分利用已编码符号的概率累计，使算术编码更好的适应当前符号的概率，提高了编码效率。

2.3.8 环路去块效应滤波器

基于块的视频编码系统在低码率时无法回避的一个问题就是令人讨厌的块效应。块效应的存在极大的降低了编码视频重建后的主观视觉质量，在 H.263 中第一次出现了去块效应滤波器，有效的降低了重建图像中的块效应，改善了主观视觉质量。

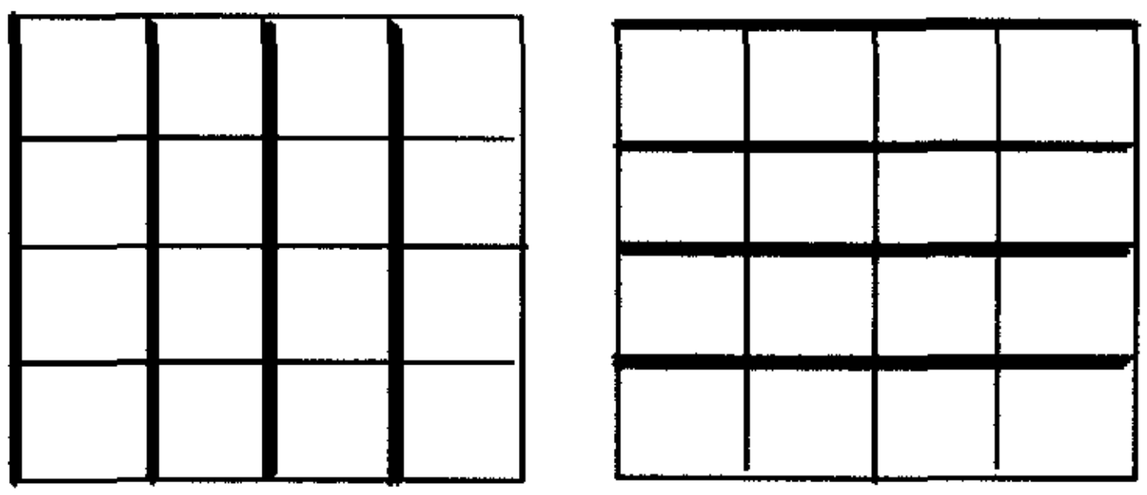

图 2.13 宏块边界示意图

对于 H.264/AVC，首先它的压缩比远高于 H.263，存在更强烈的块效应。其次由于帧内预测时图像的相关性较小，更容易出现块效应。最后在 H.264/AVC 中帧内、帧间编码都使用了预测，块效应产生的失真容易产生累积，导致图像客观质量的下降[57]。因此，在 H.264/AVC 中使用了环路去块效应滤波器。去块效应滤波器应用在反变换后，图像重构前。根据宏块中每一个块的位置和量化参数不同，对每一条块边界设置不同的滤波强度，自适应的调整滤波效果。由于在

H.264/AVC 中，变换块的大小为 4x4，因此在宏块中按下面的顺序以 4×4 块为单位进行水平和竖直边界滤波。如图 2.13。

首先对 16x16 亮度宏块的 4 个垂直边界滤波，其次对亮度宏块的 4 个水平边界滤波，再次对 8x8 色度宏块的 2 个垂直边界滤波，最后对色度宏块的 2 个水平边界滤波。通过这种自适应调整强度的去块效应滤波，有效地改善解码图象的主观视觉质量。并且在编码器中用滤波的宏块做运动补偿时，可以减小预测残差，提高压缩效率。解块滤波器的滤波将影响临近块边界的至多 3 个像素，如图 2.14。

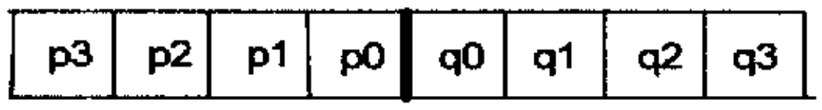

图 2.14 边界像素示意图

图中 p3 到 p0 属于块 P，q3 到 q0 属于块 Q，中间的粗体线表示块边界。环路去块效应滤波由以下三步构成：

- 自适应滤波强度的设定

  对于每一个 4x4 块的边界，都定义一个名为边界强度 BS（Boundary Strength）的变量，此变量将决定不同的滤波计算方法，使得对不同性质的边界滤波强度不同，这样就在平滑边界像素的同时最大程度的保证图像不会因此而变得模糊。BS 的自适应设定如图 2.15 所示。

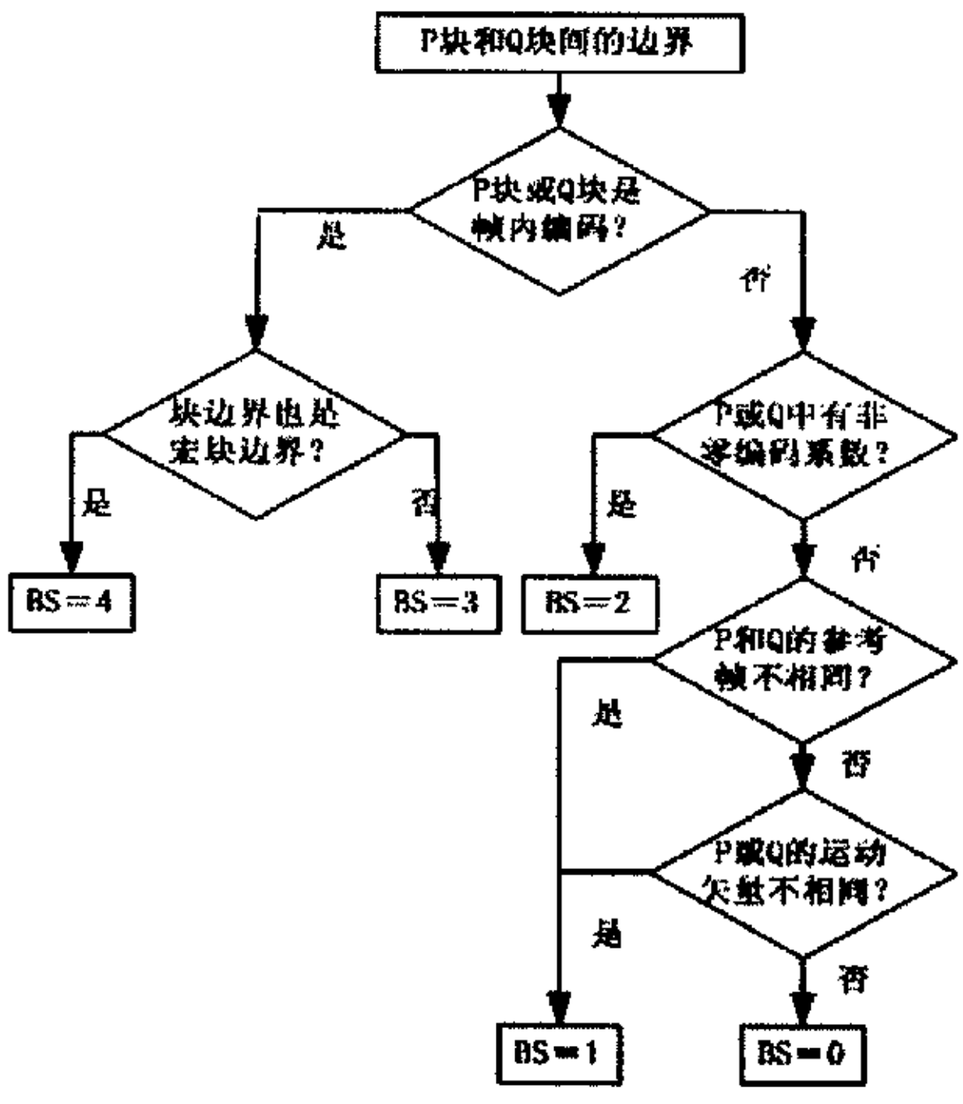

图 2.15 边界强度设定示意图

- 块边界的阈值

  由于图像中还存在着真实的物体边界，并且真实边界间像素变化的梯度通常

都是大于由块效应造成的虚假边界间像素变化的梯度。为准确保留物体的真实边界，在 H.264/AVC 中还设定了两个阈值 α 和 β，只有当下式成立，才进行滤波操作，由此来跳过对真实边界的滤波。（Abs 表示取绝对值）

$$BS != 0 \;\&\&\; Abs(p_0 - q_0) < \alpha \;\&\&\; Abs(p_1 - p_0) < \beta \;\&\&\; Abs(q_1 - q_0) < \beta \tag{2-76}$$

α 和 β 的取值由量化参数 QP 决定，可查表获得。

- 边界滤波计算（以 P 块为例，Q 块计算方法相同）

✧ 当 BS<4 时：

滤波后的样本 p'_0 为：

$$\Delta = Clip3(-t_C, t_C, (((q_0 - p_0) << 2) + (p_1 - q_1) + 4) >> 3)) \tag{2-77}$$

$$Clip3(a, b, c) = \begin{cases} a & ; c < a \\ b & ; c > b \\ c & ; \text{otherwise} \end{cases} \tag{2-78}$$

$$p'_0 = Clip1(p_0 + \Delta) \tag{2-79}$$

如果是色度分量：

$$t_C = t_{C0} + ((a_p < \beta)\ ?\ 1 : 0) + ((a_q < \beta)\ ?\ 1 : 0) \tag{2-80}$$

否则：

$$t_C = t_{C0} + 1 \tag{2-81}$$

其中：

$a_p = Abs(p_2 - p_0)$、$a_q = Abs(q_2 - q_0)$、t_{C0} 由量化参数 QP 决定，可查表获得。

滤波后的样本 p'_1 和 p'_2 为：

如果是亮度边界且 $a_p < \beta$

$$p'_1 = p_1 + Clip3(-t_{C0}, t_{C0}, (p_2 + ((p_0 + q_0 + 1) >> 1) - (p_1 << 1)) >> 1) \tag{2-82}$$

否则

$$p'_1 = p_1 \tag{2-83}$$

$$p'_2 = p_2 \tag{2-84}$$

✧ 当 BS＝4 时：

如果是亮度分量，且下式成立

$$a_p < \beta \quad \&\& \quad Abs(p_0 - q_0) < ((\alpha >> 2) + 2) \tag{2-85}$$

则

$$p'_0 = (p_2 + 2*p_1 + 2*p_0 + 2*q_0 + q_1 + 4) >> 3 \tag{2-86}$$

$$p'_1 = (p_2 + p_1 + p_0 + q_0 + 2) >> 2 \quad (2\text{-}87)$$

$$p'_2 = (2*p_3 + 3*p_2 + p_1 + p_0 + q_0 + 4) >> 3 \quad (2\text{-}88)$$

否则

$$p'0 = (2*p1+p0+q1 + 2) >> 2 \quad (2\text{-}89)$$

$$p'_1 = p_1 \quad (2\text{-}90)$$

$$p'_2 = p_2 \quad (2\text{-}91)$$

2.4 小结

本章分析了 ITU-T 和 ISO/IEC 在新世纪联合制定的视频编码国际标准 H.264/AVC 的编码原理。虽然与早期标准类似，H.264/AVC 也是基于块匹配的混合编码框架，但各个功能模块的具体细节发生了巨大的变化。而这些变化正是使其性能得到大幅度提升的关键技术。因此本章的随后部分对其中的主要功能模块：帧内预测、多种块模式的帧间预测、1/4 和 1/8 像素的运动估计、多参考帧、整数变换、自适应二进制算术编码和环路去块效应滤波器及其包含的新技术进行了详细论述。

第三章 H.264 /AVC 性能及复杂性分析

本章分析和比较了 H.264/AVC 参考编码器的编码性能，仿真结果显示前文所述的关键技术使 H.264/AVC 的压缩性能得到巨大提升．接着对编码码器各模块的复杂性进行了测试，找出编码器运行的瓶颈和热点，并据此确定了下一步编码器优化的重点．

3.1 引言

在第二章对 H.264/AVC 的编码原理和关键技术进行了详细的分析。新技术的使用必将带来编码性能的提升，因此在本章中对 H.264/AVC 中各关键模块的编码性能进行了测试，同时为了对编码器进行有效的优化，在 3.3 节对编码器的复杂性进行了测试，并据此来确定随后编码器优化工作的重点。

3.2 性能分析

本文采用 JVT 的参考软件：联合模型(Joint Model)JM61e[58]对 H.264/AVC 的性能进行了测试。缺省的编码参数为：采用 1 个参考帧，CAVLC 熵编码，支持率失真优化，I 帧和 P 帧的量化参数相同且比 B 帧的量化参数小 2，没有使用数据分割，搜索半径为 32。对帧内预测、1/4 像素的运动估计、多块模式帧间预测、多参考帧、率失真优化、去块效应滤波器、CABAC 熵编码的性能进行了测试，

