符号说明

$\exp(\cdot)$	指数运算
$\mathrm{H}(x)$	x的熵
$\mathrm{H}(x,y)$	x, y的联合熵
$\mathrm{I}(x,y)$	x, y的互信息
$\mathrm{K}(x, y)$	核函数
$\ln(\mathrm{a})$	以e为底的a的对数
$\max(\cdot)$	最大值
$\mathrm{median}(\cdot)$	中间值
$\min(\cdot)$	最小值
$p(\cdot)$	概率分布
$\mathrm{sign}(\cdot)$	符号函数
λ	拉格朗日算子
$\|\cdot\|$	绝对值
Σ	求和

缩略语

ADVS	Activity Dominant Vector Selection
AGS	Adaptive GOP Selection
CABAC	Context-Adaptive Binary Arithmetic Coding
CAVLC	Context-Adaptive Variable Length Coding
CIF	Common Intermediate Format
CPDT	Cascaded Pixel-Domain Transcoding
DCT	Discrete Cosine Transform
DCT-MC	The Discrete Cosine Transform domain Motion Compensation
DOH	Difference of Histogram
FDVS	Forward Dominant Vector Selection
FREXT	Fidelity Range Extensions
GOP	Group of Pictures
JVT	Joint Video Team
HOD	Histogram of Frame Difference
HVSBM	Hierarchical Variable Size Block Matching
IDCT	Inverse Discrete Cosine Transform
IEC	International Electrotechnical Commission
IQ	Inverse Quantization
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union Telecommunication Standardization Sector
MAD	Mean of Absolute Difference
MCTF	Motion Compensated Temporal Filtering
MC-EZBC	Motion-compensated Embedded Zeroblocks Coder
MC	Motion Compensation
ME	Motion Estimation
MI	Mutual Information

MPEG	Motion Picture Experts Group
nz_per	percentage of non-zero coefficients
PSNR	Power Signal-to-Noise Ratio
QCIF	Quarter Common Intermediate Format
QMF's	Quadrature Mirror Filters
Q_i	incoming quantization parameter
QP	Quantization Parameter
Q_r	re-quantization parameter
SAD	Sum of Absolute Difference
SVC	Scalable Video Coding
SVM	Support vector machine
UMCTF	Unconstrained Motion Compensated Temporal Filtering
VCEG	Video Coding Experts Group
VLC	Variable Length Coding
VT	Video Transcoding

原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的科研成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律责任由本人承担。

论文作者签名：　　　日　期：2008.5.28

关于学位论文使用授权的声明

本人完全了解山东大学有关保留、使用学位论文的规定，同意学校保留或向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅；本人授权山东大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其他复制手段保存论文和汇编本学位论文。

(保密论文在解密后应遵守此规定)

论文作者签名：　　　导师签名：　　　日　期：2008.5.28

摘 要

视频的应用环境非常复杂，从传输的信道，存储介质，到播放终端等都各不相同。视频自适应技术为视频的复杂应用提供了各种解决方案，其中就包含视频转码技术和可分级编码技术。如在网络接入点放置转码模块，就可以根据接入网络的特点来生成所需要的视频流格式；另外利用可分级编码技术，则只需要在编码端一次性编码高分辨率的比特流，不同的网络和终端根据本身的特点只接受部分码流解码即可。

视频转码通常分为标准内转码和标准间转码两种，标准内转码又常分为空间分辨率转码，时间分辨率转码，比特率转码三个方面。视频转码最容易实现的方式是将输入比特流完全解码，然后根据输出格式的要求进行重新编码，显然该方法也是运算复杂度最高的方法。为了提高重新编码速度，在视频转码中就需要充分利用解码得到的信息。