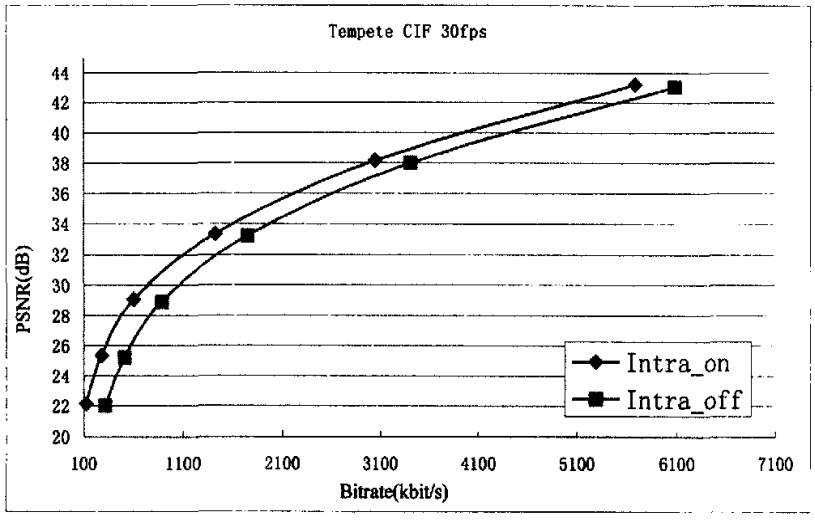

图 3.1 帧内预测 R-D 曲线

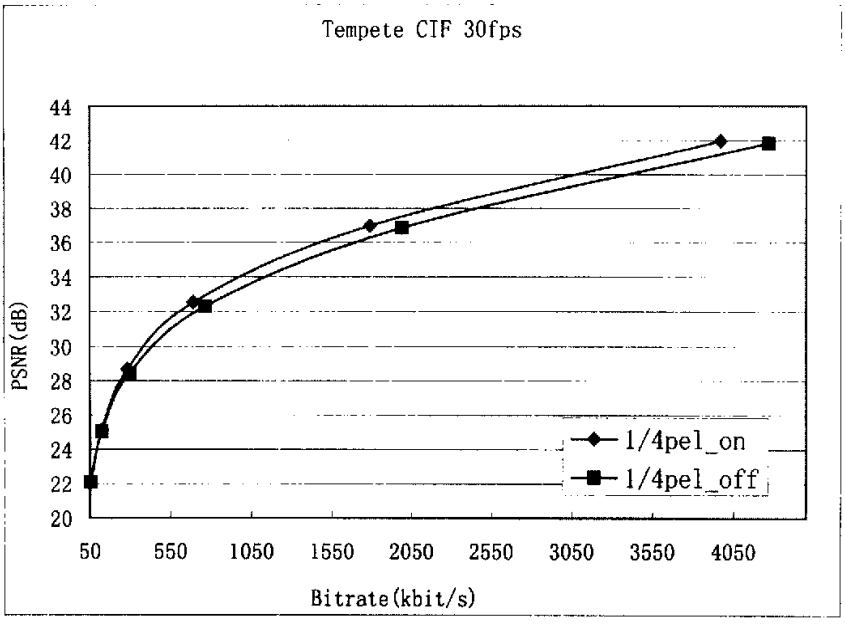

图 3.2　分数像素 R-D 曲线

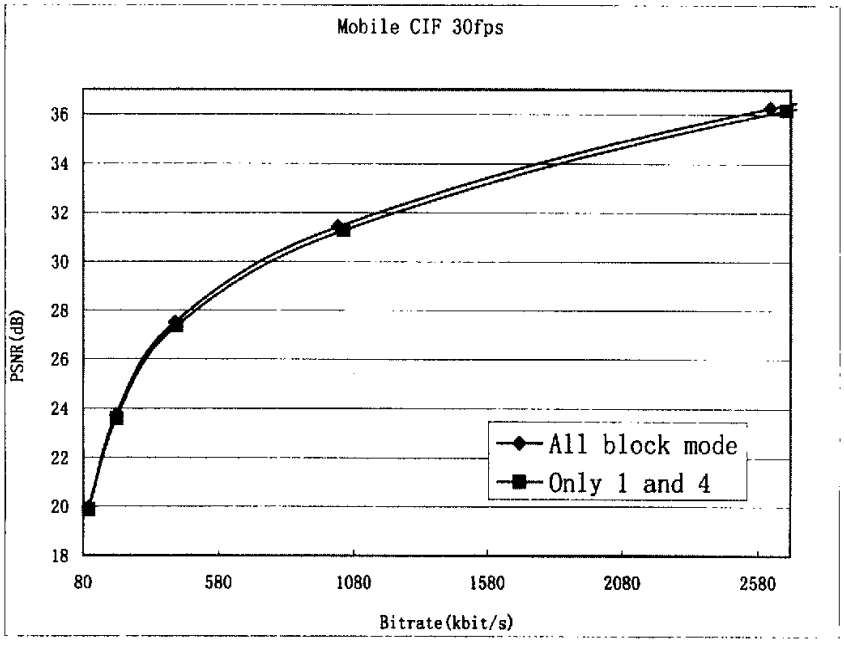

图 3.3　多块模式 R-D 曲线

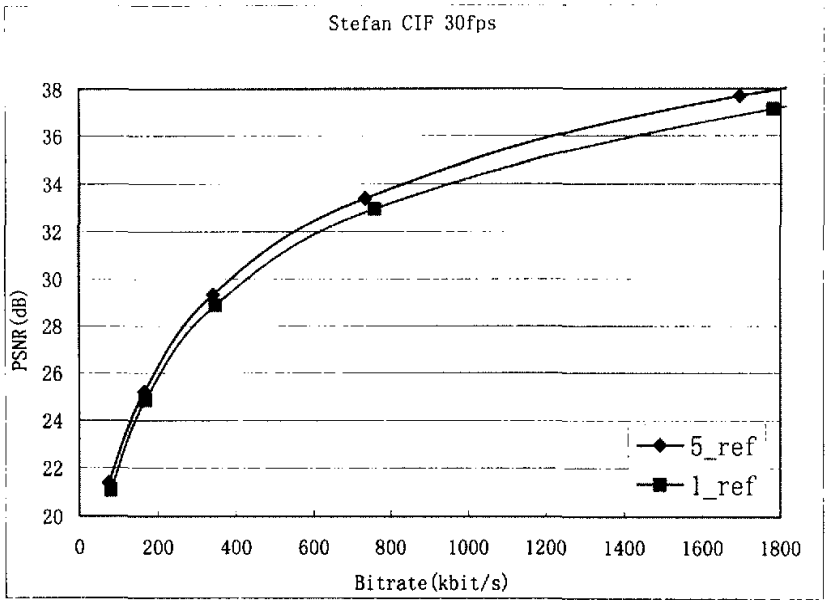

图 3.4 多参考帧 R-D 曲线

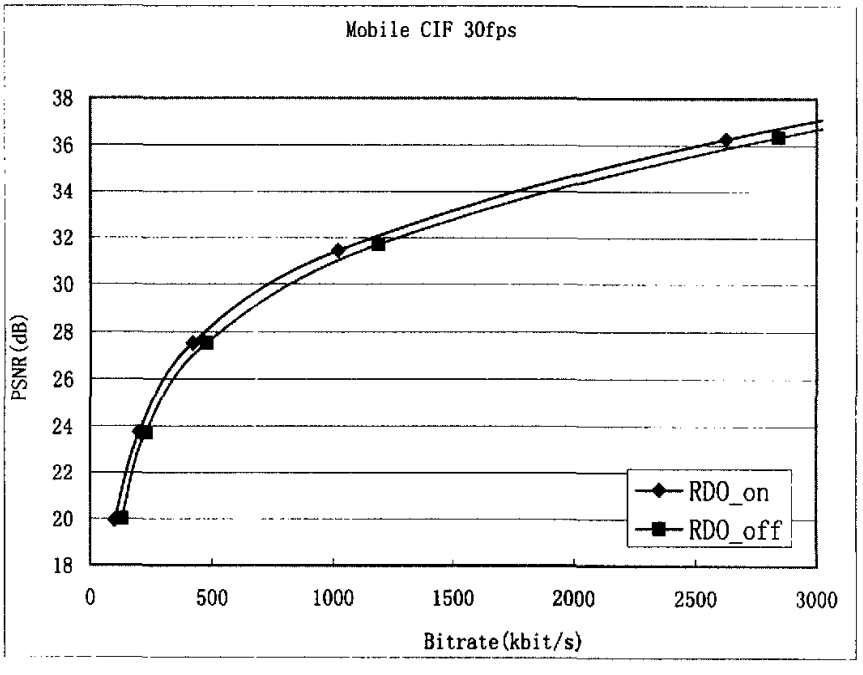

图 3.5 RDO 性能 R-D 曲线

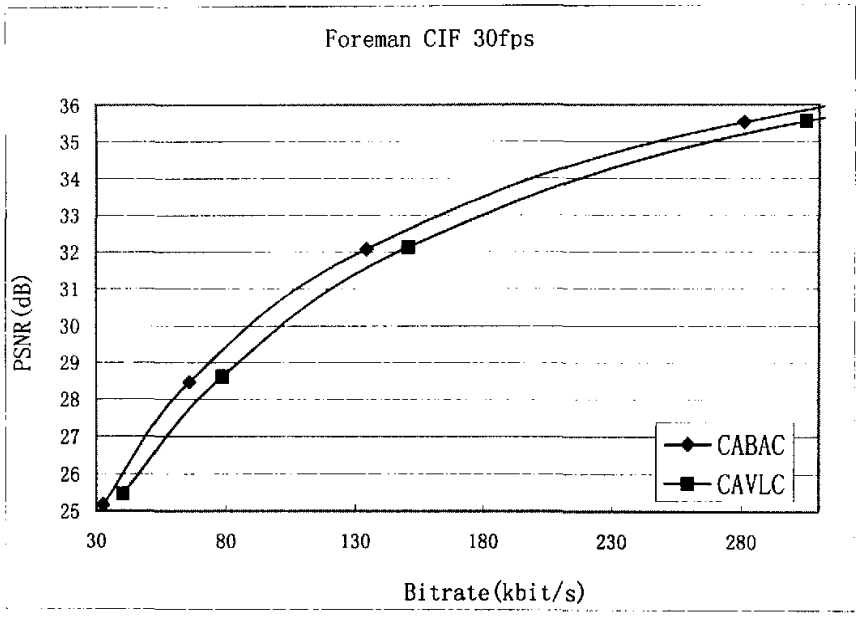

图 3.6 CABAC R-D 曲线

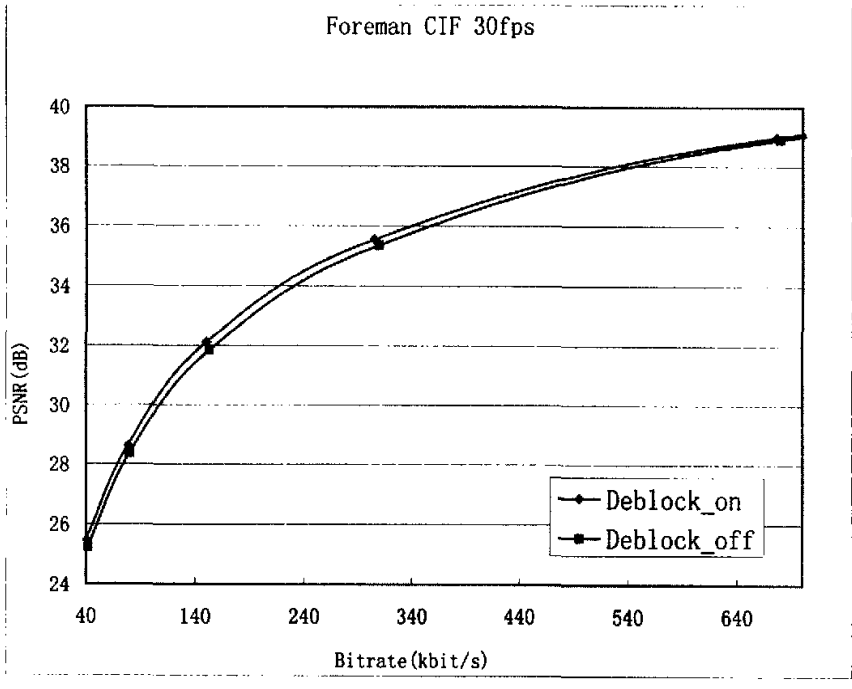

图 3.7 去块效应滤波器 R-D 曲线

图 3.8 解块滤波器主观图像质量比较

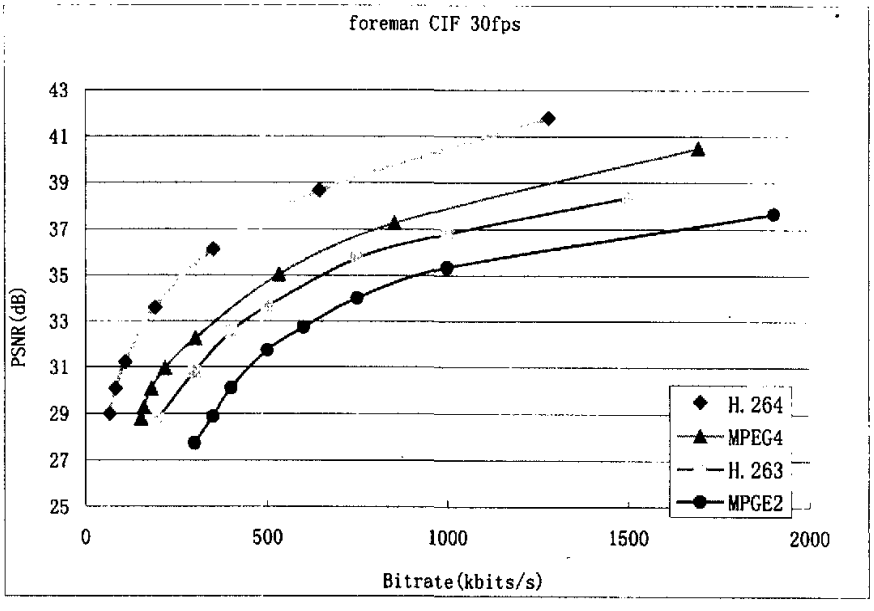

图 3.9 与其它标准的比较

测试结果显示以上每一个功能模块都能使 H.264/AVC 的编码性能得到一定程度的提升，其中帧内预测、分数像素运动估计、RDO 优化技术、CABAC 等模块对性能提升作用突出。多参考帧在高码率时效果较好，但此时编码器的计算和内存开销将成倍增加。去块效应滤波器不但能有效的改善主观视频质量，而且可以在低码率条件下提高图像的信噪比。最后在图 3.9 中比较了 H.264/AVC、MPEG-4、H.263 和 MPEG-2 的编码性能。可以看到 H.264/AVC 的编码性能大幅度的超过了其它的几个标准。

3.3 复杂性分析

计算复杂性分析是有针对性的进行高效编码器优化的一个重要步骤。而H.264/AVC 极高的计算开销也一直受到业界的关注[59-61]。本文对 JM61e 的计算复杂性进行了测试，仿真测试的环境和缺省的编码参数与性能分析中的一致，对于不同格式的序列，搜索半径 W 分别为：$W_{QCIF}=16$，$W_{CIF}=32$，$W_{CCIR601}=64$。为了定量测定编码器各模块的计算复杂性，本文使用英特尔公司的 Vtune 性能测试软件对各模块占用的 CPU 时钟数进行统计，以此作为计算复杂度的客观反映。参考软件中使用全搜索算法 FS 进行运动搜索。统计数据见表 3.1。其中率失真开销指的是基于率失真优化技术(RDO)的编码模式选择的计算开销。

表 3.1 不同模块计算开销(百分比)

序列	格式	*QP*	运动估计	率失真开销	变换和量化	其它
Akiyo	QCIF	28	28.93	30.38	21.86	18.83
Container	QCIF	28	31.23	33.68	22.39	12.70
Foreman	QCIF	28	34.79	34.26	21.35	9.60
Bream	CIF	32	55.86	20.89	16.93	6.32
Mother_Daughter	CIF	32	55.50	18.60	17.55	8.35
Mobile	CIF	32	60.17	21.19	12.12	6.52
Cheer	CCIR 601	36	84.72	7.14	5.25	2.89
Flower_garden	CCIR 601	36	80.17	8.82	6.60	4.41

从表 3.1 可以发现，参考软件编码器各模块的计算复杂度随序列的不同有明显的变化。但总体上运动估计、基于 RDO 的模式选择以及变换和量化模块的计算复杂度最高，并且随图像的尺寸增大，运动估计所占比例急剧增加，最高接近 85%。

因此本文将优化的重点放在运动估计、模式选择和变换量化模块。还要说明的是虽然 RDO 技术要求极高的计算开销，但它可以在明显降低编码比特率的同时，提高编码视频的质量[62-63]，也就是实现基于率失真的最优编码模式选择。在本文优化的编码器中依然支持这项技术。在本文中没有考虑多参考帧的优化。

3.4 小结

本章对 H.264/AVC 的参考软件编码器 JM61e 的性能进行了测试。测试结果验证了新技术带给 H.264/AVC 的令人惊讶的编码性能，但同时也可以注意到伴

随着高性能的是对硬件计算能力、存储能力要求的急剧增加，因此目前的参考软件编码器的编码效率是不能令人满意的。一个成功的视频编码标准不但要有好的编解码性能，而且必须能尽快的实现商业应用，占领主流市场。只有这样才能得到学术界、工业界和消费者的普遍承认，成为真正的国际标准。面对 H.264/AVC 对计算能力和存储能力的极高要求，一方面可以期待硬件技术的飞速发展满足 H.264/AVC 的要求，另一方面必须针对 H.264/AVC 的新特点，寻找更为高效的编解码算法，提高编解码的效率，这也是目前研究的热点，并且已经取得了一些研究成果[64-69]。

作为一种可分级的视频编码系统，H.264/ AVC 中定义了三个不同的层次(Profile)：Baseline Profile、Main Profile 和 Extended Profile。分别包含不同的编码工具，面向不同的应用。Baseline Profile 计划是放弃版权的，因此避免包含涉及专利的编码工具[70]，只支持 I、P 帧的帧编码和 UVLC 等基本的编码工具。Main Profile 将是应用面最宽泛的。它的主要特色是：支持 I、P、B 帧，支持隔行序列编码（可在帧、场、帧/场自适应、宏块帧/场自适应自行选择），支持 CABAC 和 UVLC 等主要编码工具。而 Extended Profile 主要面向专业应用[71]，包含 YUV4:2:2，16 比特样本量化，自适应块变换(ABT)等较为复杂的编码工具。由此可见，对属于不同层次的编解码器其复杂性是不同的，从 Baseline Profile 到 Extended Profile 逐步增加。本文选择了基于 Baseline Profile 的编码器作为研究对象进行了优化。具体的优化过程在第四章给出。

第四章 基于 Baseline 的编码器优化

H.264/AVC 在引入新技术提升编码性能的同时给整个系统，特别是编码器增加了巨大的计算开销，其编码效率令人难以忍受．本文对其中的帧内预测模式选择、运动搜索、整形变换等模块进行了优化，有效的提高了编码效率．

4.1 引言

在前文中对 H.264/AVC 参考软件编码器的编码性能进行了分析。仿真实验结果显示在相同的峰值信噪比 PSNR 下，H.264/ AVC 相对于 MPEG-4，H.263，MPEG-2 码率平均降低分别约为 41%，52%和 67%。显然，H.264/ AVC 的编码性能明显超过 MPEG-2，H.263 和 MPEG-4。但代价也是明显的，新技术极大的增加了编码器的计算复杂性，使得编码效率非常之低，在上一章复杂性测试的仿真实验中，对 CCIR601 格式序列编码为 0.025fps，对 CIF 格式序列为 0.242fps，对 QCIF 格式序列为 1.851fps。因此从实际应用的角度，编码器必须进行极大的优化以提高编码效率。

视频编解码程序的优化通常包括算法优化、多媒体指令优化、代码层次优化和编译器的优化[72-73]。本文因此根据第二章的分析，将优化重点放在运动估计模块和模式选择上。本章组织如下：4.2 节对帧内预测模式选择的算法进行了优化。4.3 节是运动搜索的快速算法。4.4 节是关于帧间预测模式选择。4.5 节为全零块检测及其在运动搜索中的应用。4.6 节为本章小结。