在视频可分级编码中，编码端只需要一次性编码全分辨率下的比特流，不同应用的解码端只需要根据特定的应用环境接受部分码流进行解码即可，因此就减轻了编码端的负担。基于MCTF（Motion Compensated Temporal Filtering ）的小波视频编码方案中完全抛弃了迭代编码方式，因此可以避免“漂移”效应。但是，在基于MCTF的小波编码方案中，GOP（Group of Pictures）结构是固定的，因此无法适应视频序列中运动性质的变化。

针对视频转码技术和可分级编码技术，本文主要研究了以下几个方面：

1. 在空间分辨率转码的帧内模式选择部分，本文利用原始图像中非零系数比例（*nz_per*）作为选择当前宏块类型的准则，并提出了一个$Th_I_Q_r$模型，该模型以指数曲线描述重新量化参数（Q_r）和*nz_per*阈值的关系。经过线性化处理，得到一元线性回归模型，然后利用最小二乘法估计模型中的参数。为了使得$Th_I_Q_r$模型能适应不同的视频序列，本文提出了一种在实际转码过程中更新模型参数的方法。在使用$Th_I_Q_r$模型选择了帧内宏块类型之后，本文又提出了一种快速的帧内预测模式选择方法，该方法充分利用输入原始图像中宏块的类型和帧内预测模式，大幅度降

低了当前宏块的帧内预测模式选择时间。根据最后的实验结果，相对于全搜索法，在最大PSNR（Power Signal-to-Noise Ratio）损失约0.6dB前提下，本文方法的耗时仅为全搜索法的20%~25%。

2. 在空间分辨率转码的帧间模式选择部分，本文利用*nz_per*划分出当前宏块所在区域的运动性质，从而跳过部分候选宏块类型的测试，并提出了一种$Th_P_Q_r$模型。与$Th_I_Q_r$模型类似，该模型同样使用指数曲线来描述Q_r和*nz_per*阈值的关系，并在实际转码过程中进行即时更新。另外，由于根据原始图像计算出来的运动矢量并非一定精确，尤其是当Q_r较大时。本文还提出了一种新的运动矢量细化方案，该方案中以*nz_per*作为运动矢量细化步长的准则，且随着Q_r的增加，运动矢量细化步长也逐步增加，从而保证了在运动较为剧烈的区域，运动矢量细化步长较长。本文又进一步将该方法推广到了时间分辨率转码方面。最后的实验结果表明，相比于全搜索法，在最大PSNR损失约1.1dB前提下，本文方法可以将总编码速度提高15-20倍；若仅考虑选择宏块类型部分的耗时，则可以提高约35倍。

3. 本文首次提出基于分类方法在视频转码中快速选择宏块类型。利用该方法，本文首次完成了基于H.264的同时包含三个方面（空间、时间、质量）的转码方案。从输入比特流中提取解码信息：原始图像中宏块类型、残差数据、运动矢量、量化参数等，并将这些信息输入到离线训练完毕的支持向量机模型，从而预测出目标宏块类型。本文在各种转码条件下进行了大量的实验，相比于全搜索法，在最大PSNR损失约1.2dB前提下，本文方法可以将总编码速度提高约12倍，若仅考虑选择宏块类型部分，则可以提高约30倍。

4. 本文提出了一种类haar的MCTF编码方案，该方案包含GOP结构选择和时间分解层次确定两部分。其中GOP结构根据互信息自适应的确定，又包含了GOP尺寸选择和低通帧选择两部分。本文同时利用GOP内平均互信息值和标准差来控制GOP尺寸，从而选定的GOP尺寸不仅能根据运动类型的变化自适应的改变，而且同一个GOP内部的运动类型也能保持一致。本文首次提出了一种低通帧的选择方案，该方案基于互信息技术，从一个GOP内提取出与其余帧最具相关性的帧。当解码端在时间上的解码层

次较少时，该方案得到的帧序列更能反映出原始视频序列的运动过程，另外该方案还进一步提高了压缩性能。进一步地，本文根据选择的GOP结构，自动确定时间上的分解过程，该分解过程还与传统的MCTF编码方案保持了兼容性。根据最后的实验结果，对于运动性质有明显变化或运动较为剧烈的序列，本文的GOP结构选择方法能较大地提高压缩性能。