4.2 帧内预测模式选择快速算法

帧内预测可以充分利用视频序列中固有的空间相关性，提高视频在帧内编码时的压缩效率。由于视频序列的空间相关性远小于其时间相关性，因此为确保获得高压缩比以及良好的编码视频质量，H.264/AVC 提供 9 种 4x4 块的帧内预测模式（M4＝9）、4 种 16x16 宏块的帧内预测模式（M16＝4）、4 种色度宏块的帧内预测模式（M8＝4）。这样在编码器端就会面临如何选择最优模式的问题。在 JVT 的 H.264/AVC 参考软件：联合模型 JM61e 中，有两种不同的选择方法。第一种是直接计算各个模式下宏块的绝对差 SAD，选取具有最小 SAD 开销的那一种作为最优模式，并依次对宏块进行编码。另一种则采用了 RDO 技术，即计算如下率失真开销 J，选取具有最小率失真开销的那一种作为最优模式。

$$J = SAD + \lambda * R \tag{4-1}$$

其中 SAD 为宏块（块）的绝对差、λ 是拉各朗日乘子、R 是编码比特率。因此以上模式选择算法的核心思想都是遍历所有的可选模式，选择出最佳编码模式，特别是在使用了 RDO 技术自适应的选择最优模式时，还需考虑不同模式组合时的率失真开销，所以伴随着高性能的是巨大的计算开销。因此如何降低帧内预测模式选择的计算开销就成为本文研究工作的一个重要部分[74]。

4.2.1 帧内预测的模式选择

H.264/AVC 为了实现高质量高压缩比，采用了 RDO 技术。在对 I 帧编码时，需要通过遍历所有可能的帧内编码模式组合，寻找率失真开销最小的那一种作为最优编码模式。而对于 P、B 帧编码，则需遍历所有的帧内和帧间编码模式。以编码 I 帧为例，其一个宏块组合（指一个 16x16 的亮度宏块和相应的两个 8x8 的色度宏块）的 RDO 模式选择过程如下：

1、对于亮度宏块中某一 4x4 块，按某种 4x4 帧内预测模式 $mode_{4x4}$ 建立相应的帧内预测块。

2、计算预测的 4x4 块和原始的 4x4 块之间的绝对差（SAD_{4x4}）。

3、计算该模式的率失真开销 $Rdcost_{4x4}$。

4、在该 4x4 块中重复以上 1 到 3 步，遍历所有的 9 种 4x4 帧内预测模式。

5、选取具有最小率失真开销 $Rdcost_{4x4}$ 的模式作为该预测块的最佳 4x4 帧内预测模式。

6、对宏块中 16 个 4x4 块重复以上 1 到 5 步，获得每一个 4x4 块的最佳编码模式和相应的 $Rdcost_{4x4}$，进而获得整个宏块的率失真开销 $Rdcost_{16x16}$。

7、按类似的方法遍历 4 种 16x16 宏块帧内预测模式，并计算相应的宏块率失真开销 $Rdcost_{16x16}$，选择宏块率失真开销 $Rdcost_{16x16}$ 最小的模式为最佳 16x16 宏块的帧内预测模式。

8、根据 6 和 7 中最小的宏块率失真开销 $Rdcost_{16x16}$，判断亮度宏块的 4x4 或 16x16 帧内预测模式。

9、对每一种 8x8 色度宏块的帧内预测模式（两个色度宏块使用相同的模式）计算相应的率失真开销 $Rdcost_{8x8}$，并重复以上 1 到 8 步，获得相应的宏块组合率失真开销 $Rdcost_{MB}$，选择最小的宏块组合率失真开销 $Rdcost_{MB}$ 作为该宏块组合的最佳帧内预测模式。

由此，完成了一个宏块组合的帧内预测模式选择。显然对一个宏块组合共有 M8*(M4*16+M16)种不同的模式组合，因此共需完成 592 次模式计算，RDO 才能获得最佳的帧内预测模式选择。巨大的计算量使模式选择成为帧内编码的瓶颈

所在。

4.2.2 帧内预测模式选择的快速算法

从上面的分析可以得到以下结论：

- 过多的候选模式是帧内预测模式选择的瓶颈所在。如果可以基于某些信息事先选定最优的和次优的模式作为候选模式，放弃其他的模式检测，则可以提高模式选择的效率。
- 在第二章中，已经详细的讨论了帧内预测以及各个不同预测模式的计算方法。从中可以发现，除 DC 预测外，其余的各种预测模式都是将邻块的像素沿一定的方向做外推来完成帧内预测。假如邻块的像素与沿某一方向的被预测像素有着类似的值，则这时沿这一方向进行预测效果应该相对较好。因此除 DC 预测外，帧内预测的各个模式具有很强的方向性，并且最优模式和次优模式往往具有类似的预测方向，换句话说，最优模式和次优模式通常在预测方向图中是相邻的。这就为寻找帧内预测模式选择的快速算法提供了线索[75]。
- 由于物体边界通常是连续的，沿边缘方向的像素往往具有相似的值。因此如果可以事先获得 NxN 块的边缘方向信息，并使预测方向和这个边缘方向相同时应可获得最优或次优的帧内预测。这就为寻找帧内预测模式选择的快速算法提供了基本思路。

4.2.2.1 基于 Sobel 算子的边缘方向信息检测：

Sobel 算子是常用的边缘检测算子[2]。它有两个卷积核，一个反映垂直方向的变化程度，另一个反映水平方向的变化程度。当原始的视频图像分别与 Sobel 算子的两个卷积核卷积计算后，对像素 $p_{i,j}$（$p_{i,j}$ 不在图像边界上）产生相应的边缘矢量 $D_{i,j}=\{dx_{i,j}, dy_{i,j}\}$。$dx_{i,j}$ 和 $dy_{i,j}$ 分别表示垂直和水平的变化强度，定义如下：

$$\begin{aligned} dx_{i,j} &= p_{i-1,j+1} + 2\times p_{i,j+1} + p_{i+1,j+1} - p_{i-1,j-1} - 2\times p_{i,j-1} - p_{i+1,j-1} \\ dy_{i,j} &= p_{i+1,j-1} + 2\times p_{i+1,j} + p_{i+1,j+1} - p_{i-1,j-1} - 2\times p_{i-1,j} - p_{i-1,j+1} \end{aligned} \tag{4-2}$$

因此边缘矢量的强度近似为：

$$Amp(D_{i,j}) = \left|dx_{i,j}\right| + \left|dy_{i,j}\right| \tag{4-3}$$

方向为：

$$Ang(D_{i,j}) = \frac{180^0}{\pi} \times \arctan\left(\frac{dy_{i,j}}{dx_{i,j}}\right) \tag{4-4}$$

这样，就获得了像素 $p_{i,j}$ 的边缘矢量强度和方向。

4.2.2.2 基于边缘方向信息的 4x4 模式选择快速算法：

对于 4x4 亮度块的帧内预测，共有 8 个基于方向的预测模式。因此可以将整个预测方向图用二分法划分成若干个预测区间。例如，模式 1 与模式 8 和模式 6 的夹角分别为 $\pm 26.6^0$，用二分法将模式 1、8 之间和模式 1、6 之间的夹角等分，这样就可以得到包含预测模式 1 的预测区间：

$a_1 = (13.3^0, -13.3^0)$。

依此类推，

$$\begin{aligned} a_3 &= (-125.8^0, -144.2^0)、\\ a_4 &= (-35.8^0, -54.2^0)、\\ a_5 &= (-54.2^0, -76.7^0)、\\ a_6 &= (-13.3^0, -35.8^0)、\\ a_7 &= (-103.3^0, -125.8^0)、\\ a_0 &= (-76.7^0, -103.3^0)、\\ a_8 &= (13.3^0, 35.8^0)。 \end{aligned} \tag{4-5}$$

注意，a 的下标与该预测区间所对应的帧内预测模式方向一致。这样，可以根据 4x4 块中每一个像素的边缘矢量方向 $Ang(D_{i,j})$所处的预测区间，分别在各个预测区间对该预测区间中的边缘矢量强度 $Amp(D_{i,j})$求和。显然，具有最大的 $\Sigma Amp(D_{i,j})$的那个区间方向代表了这个 4x4 块的边缘方向信息。而这个方向以及它左右相邻的两个方向与最优和次优的预测方向是密切相关的，因此作为候选模式被挑选出来。由于 DC 预测并非是基于方向的预测，并且考虑到图像的边界宏块或块，以及各区间出现边缘矢量强度 $Amp(D_{i,j})$相等的特殊情况，DC 预测也应是候选模式之一。例如，图 4.1：

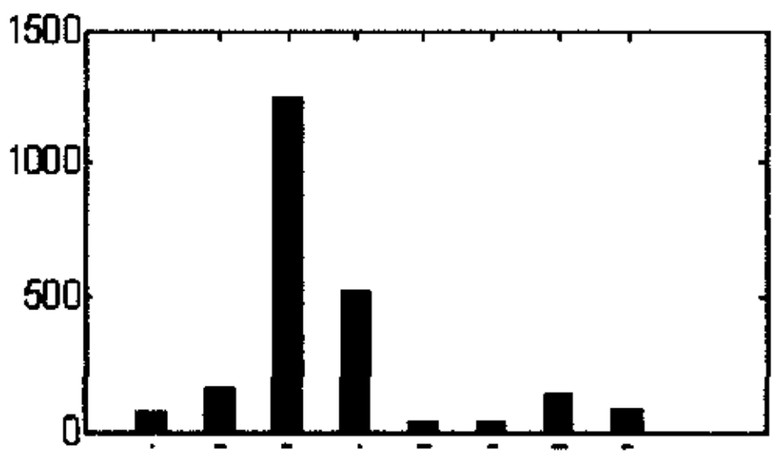

图 4.1 不同模式下边缘矢量强度和

纵坐标表示边缘方向信息 $\Sigma Amp(D_{i,j})$的大小，横坐标表示对应的帧内预测模式号。在图中，由于模式 3 具有最强的边缘方向信息，因此可以选择模式 3 和预

测方向图中相邻的模式 7、模式 8 以及 DC 预测作为候选的模式。这样就将 4x4 帧内预测的 9 种候选模式减少为 4 种。

4.2.2.3 基于边缘方向信息的 16x16 模式选择快速算法：

对于 16x16 亮度宏块帧内预测，其方向预测图如下：

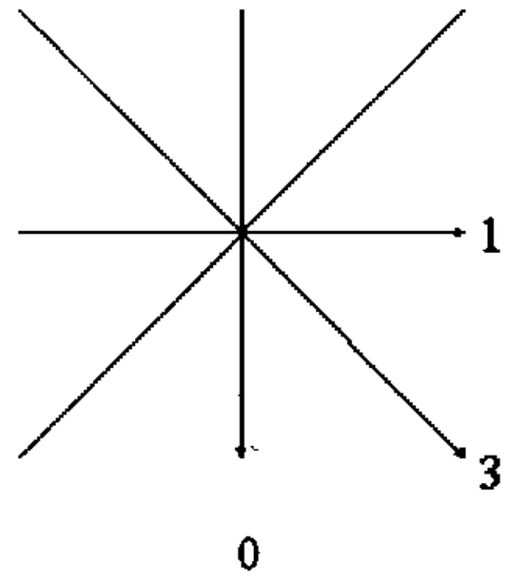

图 4.2 不同模式下边缘矢量强度和

可以采用同样的方法，利用 Sobel 算子获得这个宏块中 256 个像素的边缘矢量强度和方向，并划分出包含各预测模式的预测区间。预测区间如下：

$$a_0 = (-\infty, -67.25^0) \cup (67.25^0, \infty)$$
$$a_1 = (22.25^0, -22.25^0)、 \quad (4\text{-}6)$$
$$a_3 = \Omega - (a_0 \cup a_1)。$$

但由于此时只有 3 种预测方向，换句话说，各个预测方向之间的相关性很小，所以只需考虑最大边缘方向信息所对应的预测模式和 DC 预测作为 16x16 帧内预测的候选模式。这样就可以将 4 种候选模式减少为 2 种。

4.2.2.4 基于边缘方向信息的 8x8 模式选择快速算法：

8x8 色度宏块帧内预测与 16x16 亮度宏块帧内预测基本类似，只是将 DC 预测和垂直预测的模式编号进行了调换，此时垂直预测的模式号为 2，而 DC 预测的模式号为 0。因此 8x8 色度宏块帧内预测模式选择的快速算法与 16x16 亮度宏块帧内预测模式选择的快速算法基本一致。但色度分量包含 U 和 V 两个宏块，因此当两个色度块的最大边缘方向信息不同时，所对应的两个预测模式都应加以考虑，以确保 RDO 的精度。这样就可以将 4 种候选模式减少为 2 种（U 和 V 最大边缘方向信息不同时为 3 种）。

现在完成一个宏块组合的 RDO 计算，最大模式组合数下降为 3*（4*16＋2）＝198 种（若两个色度宏块的边缘方向信息相同，则为 132 种）。由此将 ROD 的候选模式组合数减少为原组合数的四分之一以下。

4.2.3 仿真实验及结果

本文在 H.264/AVC 的参考软件 JVT61e 中对以上帧内预测模式选择快速算法进行了仿真实验。仿真测试的编码参数为：使用率失真优化和哈达玛变换，使用变长编码，支持一个参考帧，编码 CIF 格式的具有不同运动复杂度的 M&D，Foreman，Stefan 序列，编码长度各为 100 帧。仿真测试分以下两组进行。第一组中，所有的编码帧都为 I 帧。第二组为 IPPPP（GOP＝5）编码。测试数据如下表，其中 TIMES 表示采用快速算法后总编码时间下降的比例，PSNR 和 BITS 分别表示 PSNR 和比特率变化幅度，正值表示快速算法比原算法增加，负值表示相应减少。搜索半径：QCIF 为 16，CIF 为 32，CCIR601 为 64。（以下表示类同）

表 4.1 QP=25

序列	第一组			第二组		
	BITS(%)	PSNR(dB)	TIMES(%)	BITS(%)	PSNR(dB)	TIMES(%)
M&D	3.73	-0.05	-61.11	2.95	-0.04	-27.25
Foreman	1.61	-0.04	-60.93	1.09	-0.02	-25.09
Stefan	1.65	-0.13	-60.84	0.91	-0.03	-26.31

表 4.2 QP = 35

序列	第一组			第二组		
	BITS(%)	PSNR(dB)	TIMES(%)	BITS(%)	PSNR(dB)	TIMES(%)
M&D	4.96	-0.00	-58.84	4.23	-0.02	-25.74
Foreman	2.79	-0.03	-59.51	1.74	-0.03	-24.87
Stefan	2.78	-0.10	-60.18	1.87	-0.06	-25.80

仿真结果显示，利用原始图像的边缘方向信息可以有效的减少帧内预测的候选模式数，在全 I 帧编码时，可以节约 60%左右的编码时间，在 IPPPP 编码时可以节约 25%左右的编码时间，效果十分明显。同时，比特率和 PSNR 与原参考软件遍历的模式选择算法相比，基本保持不变。而且仿真结果也显示，该算法对不同运动复杂度的序列，对不同的量化参数 QP 都具有良好的适应性。

4.3 运动搜索快速算法

在基于块匹配的混合视频编码体系中，运动搜索是其中计算复杂性最高的模

块。在绪论中已经对一些非常有效的运动搜索快速算法进行了描述。但H.264/AVC 为了获得更精确的预测和更高的压缩比，采用了多种块模式的运动估计，多参考帧和更高分辨率的运动矢量。虽然这样可以增加预测的精度，提高压缩比，但 H.264/AVC 中运动估计的计算复杂性却因此急剧增加。仿真实验显示在采用简单的编码方式（1 个参考帧，不支持 RDO，搜索半径 16）对运动复杂性很低的 QCIF 序列 Container 编码时，运动估计的开销占 60%以上。而采用复杂的编码方式（5 个参考帧，支持 RDO，搜索半径 64）对运动复杂性较高的 CCIR601 序列 Cheer 编码时，运动估计的开销占 95%以上。因此一个好的搜索算法对提高 H.264/AVC 的编码效率至关重要。

运动搜索快速算法的实质是基于某种策略在减少搜索点数提高搜索效率的同时，尽可能保持编码图像质量不变（与 FS 的情况相比）。但简单的减少搜索点数的算法（例如 TTS、2-D 对数等快速算法）通常都假设匹配误差曲面是单峰的，而实际的视频图像是非常复杂的，常常不能满足这样的假设，因此容易在搜索起始阶段落入局部最小点。为解决局部最小点的问题，通常可以采用相邻块预测的方法获得更好的搜索起点。但有些情况下特别是在运动复杂度高的序列中，相邻块会包含与当前块不同的运动对象，因此也会产生错误的运动矢量[76]。更好的策略是采用覆盖整个搜索区域的全局搜索(Global Search)的概念，即搜索点在整个搜索区域的稀疏网格上选择。虽然这种策略增加了搜索点数，但可以有效的避免在搜索起始阶段落入局部最小点以及可能的错误起始点预测。

根据 H.264/AVC 中的多预测模式和多参考帧，本文在整像素运动搜索中采用了不对称十字和多六角形网格混合搜索快速算法[76]（Unsymmetrical-Cross Multi-Hexagon-Grid Search 简称 UCMHGS）。

4.3.1 整像素运动搜索快速算法

UCMHGS 采用了混合的分级运动搜索策略。主要有以下四步组成：

1) 起始点预测
2) 不对称十字搜索
3) 多六角形网格搜索
4) 扩展六边形搜索

图 4.3 为搜索半径 16 时的示意图。图中假设预测的的搜索起点在（0，0）点。下面对搜索过程的各个步骤进行详细讨论。

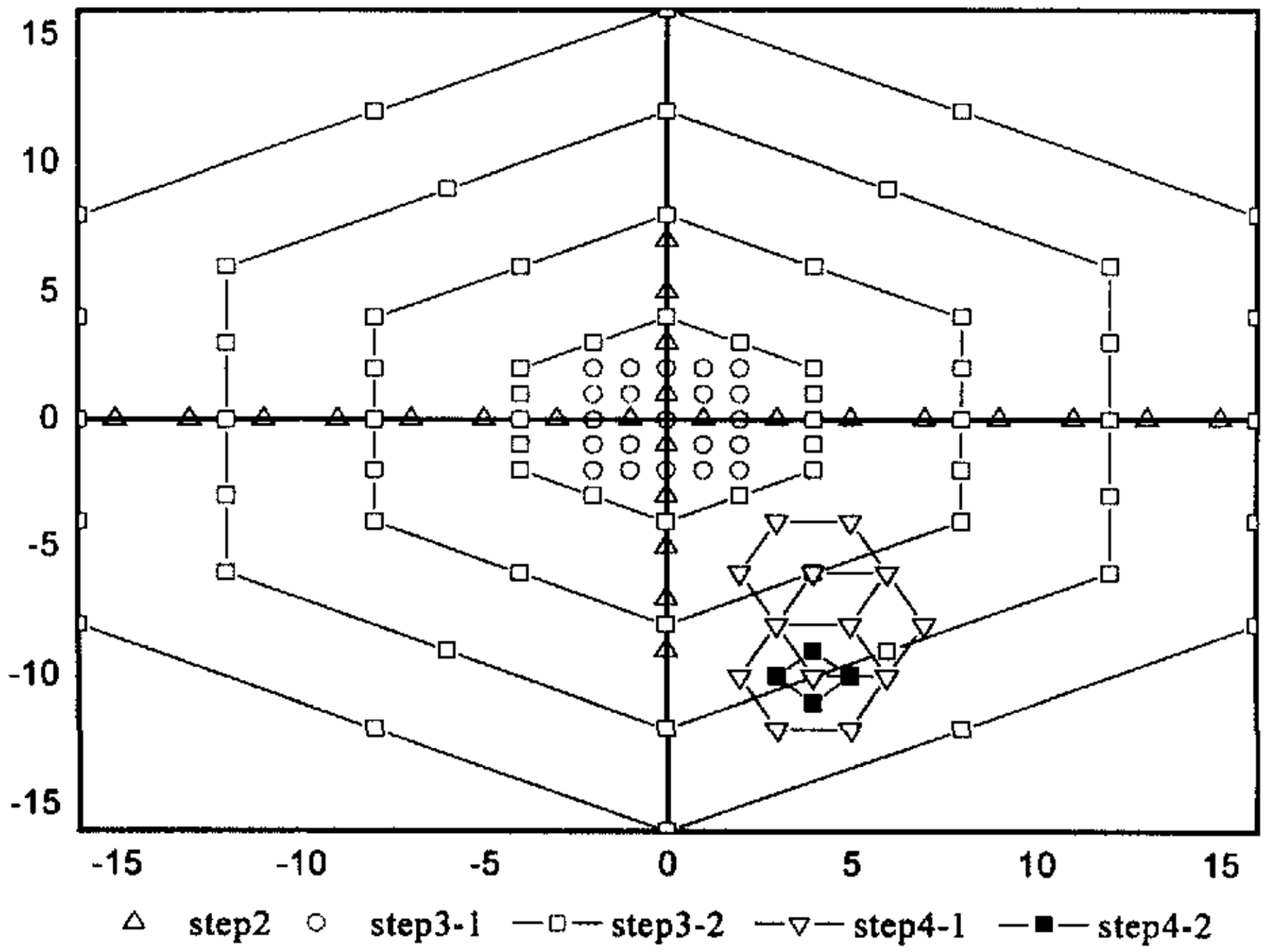

图 4.3 UCMHGS 搜索过程示意图

4.3.1.1 起始点预测：

起始点预测作为一种重要的技术被许多运动搜索快速算法采用。将搜索区域设置在整个搜索窗口中最小块失真（Minimum Block Distortion）点的周围，可以有效的改进运动搜索的性能。在本文的算法中包括了三种不同的方法预测候选起始点，最终选择 SAD 开销最小的作为下一步搜索的起始点。三种方法描述如下：

- 中值预测

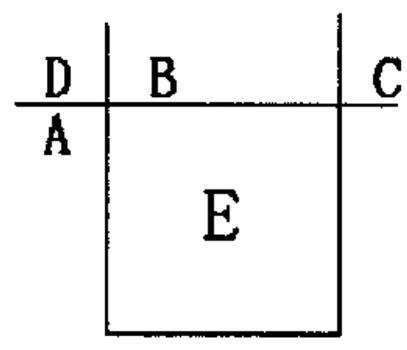

图 4.4 当前块 E 的邻块示意图

如图 4.4 所示，中值预测采用当前块 E 左边邻块 A，上边邻块 B 和右上方邻块 C 的运动矢量来预测当前块的运动矢量，即：

$$MV_{media} = media(MV\_A, MV\_B, MV\_C) \tag{4-7}$$

如果块 A 的运动矢量是不可得的，或者块 B 和块 C 的运动矢量同时不可得，用 0 矢量代替相应的运动矢量值，若块 C 的运动矢量不可得，则用块 D 的运动矢

量代替。

➢ 上层预测

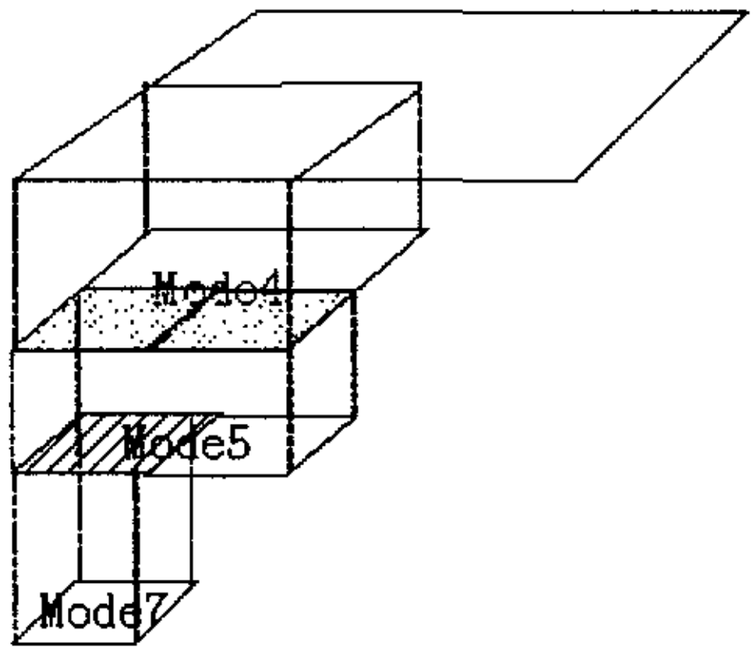

图 4.5 上层预测示意图

在 H.264/AVC 中定义了 7 种不同块形状的帧间预测模式。在当前的参考软件 JM61e 中，首先独立的依次搜索各小块模式（8x8，8x4，4x8，4x4），然后再搜索大块模式（16x16，16x8，8x16）。这样的安排不能充分利用模式之间的相关性。因此按照块的尺寸，从大到小，重新安排了各个模式的搜索次序。这样就可以将上层较大块的运动矢量作为下层运动矢量的预测。如图 4.5，Mode4(8x8)是 Mode5(8x4)，Mode6(4x8)和 Mode7(4x4)的上层，依此类推。可以得到预测的运动矢量如下：

$$MV_{pre\_up} = MV_{uplayer} \tag{4-8}$$

➢ 对应块预测

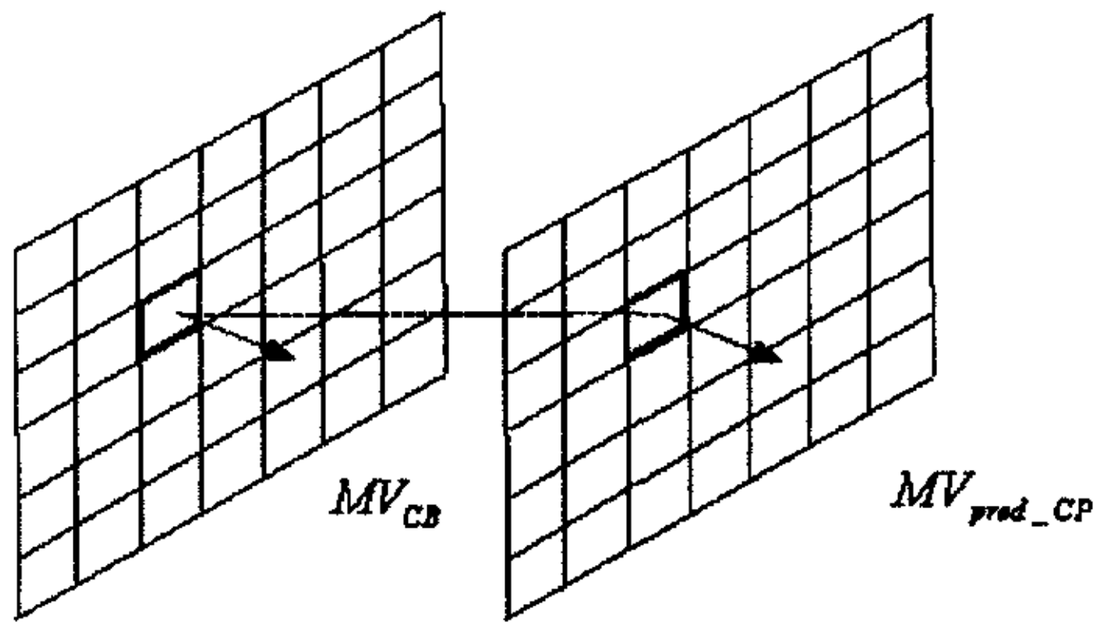

图 4.6 对应块预测示意图

对于自然视频来说，除非发生场景切换，否则移动物体的运动轨迹是连续的。换句话说，运动矢量有很强的时域相关性。利用这个特性，可以用当前帧的前一

编码帧中与当前块相对应块的运动矢量作为当前块运动矢量的预测。如图 4.6。可以得到预测的运动矢量如下：

$$MV_{pre\_cp} = MV_{cb} \tag{4-9}$$

需要注意的是 MV_{pred_up} 不能用于对 Mode1(16x16)进行预测，而如果当前帧是图片组(Group of Picture)中的第一个 P 帧时，MV_{pred_cp} 是无法使用的。因此对 Mode1 使用中值预测、(0, 0)运动矢量、图 4.4 中的三个邻块 A，B，C 的运动矢量作为候选的预测。而对于其它模式，则可以采用中值预测、(0, 0)运动矢量、以及可以使用的上层预测和对应块预测作为候选的预测。

4.3.1.2 不对称十字搜索：

在自然视频序列中，水平方向的运动往往比垂直方向的运动大的多。基于这样的认识，不对称十字搜索将水平方向的搜索范围设为 W，而垂直方向的搜索范围设为 W/2，如图 4.3 的第 2 步。对于一个运动相对简单的（特别是在垂直方向的运动较小）视频系列，不对称十字搜索几乎可以非常精确的预测最佳运动矢量，给下一步一个准确搜索起点。当然如果编码器的实际应用领域中可能会经常遇到垂直运动剧烈的序列，也可以将垂直方向的搜索范围设为 W。

在不对称十字搜索结束后，选取具有最小开销的运动矢量作为下一步的搜索起点。

4.3.1.3 多六角形网格搜索：

六角形网格由 16 个点组成，并且基于同样的认识，水平方向的搜索点数多于垂直方向，见图 4.7。

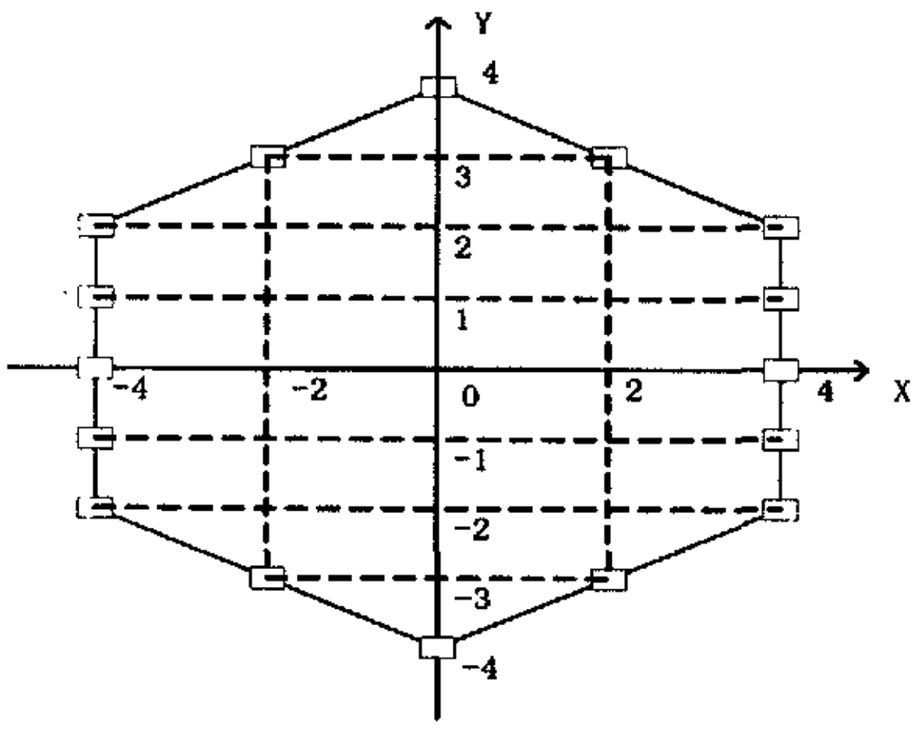

图 4.7 16 个点的六角形网格方案

多个六角形网格以不同的尺度（W/4 到 W）由内向外扩展搜索，覆盖几乎全部

的搜索空间，如图 4.3 所示。

多六角形网格搜索由两步组成。由于不对称十字搜索可以给出较精确的搜索起点，因此第一步先以搜索半径 2 在当前搜索中心进行全搜索，如图 4.3 的第 3-1 步。针对运动剧烈并且复杂的情况，在第二步进行以上的多六角形网格搜索，如图 4.3 的第 3-2 步。

在多六角形网格搜索结束后，选取具有最小开销的运动矢量作为下一步的搜索起点。

4.3.1.4 扩展六边形搜索：

显然，多六角形网格搜索可以获得不同精度的最佳运动矢量，距多六角形网格的搜索中心越远，搜索精度越低。因此需要采用一些基于中心的搜索方法来改进其精度。另一个原因是对于 Mode7(4x4)这样的小块来说，由于尺寸越小，块之间的运动相关性越高，因此在第一步预测的运动矢量已经具有很高的精确度。可以跳过不对称十字搜索和多六角形网格搜索，直接进入这一步。小块模式的选择可以在质量和速度间折中选取。

六边形搜索算法可以在较少的搜索点数上获得与钻石搜索算法相同的运动矢量。扩展六边形搜索与六边形搜索基本相同，差别在于当大小六边形转换时，搜索继续进行，直至具有最小块失真的点位于最新六边形的中心。

4.3.2 分数像素运动搜索快速算法

由于 H.264/AVC 更高的分数像素预测精度，因此在整像素搜索算法加速后，减少分数像素运动估计的计算量就变得有意义和必须的了。在 JVT 的参考软件中，分数像素的运动搜索为：首先搜索最佳整像素点周围的 8 个 1/2 像素点，获得最佳半像素位置。然后搜索最佳半像素点周围的 8 个 1/4 像素点，获得最佳 1/4 像素位置。

虽然整像素运动搜索的匹配误差曲面常常不满足单峰的假设，但在获得最佳整像素预测后，当前块和预测块间存在极强的相关性[77]，围绕这个最佳整像素位置进行的分数像素运动搜索的匹配误差曲面一般都可以满足单峰的假设。本文采用了基于中心的分数像素搜索快速算法(Center-Based Fractional Pel Search 简称 CBFPS)。算法由两步组成。

1) 分数像素运动矢量预测
2) CBFPS 搜索

4.3.2.1 分数像素运动矢量预测：

在获得最佳整像素运动矢量后，很显然此时的分数像素运动矢量具有很强的空间相关性。因此在获得邻块的分数像素运动矢量信息后，可以用整像素中相同的方法对分数像素运动矢量进行中值预测。

$$frac\_pred\_mv = (pred\_mv - mv)\%\beta \tag{4-10}$$

所有的运动矢量以相应的分数像素为单位表示。*Pred_mv* 为当前块包含分数像素运动矢量信息的中值预测，方法与 4.3.1.1 中描述的相同。*mv* 为已获得的整像素运动矢量。β 分别取 2，4，8 以获得相应的分数像素运动矢量。

4.3.2.2 CBFPS 搜索：

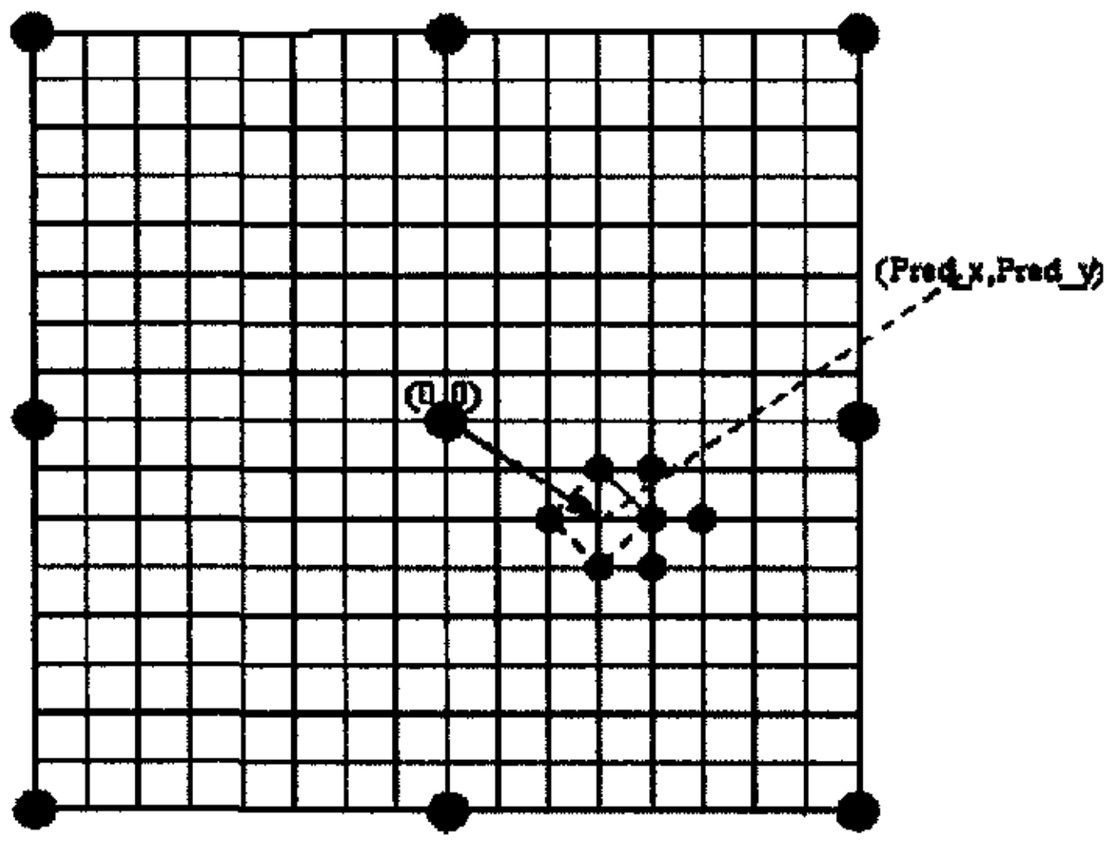

图 4.8 CBFPS 搜索过程示意图

CBFPS 算法示意图见图 4.8。它由以下三步组成。

第一步：计算预测的搜索中心(Pred_x, Pred_y)和原始搜索中心(0, 0)的开销，选择具有最小匹配误差的最为下一步的搜索起点。

第二步：采用钻石搜索算法进行搜索。如果最小块失真点位于当前钻石的中心，停止搜索，进入第三步。否则选择最小块失真点作为新的搜索起点重复这一步。

第三步：选择上一步获得的位于当前搜索钻石中心的最小块失真点作为最佳分数像素位置。并取得当前块完整的运动矢量。

4.3.3 仿真实验及结果

实验中采用了 JVT 的参考软件 JM61e 进行了仿真测试，支持 7 种块模式、RDO 优化、UVLC 熵编码。搜索半径：QCIF 为 16，CIF 为 32，CCIR601 为 64。编码序列为 IPPPP，编码 100 帧进行数据采样。本文选择了不同格式和复杂度的 5 个序列，它们分别是：Carphone，Container，Akyio，Mobile、Cheer 序列。实验结果见表 4.3。TIMES_ME 表示运动估计模块所占时间下降的比例。

表 4.3 运动搜索快速算法统计表

序列	格式	TIMES(%)	TIMES_ME(%)	PSNR(dB)	BITS(%)
Container	QCIF	-19.8	-80.9	0	0
Carphone	QCIF	-21.7	-84.3	-0.02	-0.18
Akyio	CIF	-39.6	-76.0	-0.01	-0.21
Mobile	CIF	-51.6	-86.2	0	-0.29
Cheer	CCIR601	-78.7	-86.9	0	0.38
平均	—	-42.28	-82.86	-0.006	0.06

从表 4.3 可以发现，本文采用的算法可以有效的减少总的编码时间，并且保持比特率和 PSNR 几乎没有变化，特别是比特率在某些情况下还有所降低。另一方面由于对不同格式和不同复杂度的序列编码时，运动估计消耗时间占总编码时间的比例不同，因此总的时间节余是变化的。图像尺寸越大，运动复杂度越高，效果越好，平均 40%左右。但对于运动估计模块本身而言，效率提高都在 80%左右，具有良好的适应性。

4.4 帧间预测模式选择

4.4.1 多块模式的帧间预测

精确的预测可以有效的降低原始图象和预测图象之间的误差。出于这个原因，H.264/AVC 采用了多块尺寸的运动估计和补偿以获得更好的压缩性能。对一个宏块，它可能包含多个对象，而这些对象的运动方向可能不同。因此仅使用一个运动矢量描述一个宏块中所有对象的运动也许是不够的。仅使用一个运动矢量，只有宏块中的部分区域可以被很好的运动补偿，而由于宏块中剩余部分的失配，导致残差能量过大。在另一种情况下，一个宏块中包括多种不同的纹理，而这些纹理相对的运动状况并不相同，这样就有可能出现一部分纹理用整像素运动估计可以获得很好的匹配，而另一部分纹理则在分数像素（1/2 或 1/4 像素）运

动估计中获得最佳匹配。仅使用一个整像素或分数像素的运动矢量都很难给出这样宏块最佳匹配。如果允许多块运动补偿，宏块可以被分割成更小的区域。每一个小区域都将产生一个运动矢量，指向预测图象的最佳匹配区域，给不同的对象或不同的纹理区域最佳的预测。在 H.264/AVC 中，支持 7 种不同形状的块尺寸，第二章中已经做了讨论。利用 RDO 技术，计算各个不同块模式的率失真开销函数 J 如下：

$$J(m,\lambda_{MOTION}) = SAD(s,c(m)) + \lambda_{MOTION} * R \tag{4-11}$$

这里 s 表示原始视频信号，$c(m)$表示编码的视频信号，$m=(m_x,m_y)^T$ 表示运动矢量，而 λ_{MOTION} 是拉各朗日乘子。R 表示编码运动信息所用比特数。SAD 计算如下：

$$SAD(s,c(m)) = \sum_{x=1,y=1}^{B,B} \left| s[x,y] - c[x-m_x, y-m_y] \right| \tag{4-12}$$

这里 B=16,8,4。这样就可以获得不同块模式编码的率失真开销，选取具有最小宏块率失真开销的模式作为最佳匹配。

4.4.2 大块模式选择快速算法

虽然 H.264/AVC 支持 7 种不同块尺寸的帧间编码模式，在对不同的视频序列编码时，各个模式所占的比例相差甚远。统计结果见表 4.4[78]

表 4.4 编码模式统计表(百分比)

序列	16x16	16x8	8x16	8x8	intra
Container	92	1	1	4	2
Hall Monitor	90	1	1	7	1
M&D	89	3	3	4	1
Weather	87	1	2	9	1
Silent	83	3	3	10	1
Table Tennis	76	4	4	13	3
Foreman	64	8	8	17	3
Coastguard	57	6	6	30	1
Mobile	49	6	7	37	1
Stefan	47	6	5	38	4
平均	73	4	4	17	2

表中的 8x8 包括 8x8、8x4、4x8 和 4x4 所有的小块模式。在 7 种不同的块模式中，16x16 所占比例最高。特别是在运动复杂度较低的序列中。而对于 Stefan 等运动相对复杂的系列，小块模式（表中的 8x8）所占比例明显增加。显然，这也符合

多块模式帧间编码有利于实现运动隔离，提高预测精确度的初衷。从平均结果看，16x16 所占比例超过 70%。而大块模式（指 16x16、16x8 和 8x16）中其余两种所占比例各为 4%，小于此时小块模式的 17%。因此可以在大块模式的选择中进行如下处理。如果模式 16x16 足够好，则早期中止，跳过模式 16x8 和 8x16。否则进行块分割[79]。算法描述如下：

1、初始化

定义 3 个变量如下：

T：早期中止阈值

SAD_{SUM}：选择模式 16x16 编码宏块的 SAD 累加

N1：选择模式 16x16 编码宏块的宏块数累加

设置：T=0, SAD_{SUM} =0, N1=0

遍历每一个宏块，完成一下 2 步。

2、早期中止

对 16x16 的宏块，使用全搜索或同类的快速搜索算法发现最佳整象素位置，并且计算此时模式 16x16 的 SAD：$S1_{min}$。如果 $S1_{min}$<T，选择模式 16x16 作为最终的块类型并用下面公式更新 T。

$$SAD_{SUM} = SAD_{SUM} + S1_{min} \quad (4\text{-}13)$$

$$N1=N1+1 \quad (4\text{-}14)$$

$$T= SAD_{SUM} /N1 \quad (4\text{-}15)$$

否则到第三步

3、块分割

对第二步获得的最佳整象素位置，每个整像素点都有 8 个相邻的 1/2 象素点，如图 4 所示。

a b c

d I e

大写字母 I 表示整像素

f g h 小写字母 a 到 h 表示 1/2 像素

图 4.9 分数像素位置

则对于每一个整像素点都可以划分如下：将 I 的上部划分为 h1，下部相应为 h2。I 的左面为 v1，右面相应为 v2。对整个宏块中所有的整像素划分完毕后，宏块共包含 4 个区域，H1 为全部的 h1 之和，H2 为全部的 h2 之和，V1 为全部的 v1 之和，V2 为全部的 v2 之和，并计算 SADs。每一个区域的最大 SAD

差计算如下：

$$mSAD(r) = \max[SAD(I, r) - SAD(p, r)]\ p \in a—h \tag{4-16}$$

这里 r 表示区域，p 是 8 个 1/2 象素位置。令

$$mSAD\_H = mSAD(H1)+mSAD(H2) \tag{4-17}$$

$$mSAD\_V = mSAD(V1)+mSAD(V2) \tag{4-18}$$

三个条件 C1,C2,C3 将被检查

C1：如果 mSAD _H> mSAD _V，选择模式 2 进行运动估计，并在模式 1 和模式 2 之间选择最佳模式。否则检查系数 C2。

C2：如果 mSAD _H< mSAD _V，选择模式 3 进行运动估计，并在模式 1 和模式 3 之间选择最佳模式。否则检查系数 C3

C3：如果 mSAD _H= mSAD _V=0，选择模式 1 为最终的块类型，不需要做进一步的运动估计。否则对模式 2 和模式 3 完成运动估计，选择最佳模式为最终块类型。

4.4.3 仿真实验及结果

实验中采用了 JVT 的参考软件 JM61e 进行了仿真测试。编码参数同前。根据表 4.4 中 16x16 帧间预测模式的比例不同，本文选取其中具有代表性的 4 个标准 CIF 测试序列，它们分别是：M&D， Foreman，Stefan 以及 Coastguard 序列。实验结果见表 4.5。

表 4.5 大块模式选择快速算法统计表

序列	TIMES(%)	PSNR(dB)	BITS(%)
M&D	-11.85	-0.03	0.56
Foreman	-15.65	-0.04	1.87
Stefan	-15.09	-0.01	1.33
Coastguard	-13.07	-0.02	0.64
平均	-13.915	-0.025	1.1

在 H.264/AVC 中，基于 RDO 的帧间预测多块模式选择的基本思想依然是采用遍历的方式搜索所有 7 种可能的块模式，所以在这个过程中必然包含大量的冗余操作。统计结果表明，在不同复杂度的序列中大块模式总是占有相对多数，因此最佳的模式选择算法应从 16x16 逐步向更小的块进行，并且在相对较大的块中找到足够好的匹配后，立刻跳出其后所有的较小块模式搜索。但这样的中止条件很难选择。特别是在运动复杂的序列中，小块模式的比例迅速增加，简单的跳过

小块模式搜索容易造成图像质量下降过快。因此本文采用了这种折中的方式，只对大块模式进行选择。表 4.5 的数据表明，该算法对于不同复杂度的序列具有良好的适应性，可以明显的提高编码效率，同时保持 PSNR 和比特率基本不变。对于 M&D 序列，由于其运动复杂度最低，运动估计所占比例明显小于其他序列，因此虽然它的 16x16 编码模式比例最高，但在编码时间下降的比例上并不会优于其它的复杂序列。

4.5 零块检测及其在运动搜索中的应用

4.5.1 H.263 中的全零块检测技术

在低码率视频应用中，常见的是运动缓慢的具有静止背景的头肩序列，可以获得较精确的运动估计。因此帧间编码时，经过运动估计的残差块信号绝对值通常很小。这样的残差块信号经过变换，量化后的系数常常为零，即存在大量的全零块。如果可以在变换和量化前检测到这种全零块，就可以省去变换、量化以及反变换和反量化的过程，降低编码的计算复杂性，提高编码效率。

H.263 是一个非常成功的低码率视频压缩编码标准。在 H.263 的发展过程中，许多专家学者对提高编码效率进行了研究。1997 年，Alice YU[80]提出了一种基于离散 DCT 变换系数统计特性的全零块检测方法。主要思想是：由于 H.263 的量化步长恒定等于 $2Q$（Q 是量化系数），因此当残差块的变换系数 $F(u, v)$ 小于 $2Q$ 时，量化结果为零。由于在大多数情况下满足 DC 变换系数 $F(0, 0)$ 大于剩余的 AC 系数，所以近似的认为若 $ABS[F(0, 0)] < 2Q$ 成立（$ABS[\ ]$：绝对值运算），则检测到该块为全零块。因为 8x8 的 DCT 变换的 DC 系数为：

$$F(0,0) = \frac{1}{8}\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) \tag{4-19}$$

所以 $ABS[\ \Sigma\Sigma f(x,y)] < 16Q$ 成为检测全零块的条件。由于统计特性的误差存在，为减少误判，Alice Yu 将阈值减小为 8Q。

周璇[81]进一步研究了离散 DCT 变换的特性。提出了一种没有错检的全零块检测最优方法。对于 8x8 的 DCT 变换有：

$$F(u,v) = \frac{1}{4}k_u k_v \sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right] \tag{4-20}$$

由于以下不等式成立：

$$k_u \cos((2x+1)u\pi/16) \le \cos(\pi/16) \tag{4-21}$$

$$k_v \cos((2y+1)v\pi/16) \le \cos(\pi/16) \tag{4-22}$$

因此可得：

$$F(u,v) \le \frac{1}{4}\sum_{x=0}^{7}\sum_{y=0}^{7} ABS(f(x,y))\cos(\frac{\pi}{16})\cos(\frac{\pi}{16}) < \frac{1}{4}\sum_{0}^{7}\sum_{0}^{7} ABS(f(x,y)) \tag{4-23}$$

所以

$$\sum_{x=0}^{7}\sum_{y=0}^{7} ABS(f(x,y)) < 8Q \tag{4-24}$$

是检测全零块的无错检最优阈值。

4.5.2 H.264/AVC 中的全零块检测技术

4.5.2.1 H.264/AVC 的变换和量化：

H.264/ AVC 采用了近似 DCT 变换的 4x4 的整形变换，采用整数运算，避免了正反变换失配，而且更小的块变换也提高了变换精度，减小了解码图像的块效应。当前标准中采用的是一种改进的整形变换，变换矩阵的系数只有 1 和 2，使得在大多数硬件平台上可以只用加和平移操作，避免了乘操作，加速了执行效率。

在 H.264/ AVC 中，为了充分利用视频图象的空间相关性，对帧内编码也引入了预测。而预测信号特别是I帧的预测信号对偏差的敏感性要求变换必须是无偏的，否则将造成误差积累，使图象质量迅速下降。但视频序列的空间相关性远小于时间相关性。因此运动补偿的残差信号在统计上更接近平稳信号，加之帧内编码块的数量较少，因此本文仅对帧间编码块进行全零块检测。对 4x4 的预测残差信号 **P**，变换系数 **C** 为：

$$C(x,y) = \sum_{x=0}^{3}\sum_{y=0}^{3} \mathbf{H}P(x,y)\mathbf{H}^{\mathrm{T}} \quad \text{其中 } \mathbf{H} \text{ 为：} \mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \tag{4-25}$$

$$C(x,y)_Q = \left(C(x,y) * QE[q_{rem}][x][y] + f\right)/2^{qbits} \tag{4-26}$$

其中 $qbits$=15+QP/6，q_{rem}=QP%6，f=(1<<$qbits$)/6，***QE*** 是量化参数矩阵。对于给定的 q_{rem}，$QE[q_{rem}][0][0]$为 $QE[q_{rem}][x][y]$中的最大值。因此可令最小量化步距为：

$$A(QP) = \frac{5}{6}((2^{qbits})/QE[q_{rem}][0][0]) \tag{4-27}$$

由于量化前后的变换系数都是整数，因此，对 4x4 变换块的任一变换系数 $C(x, y)$，满足 $C(x, y)<A(QP)$时 $C(x, y)_Q$ 为零。

4.5.2.2 H.264/AVC 的全零块检测门限：

由柯西一施瓦兹不等式[82]，公式（7）可展开得：

$$C(x,y) \le \left|\sum_{x=0}^{3}\sum_{y=0}^{3}P^2(x,y)\right|^{\frac{1}{2}} * \left|\sum_{x=0}^{3}\sum_{y=0}^{3}\left(\mathbf{H}\mathbf{H}^T\right)^2\right|^{\frac{1}{2}} = \sqrt{232}\left|\sum_{x=0}^{3}\sum_{y=0}^{3}P^2(x,y)\right|^{\frac{1}{2}}$$

$$< 16\left|\sum_{x=0}^{3}\sum_{y=0}^{3}P^2(x,y)\right|^{\frac{1}{2}} < 16 * \frac{1}{2\sqrt{2}}\sum_{x=0}^{3}\sum_{y=0}^{3}\left|P(x,y)\right| \tag{4-28}$$

所以当

$$16 * \frac{1}{2\sqrt{2}}\sum_{x=0}^{3}\sum_{y=0}^{3}\left|P(x,y)\right| < A(QP) \tag{4-29}$$

即

$$\sum_{x=0}^{3}\sum_{y=0}^{3}\left|P(x,y)\right| < \frac{\sqrt{2}}{8}A(QP) \tag{4-30}$$

成立时，4x4 块中的 16 个变换系数的量化值全部为零。因此可以获得了 H.264/AVC 中无错检的全零块检测门限。

由于以上推导过程中的不等号传递的条件过于宽泛，因此全零块检测门限公式是一种弱势判断。为了提高零块检测的比例，对以上全零块检测门限修正如下：

$$\sum_{x=0}^{3}\sum_{y=0}^{3}\left|p(x,y)\right| < \frac{\sqrt{2}}{2}A(QP) \tag{4-31}$$

4.5.2.3 仿真实验及结果：

实验中采用了 JVT 的参考软件 JM61e 进行了仿真测试。编码参数同前。本文选取具有不同复杂性的 3 个标准 CIF 测试序列，它们分别是：典型的头肩序列 akiyo，具有中等运动强度的 foreman 以及剧烈运动的 stefan 序列。使用 intel 的 Vtune 性能测试软件对 4x4 整形变换和量化模块占全部运行时间的比例进行统计。量化系数 QP 分别取：16，24，30，36，42 和 48。实验结果见下表。其中 ZERO 表示正确检测到的零块占全部零块的比例，ERROR 表示误检的非零块占检测到的全部零块的比例， THRANS 表示通过全零块检测使 4x4 变换和量化模块运行时间与该模块原运行时间相比下降的幅度。

表 4.6 akiyo 序列

QP	ZERO	ERROR	THRANS	PSNR	BITS
16	12.5%	0	11%	0	0
24	67.1%	0	34%	0	0
30	81.2%	0	48%	0	0
36	86.4%	0	55%	0	0
42	90.9%	0	70%	0	0
48	93.6%	0	74%	0	0

表 4.7 foreman 序列

QP	ZERO	ERROR	THRANS	PSNR	BITS
16	25.7%	0	4%	0	0
24	34.2%	0	9%	0	0
30	62.4%	0	24%	0	0
36	78.7%	0	45%	0	0
42	86.4%	0	58%	0	0
48	90.1%	0	67%	0	0

表 4.8 stefan 序列

QP	ZERO	ERROR	THRANS	PSNR	BITS
16	9.2%	0	2%	0	0
24	60.2%	0	6%	0	0
30	73.5%	0	11%	0	0
36	75.4%	0	21%	0	0
42	73.2%	0	32%	0	0
48	74.0%	0	45%	0	0

从上面的试验数据可以发现，本文提出的全零块检测技术没有误检，因此不会造成 PSNR 的下降和比特率的增加。并且基本保持随 QP 的增加而自适应的修正检测门限，提高检测比例，具有良好的自适应性。同时可以有效的减少变换、反变换、量化和反量化在总编码时间中所占的比例。对典型的头肩序列 akiyo，在 QP 大于等于 30 时，全零块的检测比例超过 80%，并可以减少近 50%以上的变换和量化的时间，效果最好。而中等运动强度的 foreman 序列，由于在局部运动中存在运动补偿效果较差的转动和垂直方向的运动，因此效果较 akiyo 差。但当 QP 大于 30 时，也有超过 60%的全零块检测率和近 30%以上的变换量化时间节省。在 stefan 序列中，由于存在剧烈的水平运动，编码块中全零块的数目明显减少，因此相对的效果较差，但在低比特率的条件下，也可以达到 70%以上的全零块检测率。并能部分减少变换和量化部分所占时间。因此本文提出的全零块检测技术在中低运动强度特别是低运动强度的头肩序列中可以取得令人满意的

检测结果，显著减少变换和量化模块所占时间，降低编码器的计算复杂性。

4.5.3 基于全零块检测的运动估计快速算法

虽然运动估计的目的是给出当前块的最佳匹配，但基于全零块检测运动估计快速算法的核心思想是够用就好。图 4.10 为基于全零块检测运动估计快速算法的流程图。由于分数像素运动搜索的复杂性远小于整像素运动搜索，而且对于提高预测精度作用明显。因此此时的全零块检测是在整像素运动搜索阶段进行的。

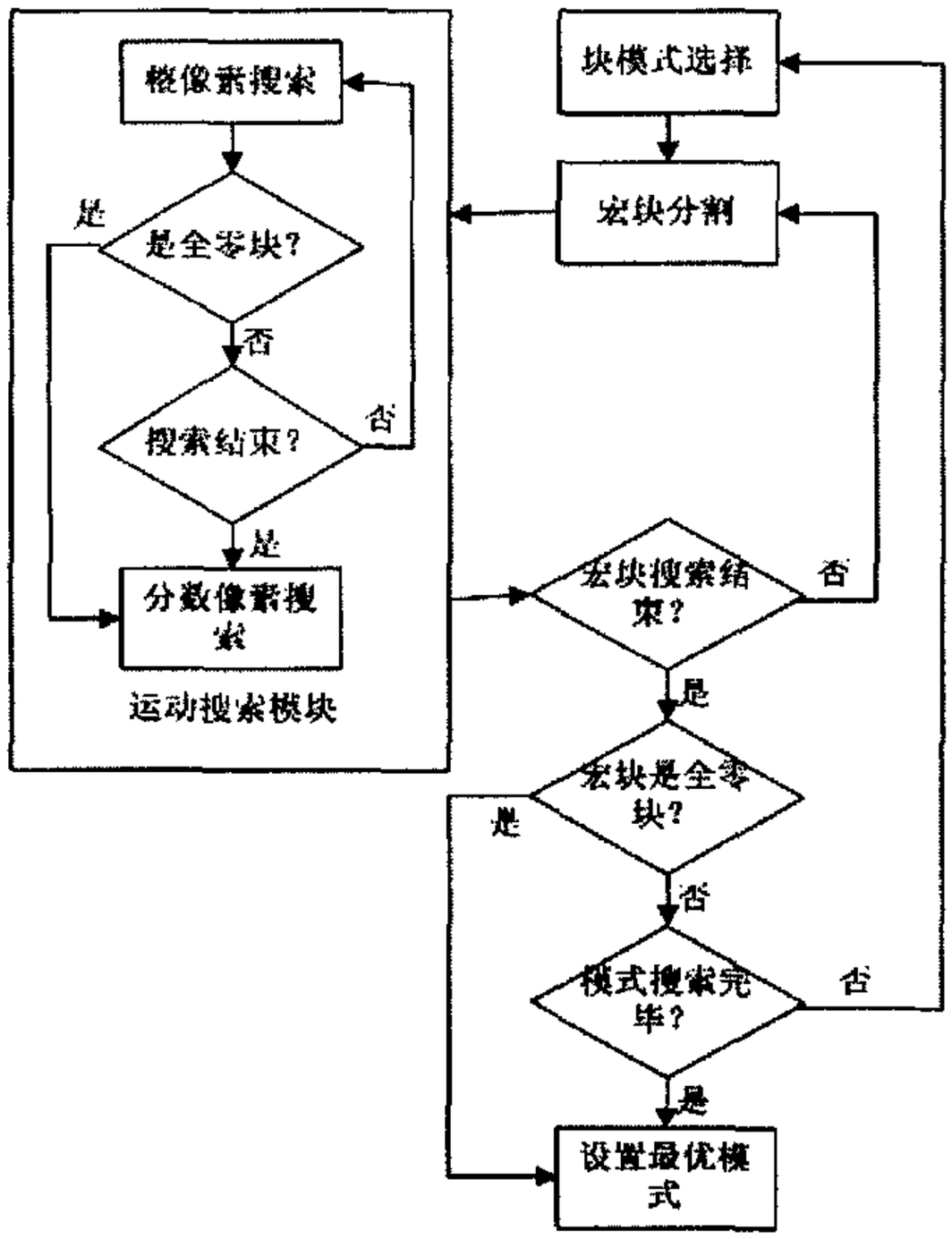

图 4.10 基于全零块检测的运动搜索

需要说明的是由于本文给出的全零块检测门限基于 4x4 的块，而 H.264/AVC 中共有 7 种不同形状的帧间预测模式，所以本文在运动搜索阶段和模式选择阶段对全零块检测结果的处理方式是不同的。

在整像素运动搜索时，如果当前预测模式为 NxM 的块，则全零块检测基于 NxM 的块。搜索是否停止的判据如下：

$$SAD_{N\times M} \le K_{block} * T_{ZERO} \tag{4-32}$$

其中 K_{block} 为 NxM 块中 4x4 块的数目。显然本算法没有逐一仔细判断是否 NxM 块中所有的 4x4 块都为全零块，而是简单的把检测门限相应放大。由于帧间预测

的残差信号通常比较平稳，因此这样处理可以有效的提高搜索效率。

在模式选择阶段，全零块检测是基于宏块的。对于预测模式 $Mode_{NxM}$，令 Num_{MB} 为宏块中 NxM 块的个数，当宏块中全部 Num_{MB} 个 NxM 块在运动搜索完成后都是全零块，则认为该宏块为全零块，这种预测模式 $Mode_{NxM}$ 为最优预测模式，停止模式选择。例如对于 $Mode_{16x8}$，当宏块中两个 16x8 块完成运动搜索，且都满足(4-32)式，则认为该宏块为全零块，$Mode_{16x8}$ 为该宏块的最优编码模式，跳过其它模式的选择过程，否则模式选择继续进行。

4.5.3.4 仿真实验及结果：

仿真测试环境和编码参数与前一实验相同。由于全零块检测的比例与量化参数 *QP* 直接相关，因此测试了不同 *QP* 下基于全零块检测运动估计快速算法的性能。测试结果见表 4.9，表 4.10 和表 4.11。

表 4.9 stefan 序列

QP	TIMES(%)	PSNR(dB)	BITS(%)
16	-4.96	-0.01	0
24	-13.07	-0.02	-0.061
30	-22.09	-0.03	0.126
36	-35.45	-0.06	0.322
42	-51.20	-0.09	-0.182
48	-67.59	-0.13	0.499

表 4.10 foreman 序列

QP	TIMES (%)	PSNR(dB)	BITS(%)
16	-10.89	0	-0.008
24	-25.46	-0.01	0.026
30	-46.69	-0.05	0.162
36	-58.79	-0.14	0.825
42	-64.00	-0.17	1.133
48	-64.96	-0.24	1.292

表 4.11 akiyo 序列

QP	TIMES (%)	PSNR(dB)	BITS(%)
16	-31.79	0	0.002
24	-48.68	-0.01	-0.158
30	-54.65	-0.01	-0.204
36	-58.99	-0.03	-0.316
42	-60.34	-0.07	-0.171
48	-60.50	-0.04	-0.265

由于 H.264/AVC 编码器巨大的计算开销，加之 4x4 整数变换的计算复杂度较之 8x8DCT 变换有所减小，因此仅在变换和量化阶段使用全零块检测在提高编码器整体效率上效果十分有限。但将全零块检测与运动估计模块相结合，可以得到较好的效果。尤其是在 *QP* 较大，而序列的运动复杂度较低的情况下效果非常明显。例如在 Akiyo 序列，即使 *QP* 为 16 时，也可以减少 30%以上的编码时间。但对于简单的序列，其编码时运动估计所占时间相对较少，因此在高量化值时也只能减少 60%左右的编码时间。对于复杂序列，随着 *QP* 的增加，编码效率逐步提高，在高量化值时也能达到 60%以上。同时也应该注意到基于全零块检测而获得的运动矢量通常不是最优的，因此一方面造成 PSNR 随 QP 的增加而下降，另一方面在率失真模式选择时帧内预测块的数量也因此而增加，使得比特率在高量化值时增加明显，但总体上还在可以接受的范围内。

4.6 综合优化仿真实验及结果

本节中将前述的优化算法进行了综合，并在此基础上进行了相关仿真测试。测试条件同前所述。测试结果如下。

表 4.12　QCIF 序列综合优化编码器性能统计表

序列	QP	TIMES (%)	PSNR(dB)	BITS(%)
Container	25	-77.85	-0.09	2.897
	35	-76.64	-0.11	1.894
	45	-73.96	-0.09	2.725
Foreman	25	-77.21	-0.05	4.492
	35	-76.38	-0.10	1.056
	45	-72.77	-0.03	2.081

表 4.13　CIF 序列综合优化编码器性能统计表

序列	QP	TIMES (%)	PSNR(dB)	BITS(%)
Akiyo	25	-86.61	-0.11	5.653
	35	-85.78	-0.07	6.323
	45	-83.66	-0.02	9.724
Mobile	25	-82.62	-0.10	0.956
	35	-83.32	-0.06	1.833
	45	-80.83	-0.03	3.376

表 4.14 CCIR601 序列综合优化编码器性能统计表

序列	QP	TIMES (%)	PSNR(dB)	BITS(%)
Cheer	25	-92.93	-0.09	1.659
	35	-91.78	-0.08	2.858
	45	-90.28	-0.08	4.767

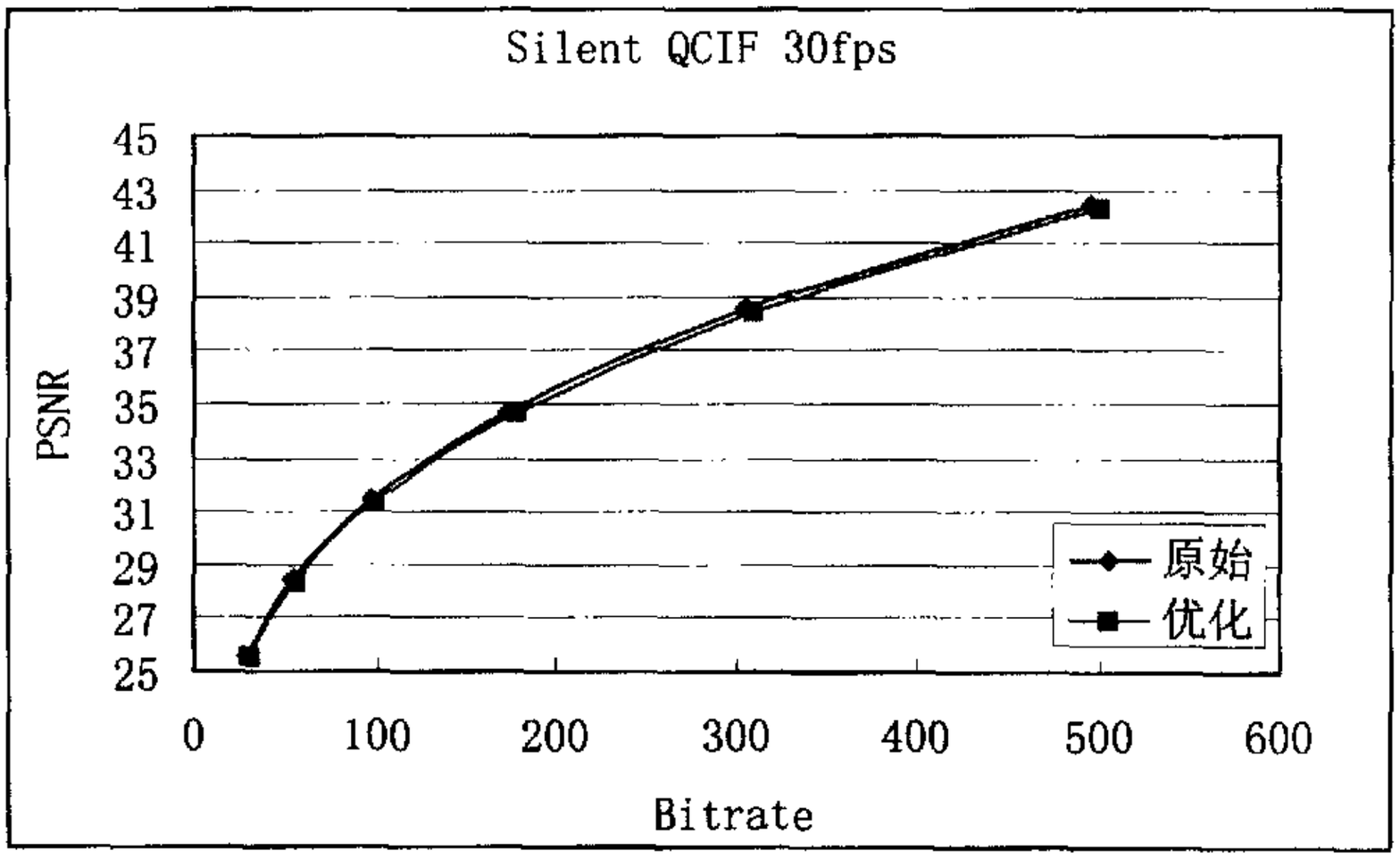

图 4.11 Silent_QCIF R—D 曲线

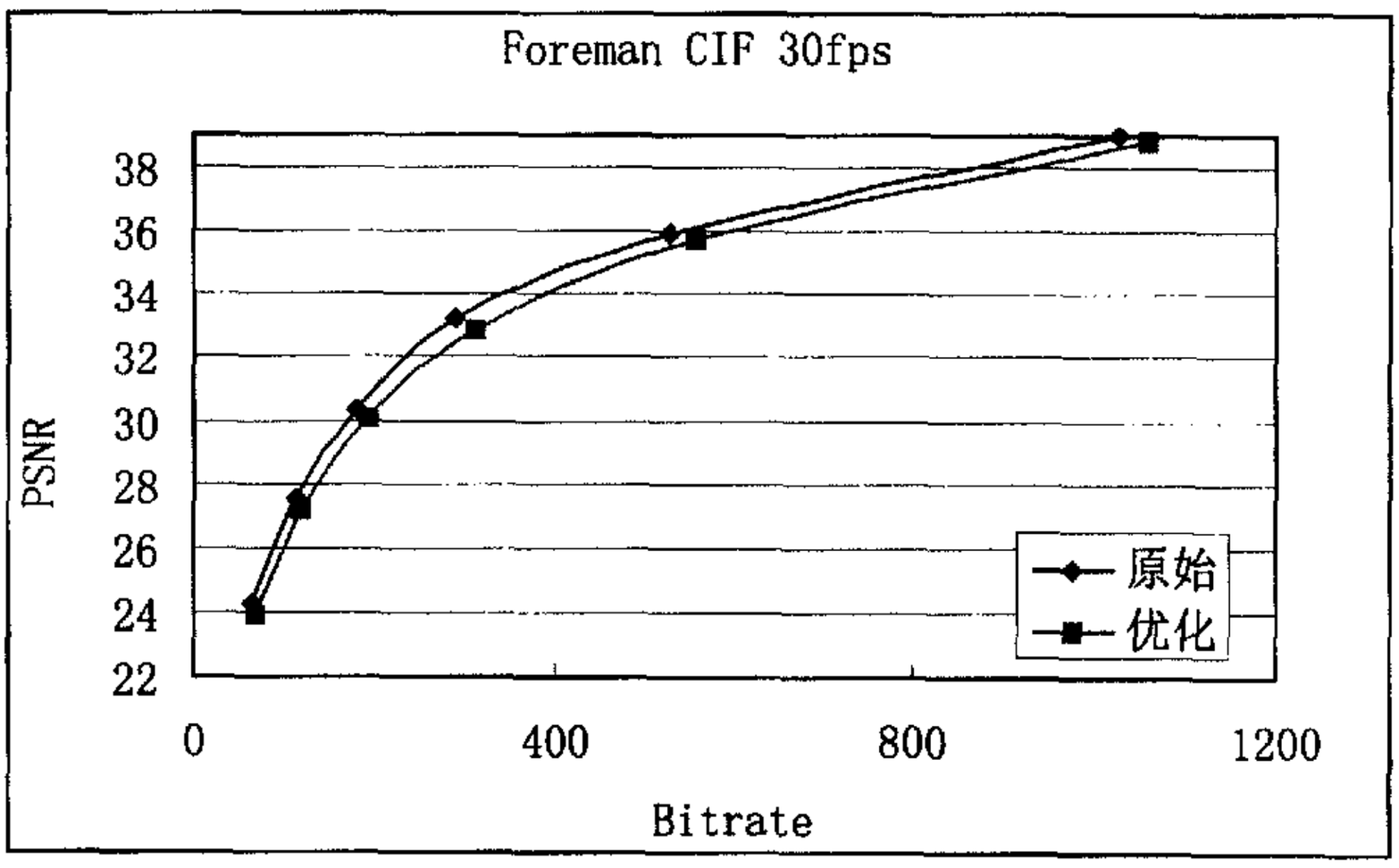

图 4.12 Foreman_CIF R—D 曲线

仿真实验结果显示，通过本文中所论述的方法可以在保持 PSNR 和比特率几乎不变的情况下有效的降低编码器的计算复杂性，提高编码效率。由于目前的优化措施主要集中在运动估计模块，因此对于较大的图像和较大的搜索半径效果最为明显。对 CCIR601 格式的 Cheer 序列，可以节约 90%以上的编码时间，对 CIF 格式可以减少 80%以上，而对 QCIF 序列效果相对最差，只能达到 70%。

PSNR 下降最大为 0.11dB，码率增加不超过 10%。码率增加主要是由于帧内编码块增加引起的，因此在运动复杂度最低的 Akiyo 序列中表现的最为突出。

4.7 小结

对于视频编解码系统来说，其性能的整体评价是由多个因素构成的。主要包括：编码复杂度、码率/带宽、失真/质量、延时、容错性能等。所谓“优化”实质是根据具体应用的需要，在这个多维因素中进行折中考虑。但在这其中如何降低编码器的复杂性始终是研究的热点问题。

由于 H.264/AVC 是一个新的视频编码标准，其标准的制定工作在今年才基本结束。因此本文选择了其中最为简单的 Baseline 编码器作为对象，针对 H.264/AVC 的特点，重点对帧内预测模式选择，运动搜索和帧间预测模式选择进行了优化。根据不同格式和运动复杂性的序列，可以节约 70%到 90%的编码时间，并且保持 PSNR 和比特率的变化在可以接受的范围内。

但同时还应该说明的是对于 H.264/AVC 编码器的优化这仅仅还是个开始，从实时编码的角度看还相差很远。目前根据本文算法优化的编码器在 Pentium IV 1.8G 计算机上对 QCIF 格式序列的最高编码速度在 19fps 帧左右。另一方面，不同快速算法之间的协同性还值得进一步考虑。例如在运动搜索优化后，基于全零块检测的运动估计快速算法的效果被明显削弱。而这一切有待于今后更进一步的研究加以解决。

第五章 结 论

5.1 全文总结

20 世纪 80 年代末 ITU-T 发布了第一个具有标志性的 H.261 视频编码国际标准，在其后的十几年间，随着计算机软硬件技术的飞速发展以及视频编码和处理算法研究的持续深入，ISO/IEC 和 ITU-T 又相继发布了 MPEG-1、MPEG-2、H.263 、MPEG-4 视频编码国际标准，并且都在各自的应用领域取得巨大的成功。在新的世纪，ITU-T 和 ISO/IEC 再一次携手，共同发展和制定新一代的视频编码国际标准，即 H.264/AVC。虽然 H.264/AVC 的编码框架同样是基于运动补偿和变换编码，但在实现细节中采用了众多新技术，使 H.264/AVC 的性能明显超过了现有的视频编码标准。在标准制定的过程中，H.264/AVC 还充分考虑了现有的和未来的各种传输网络，具有广泛的适用性。因此 H.264/AVC 吸引了学术界和工业界的广泛关注。

伴随着 H.264/AVC 高性能的是其难以忍受的巨大计算量，因此如何降低 H.264/AVC 编解码器的计算复杂性就成为 H.264/AVC 能否尽快进入实际商业应用领域，取得成功的关键因素之一。本文在充分研究 H.264/AVC 视频编码国际标准的文档，详细分析了参考软件编码器的基础上，研究工作主要取得了如下进展：

◆ 帧内预测模式选择快速算法

帧内预测可以充分利用图像中的空间相关性。为确保预测精度，标准提供了多种基于方向的预测模式，对 4x4 小块有 9 种预测模式，对 16x16 大块有 4 种预测模式，对色度 8x8 块有 4 种预测模式。全搜索所采用的遍历计算选择最优编码模式的策略是整个帧内预测的计算瓶颈。本文采用基于块边缘方向信息的模式选择策略，有效的减少了候选模式数，提高了帧内编码的效率。

◆ 运动搜索快速算法

运动估计模块是编码器中计算开销最高的模块。根据 H.264/AVC 中的多预测模式和多参考帧，本文在整像素运动搜索中采用了不对称十字和多六角形网格混合搜索快速算法，对于分数像素采用了基于中心的分数像素搜索快速算法。该算法可以在兼顾全局搜索的基础上，有效的减少搜索点数，并保持与全搜索相仿

的 PSNR 和比特率。仿真实验显示可以有效的提高编码器的编码效率。

◆ 大块模式选择快速算法

H.264/AVC 的一个重要的特点就是支持 7 种不同形状和尺寸的帧间预测模式。但统计结果显示在这 7 种块模式中，Mode1（16x16）作为最优模式的比例最高，平均在 70%以上。因此本文中对宏块的运动强度，水平方向宏块上半部和下半部的运动强度，垂直方向宏块左半部和右半部的运动强度进行了检测，并以此为判据进行大块模式的快速选择。

◆ 零块检测及其在运动搜索中的应用

全零块检测是低码率视频编码器常用的优化技术之一。本文根据整数变换的特点，给出 H.264/AVC 的全零块检测门限。在此基础上，利用全零块检测技术，明显提高了运动搜索和帧间预测模式选择的效率。

5.2 工作展望

作为一种新的视频压缩编码标准，H.264/AVC 表现出了非常卓越的编码压缩性能，同时也显示出巨大的计算开销和内存开销。因此如何根据 H.264/AVC 的特点，找到适合 H.264/AVC 的快速算法就成为当前研究的一个热点。作者计划下一步的研究重点为：

◆ CABAC 熵编码模块的研究和优化

基于内容的自适应二进制算术编码是H.264/AVC中可选的熵编码方式之一。CABAC 编码压缩的原理就决定了它的压缩效率要高于广泛使用的通用变长熵编码，但计算开销需增加 40%左右，目前还没有好的快速算法。

◆ 编码模式选择的快速算法

与以往标准中较少的编码模式选择不同，H.264/AVC 支持多种不同的帧内、帧间编码模式。因此编码器在对一个宏块（以 P 帧为例）编码时，必须在帧内预测各种模式、帧间预测各种模式中选择采用帧内还是帧间编码方式，以及是其中的那一种预测模式。如果使用 RDO 技术，计算量还将成倍增加。虽然本文采用的方法可以有效的减少模式选择的开销，但从实时的角度看还是远远不够的，还需要更有效的算法。

◆ 码率控制

由于在 JVT 的参考软件中支持 RDO 技术，使得在其中实现高效的码率控制比较困难。虽然码率控制不是标准的必要组成部分，但在视频编码的实际应用领域中，码率控制是一项非常重要和实用的技术。

参考文献

[1] A. M. Tekalp, "Digital Video Processing, " Englewood Cliffs: Printice Hall, 1995.
[2] K. R. Castleman, "Digital image processing," Englewood Cliffs: Printice Hall, 1996.
[3] 沈兰荪，卓力等，"视频编码与低速率传输，" 北京：电子工业出版社，2001.
[4] R. J. Clark, "Transform Coding of Images, " London: Academic Press, 1985.
[5] 章毓晋，图象工程(上册)——图象处理和分析，北京：清华大学出版社，1999
[6] N. Ahmed, T. Natarajan and K. R. Rao, "Discrete Cosine Transform", *IEEE Trans. Signal Computer*, vol. 23, 1974, pp.90-93.
[7] E. Feig,and S.Winograd, " Fast algorithms for the discrete cosine transform ," *IEEE Trans. Signal Processing*, vol. 40(9) , Sept. 1992, pp. 2174 –2193.
[8] B. Lee, "A New Algorithm to Compute the Discrete Cosine Transform," *IEEE Trans. Signal Processing*, vol. 32 (6), Dec. 1984, pp.1243-1245.
[9] J. W. Woods and S. D. O'Neil, "Subband coding of Images," *IEEE Trans. Acoust. Speech and Sign.* , Vol.34, pp.1278-1288, Oct. 1986
[10] M. Vetterli, "Multidimensional sub-band coding: Some theory and algorithms," *Signal Processing*, Vol. 6 , pp. 97-112, 1984.
[11] P. G. Howard and J. S. Vitter, " Arithmetic coding for datacompression," *Proceedings of the IEEE*, Vol 82(6), pp. 857–865, 1994.
[12] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for datacompression.," *Communications of the ACM*, Vol 30(6), pp.520–540, June 1987.
[13] J. Jiang, "Novel design of arithmetic coding for data compression, " *IEE Proc. Comput. Dig. Tech.*, Vol 142(6), pp.419–424, Nov, 1995.
[14] A. Moffat, R. M. Neal and I. H. Witten, "Arithmetic coding revisited," *ACM Trans. on Inf. Systems* Vol 16(3), pp. 256–294, July 1998.
[15] H. Li, M. Novak, and R. Forchheimer, "Fractal-based image sequence compression scheme," *Opt. Eng.*, vol. 32, no. 7, pp. 1588-1595, July 1993.
[16] M. S. Lazar and L. T. Bruton, "Fractal-based image sequence compression scheme," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 4, pp. 297-308, June 1994.
[17] J.D. Gibson, T. Berger, T. Looabaugh and et al., " Digital compression for multimedia principles & standards," Copyright©1997 by Morgan Kaufmann Publishers, Inc.
[18] T. S. Huang, S. C. Reddy, and K. Aizawa, "Human facial modeling, analysis, and synthesis for video compression," *Proc. SPIE Conf. Visual Communications and Image Processing*, Vol. 1605, pp. 234-241, Nov. 1991.
[19] D. Pearson, "Developments in model-based video coding," *Proc. IEEE*, vol. 83, pp. 892-906, June 1995.
[20] Yao Wang, Jorn Ostermann and Ya-Qin Zhang "Video Processing and Communications," Pearson Education: Printice Hall, 2002.

[21] 陈国斌, “视频编码中质量、复杂度和码率控制,” [博士毕业论文], 杭州: 浙江大学, 2003.
[22] Borko Furht, Joshua Greenberg, Raymond Westwater, “Motion Estimation Algorithms for Video Compression ,” Copyright © 2000, Kluwer Academic.
[23] J.Koga, K. Iiunuma, A. Hirani and et al., “Motion compensated interframe coding for video conferencing,” *in Porc. National Telecommunications Conference*, 1981, pp. G5.3.1-5.3.5
[24] R. Li, B. Zeng and M.L. Liou, “A new three-step search algorithm for block motion estimation,” *IEEE Trans Circuits Syst. Video Technol.*, vol. 4(4), Aug. 1994, pp. 438 –442
[25] J. Jain and A. Jain, “Displacement Measurement and its Application in interframe Image Coding,” *IEEE Trans. Communications*. vol. 29(12), Dec. 1981, pp. 1799 –1808
[26] R. Srinivasan, K. Rao, “Predictive Coding Based on Efficient Motion Estimation,” *IEEE Trans. Communications*, vol.33(8), Aug 1985, pp. 888 -896
[27] B. Furth, J. Greeberg, R. Westwater, “Motion estimation algorithms for video compression,” Copyright©1997 by Kluwer Academic Publishers
[28] ISO/IEC JTC1/SC29/WG11, N3324, “Optimization Model: Version 1.0,” Noordwijkerout, 2000.
[29] ISO/IEC JTC1/SC29/WG11, N3675, “Optimization Model: Version 2.0,” La Baule, Oct 2000.
[30] A.M. Tourapis and M.L. Liou, “New Results on Zonal Based Motion Estimation Algorithms - Advanced Predictive Diamond Zonal Search,” *IEEE Int. Symp. Circuit Syst. (ISCAS 2001)*, vol. 5, 2001, pp. 183–186.
[31] ITU-T Recommendation H.261, “Video codec for audiovisual services at $p \times 64$ kbit/s,” in Proc. *COM 15R 16-E,* March, 1993
[32] ISO/IEC CD 11172, “Coding of moving pictures and associated audio for digital storage media at up to 1.5Mbits/sec — Part 2: Coding of moving pictures information,” Dec. 1991
[33] ISO/IEC 13818-2, “Information technology — Generic coding of moving pictures and associated audio Part 2: Video,” 1995
[34] ITU-T Recommendation H.263, “Video coding for low bitrate communication,” May, 1996
[35] B. Girod, U.Horn and b. Belzer, “Scalable video with multiscale motion compensation and unequal error protection,” *Multimedia Communications and Video Coding*. New York: Plenum Press, 1996
[36] ITU-T Recommendation H.263 Version 2, “Video coding for low bitrate communication,” Jan. 1998.
[37] ITU-T Draft for “H.263++” Annexes U, V, and W to Recommendation H.263. Nov. 2000.
[38] ISO/IEC FDIS 14496-2, “Information technology — Generic coding of audio-visual objects Part 2: Visual,” Oct. 1998

[39] 钟玉琢，王琪，贺玉文. 基于对象的多媒体数据压缩编码国际标准——MPEG-4 及其校验模型. 北京：科学出版社. 2000 年 10 月
[40] ITU-T TSG16. Draft call for proposal for H.26L Video coding[R]. ITU-T VCEG 16,1998
[41] ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, Document JVT-G050, 7th Meeting: Pattaya, Thailand, March, 2003
[42] ThomasWiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 13(7), pp. 657-673, July 2003.
[43] Ralf Schäfer, Thomas Wiegand, Heiko Schwarz, "The emerging H.264/AVC Standard" EBU Technical Review, Jan. 2003
[44] Ye-Kui Wang, Miska M. Hannuksela, Viktor Varsa, "The Error Concealment Feature in the H.26L Test Model" Proc. IEEE International Conference on Image Processing (ICIP'02), Vol. II, pp. 729-732, Sept. 2002.
[45] S.Wenger, T. Stockhammer, "An Overview on the H.264 NAL Concept" Doc. JVT-B028, Geneva, Switzerland, January 2002
[46] VideoLocus Inc. "AVC Real-Time SD Encoder Demo" Doc. JVT-D023, Klagenfurt, Austria, July 2002
[47] Anthony Joch, "Demostration of Main Profile Decoder on TI C64x", Doc. JVT-F075, Awaji, JP, Dec, 2002
[48] Gary Sullivan, "JVT IPR Status Report" Doc. JVT-C110, Fairfax, Virginia, USA, May, 2002
[49] Thomas Stockhammer, Miska M. Hannuksela, and Thomas Wiegand, "H.264/AVC in Wireless Environments," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 13(7), pp. 657-673, July 2003.
[50] Stephan Wenger, "H.264/AVC Over IP," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 13(7), pp. 645-656, July 2003.
[51] Tu-Chih Wang, Hung-Chi Fang, and Liang-Gee Chen, "Low-Delay and Error-Robust Wireless Video Transmission for Video Communications," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 12(12), pp. 1049-1058, Dec. 2002.
[52] Yan Lu, Wen Gao, Feng Wu. "Sprite generation for frame-based video coding". *In Proc. 2002, IEEE Int. Conf. Image Processing*, vol. 1, pp. 437 -476.Sep. 2001,
[53] Yan Lu, Wen Gao, Feng Wu, "High Efficient Sprite Coding With Directional Spatial Prediction," *In Proc. 2002, IEEE Int. Conf. Image Processing*, vol. 1, pp. 201 -204.Sep. 2002,
[54] VideoLocus Inc. "H.264/MPEG-4 AVC Compression Tutorial." http://www.videolocus.com/
[55] Henrique S Malvar, Antti Hallapuro, Marta Karczewicz, Louis Kerofsky. "Low-Complexity Transform and Quantization in H.264/ AVC" *IEEE Trans. Circuits Syst. Video Technol.* Vol. 13(7), pp. 598-603, July 2003.
[56] Detlev Marpe, Heiko Schwarz, and Thomas Wiegand. "Context-based adaptive binary arithmetic coding in the H.264 AVC video compression standard.pdf" *IEEE*

Trans. Circuits Syst. Video Technol. Vol. 13(7), pp. 598-603, July 2003.
[57] M.-C. Hong and H.-S. Hahn, "A Loop/Post Filter to Suppress Blocking and Ringing Artifacts for H.26L VideoCodec" *In Proc. 2002, IEEE Int. Conf. Image Processing*, vol. 1, pp. 940 –947, Jan. 2001.
[58] JVT Reference Software Version 61e http://bs.hhi.de/~suehring/tml/download
[59] Sergio Saponara, Carolina Blanch, Kristof Denolf, "Data Transfer and Storage Complexity Analysis of the AVC/JVT Codec on a Tool-by-Tool Basis" Doc. JVT-d138, Klagenfurt, Austria, July, 2002.
[60] Christophe Clerc, Massimo Ravasi, Marco Mattavelli, "Complexity Evaluation of Different Configurations of JVT Codec" Doc. JVT-G038, Pattaya, Thailand, March, 2003.
[61] Kristof Denolf, Carolina Blanch, "Initial Memory Complexity Analysis of the AVC Codec," Fairfax, Virginia, USA, May, 2002.
[62] Thomas Wiegand, Bernd Girod, "Lagrange Multiplier Selection in Hybrid Video Coder Control,"*In Proc. IEEE International Conference on Image Processing*, Thessaloniki, Greece, Sep. 2001.
[63] Anthony Joch, Faouzi Kossentini, Heiko Schwarz, Thomas Wiegand ,Gary J. Sullivan "Performance Comparison of Video Coding Standards using Lagrangian Coder Control" *In Proc. IEEE International Conference on Image Processing (ICIP 2002)*, NY, USA, Sep. 2002.
[64] JN Zhang, YW He, SQ Yang, YZ Zhong, "Performance and Complexity Joint Optimization for H.264 Video Coding", *In ISCAS'2003*, May 25-28, 2003,
[65] Hye-Yeon Cheong, Alexis Tourapis, "Fast Motion Estimation within the H.264 Codec", *In Proc. IEEE International Conference on Multimedia & Expo (ICME 2003)*, Baltimore, USA, July. 2003
[66] Pen Yin, Hye – Yeon Cheong Tourapis, Alexis Michael Tourapis and Jill Boyce "Fast Mode Decision And Motion Estimation For JVT/H.264", *In Proc. IEEE International Conference on Image Processing (ICIP 2003)*, Barcelona ,spain, Sep. 2003
[67] Patrick Ndjiki-Nya, Bela Makai, Aljoscha Smolić, Heiko Schwarz, and Thomas Wiegand, "Improved H.264/AVC Coding Using Texture Analysis and Synthesis", *In Proc. IEEE International Conference on Image Processing (ICIP 2003)*, Barcelona, Spain, September 14-17, 2003
[68] Aljoscha Smolić, Yuriy Vatis, Heiko Schwarz, and Thomas Wiegand, "Improved H.264/AVC Coding Using Long-Term Global Motion Compensation", *In Proc. VCIP 2004, SPIE Visual Communications & Image Processing*, San Jose, CA, USA, January 2004.
[69] Bojun Meng, Oscar C, Chi-Wah Wong, Hong-Kwai Lam, "Efficient Intra-Prediction Mode Selection for 4x4 Blocks in H.264", *In Proc. IEEE International Conference on Multimedia & Expo (ICME 2003)*, Baltimore, USA, July. 2003
[70] Gary Sullivan, "JVT IPR Status Report" Doc. JVT-C110, Fairfax, Virginia, USA,

May, 2002.
[71] Tom McMahon, Thomas Wiegand, Gary Sullivan, "Draft Prof. Ext Amendment," Doc. JVT-I047, San Diego, CA, USA, Sep., 2003.
[72] Richard Gerber, "The Software Optimization Cookbook," Copyright @ Intel Corporation. 2002.
[73] Ville Lappalainen1 and Timo D. H, "Unified Method for Optimization of Several Video Coding Algorithms on General-Purpose Processors,"*In Proc. 2002, IEEE Int. Conf.* Information Technology, vol. 1, pp. 431 –439, Jan. 2002.
[74] Bojun Meng, Oscar C. Au, Chi-Wah Wong, "Efficient Intra-Prediction Algorithm in H.264" *In Proc. IEEE International Conference on Image Processing (ICIP 2003)*, Barcelona ,spain, Sep. 2003
[75] Feng PAN, Xiao LIN, Rahardja, "Fast Mode Decision for Intra Prediction" Doc. JVT-G013, Pattaya, Thailand, March, 2003.
[76] Zhibo Chen, Peng Zhou, Yun He, "Fast Integer Pel and Fractional Pel Motion Estimation for JVT," Doc. JVT-F017, Awaji, JP, Dec. 2002
[77] Cheng Du, Yun He , Junli Zheng, "PPHPS A Parabolic Prediction-Based, Fast, Half Pixel Search Algorithm for Very Low Bit-rate Moving Picture Coding", *IEEE Transaction on CSVT*, VOL. 13, NO. 6, pp.514-518, June, 2003
[78] Yu-Wen Huang, Bing-Yu Hsieh, Tu-Chin Wang, "Analysis and Reduction of Reference Frames for Motion Estimation in MPEG-4 AVC/JVT/H.264," *In Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP 2003)*, HK, April. 2003
[79] Andy Chang, Oscar C.Au, Y. M. Yeung, "A Novel Approach to Fast Multi-Block Motion Estimation for H.264 Video Coding," *In Proc. IEEE International Conference on Multimedia & Expo (ICME 2003)*, Baltimore, USA, July. 2003
[80] Yu Alice, Lee R, Flynn M. Early detection of all-zero coefficients in H.263. PCS'97, Berlin, Germany, 1997
[81] 周璇，谭径微，余松煜. "H.263 中预先判别全零系数的新方法" *上海交通大学学报*，vol. 32(9), pp. 107-109 Sep. 1998，
[82] 王熹微，栗强，崔慧娟，唐昆，"最佳零块判决准则下的运动搜索算法，" *清华大学学报（自然科学版）*., vol. 43(7), pp. 938-941 2003.

致　谢

两年半的学习即将结束，辍笔在即，思绪如潮。离开大学校园十年后能有机会再次进入大学深造，这是我人生中宝贵并且意义深刻的一段经历。在此期间，得到众多老师、领导和朋友们的帮助、支持。在此深表谢意！

首先，要真诚感谢导师刘济林教授。感谢他给了我十年后重入校园深造的机会，感谢他为我精心安排培养计划，感谢他给予的热情指导与无私关怀。刘老师渊博的学识、严谨的治学、求实创新的科研态度都是我将来教学和科研中学习的楷模。希望不久的将来能有机会再次亲聆刘老师教诲。感激之情难以言表，再次对刘济林教授的关怀、指导、帮助致以崇高的敬意和衷心的感谢！

感谢王兴国副教授在我科研工作中给予的大量细致入微的指导和帮助，他扎实的专业功底、开阔的视野、敏锐的科研洞察力都是我学习的榜样。感谢何杞鑫副教授和陈邦媛副教授，正是他们的鼓励才使我鼓起继续求学深造的勇气。感谢叶秀清教授、李宏东副教授，他们严谨的科学态度给我留下了深刻的印象。此外还要感谢浙江大学王大力老师和包立伟老师给予的帮助。感谢 311 实验室的众位伙伴给予的无私帮助和指导，他们是陈国斌、薛全、王正友、陈建乐、张颖、孙小叶、姜超、张文俊、叶建宏、谢亚光、王校等，和他们在一起的日子是如此的愉快和难以忘怀。

在我读硕期间，还得到了我的工作单位：国际关系学院杭州校区各位领导和同事的大力支持和帮助。学校的四位校长都亲自过问我的学习和科研情况，部门领导在安排我的校内工作时也给我的学习提供了许多便利，同事们也给予我极大的帮助。在这里向他们表示由衷的感谢。

最后我还要特别感谢我的妻子、儿子以及我的家人。正是因为她们的默默支持、鼓励和始终不渝的陪伴才使我能够全身心的投入到学习和科研中去，使我有信心去面对困难，迎接挑战。我的一点一滴的进步都与她们的默默奉献分不开。

王 嵩

2004 年 2 月

H.264/AVC视频编码标准研究及其编码器的优化

作者：王嵩
学位授予单位：浙江大学
被引用次数：12次

引证文献(12条)

1. 陈亚菲 基于视频编码标准的去块效应算法研究[学位论文]硕士 2007

2. 袁夫全.韩军 H.264/AVC算术编码的优化实现[期刊论文]-中国有线电视 2006(23)

3. 王卓凌 H.264视频编码标准中模式选择和快速搜索算法研究[学位论文]硕士 2006

4. 汪燮彬 多媒体处理库（MML）在BF53x上的优化研究[学位论文]硕士 2006

5. 戴虎 基于H.264/AVC的码率控制算法的研究[学位论文]硕士 2006

6. 梁铁 DCT域HDTV至SDTV视频转码器关键算法的研究与实现[学位论文]硕士 2006

7. 曹宇辉 H.264运动估计整像素部分算法的改进与实现[学位论文]硕士 2006

8. 何凌 基于TMS320C6416的H.264视频编码算法研究与实现[学位论文]硕士 2005

9. 袁春 H.264/AVC中多参考帧下的失真度估算算法[学位论文]硕士 2005

10. 曾勇 基于H.264/AVC的率失真优化和码率控制算法研究[学位论文]硕士 2005

11. 石迎波 MPEG-4视频编码系统的研究与实现[学位论文]硕士 2005

12. 张应晨 基于H.264图像压缩系统的设计及调制方式的研究[学位论文]硕士 2005

本文链接：http://d.g.wanfangdata.com.cn/Thesis_Y581503.aspx