综上所述，在视频转码的研究中，本文首次提出了一种基于H.264的同时包含空间、时间、质量三个方面的转码方案，论文尤其对输出比特流中的宏块类型选择问题进行了深入的研究。本文提出的方案中，输入和输出比特流均为H.264格式，输入的H.264比特流需要完全解码（像素域转码），在更改图像格式之后重新编码输出，其中图像格式的更改包含三个方面：空间分辨率，时间分辨率，图像质量。在可分级编码研究中，本文基于互信息技术提出了一种自适应的GOP结构选择方案，并根据选定的GOP结构进一步的确定了时间分解过程。最后，论文对提出的视频转码方案及GOP结构选择方案中存在的问题进行了分析，并讨论了下一步的研究方向和研究内容。

关键词：H.264，视频转码，宏块类型选择，线性回归，互信息，Motion Compensated Temporal Filtering，帧组尺寸.

ABSTRACT

Video sequences are often used in different application environments, ranging from transmitting channels, storage media and display terminals. Video adaptation provides different technical schemes including video transcoding and scalable video coding, which all provide the responding resolutions. For instances, video transcoding module can be adopted in network access point, and the required video format can be transcoded directly. On the other hand, in scalable video coding, source video is encoded once, and decoder can receive partial bitstream according to its special application.

Video transcoding can be classified as homogeneous transcoding and heterogeneous transcoding according to incoming bitstream standard and outgoing bitstream standard. In homogeneous transcoding, there are main three aspects: spatial resolution transcoding, temporal resolution transcoding, and bit rate transcoding. The easiest implement of video transcoding is cascaded pixel domain transcoding and it is also the most computational complexity scheme. To speed up the re-encoder process, the decoded information from incoming bitstream should be utilized in video transcoding.

In scalable video coding, source video is encoded at the highest resolution, and the decoder can receive partial bitstream depending on specific rate resolution required by a certain application which can release the burden of encoder. The popular hybrid motion compensated prediction and block transform scheme will cause the "drift" effect when the decoder receives the in-complete bitstream because of its recursive structure. The wavelet encoding scheme based on motion compensated temporal filtering, which entirely abandons recursive structure, can provide high flexibility in bitstream scalability for different spatial, temporal and quality resolutions. However, in conventional motion compensated temporal filtering encoding scheme, the group of picture structure is fixed which don't consider the variation of motion activities in real video sequences.

In video transcoding and scalable video coding, the main contributions in this thesis including:

1. In intra prediction mode selection of spatial resolution transcoding, the

percentage of non-zero coefficients (*nz_per*) in pre-coded frame is utilized as criterion to select macroblock mode in downsized frame. A *Th_I_Q*$_r$ model describing the relationship between re-quantization parameter and threshold of *nz_per* which implemented by an exponent curve is proposed in this part. This model is converted into a linear regression model, and least square method is adopted to estimate parameters in the model. To meet up with the requirement of specific video sequence, an update process of parameters in the model is proposed in this thesis, which utilizing selected macroblock modes in re-encoder process. After the selection of intra macroblock mode, a fast intra prediction mode selection is proposed in the thesis, which utilizing incoming macroblock modes and prediction modes in pre-coded frame, and computational complexity can be reduced greatly by the proposal. According to the experimental results, on the pre-condition that the maximum PSNR loss is about 0.6dB, the computational complexity can be saved is about 20%~25% by the proposal comparing to full search algorithm.

2. In the inter mode selection part of spatial resolution transcoding, *nz_per* is utilized to classify the motion activity of current macroblock, and some candidate macroblock modes are skipped according to the classified result. A *Th_P_Q*$_r$ model is proposed to descript the relationship between *nz_per* threshold and re-quantization parameter in re-encoder process. As similar to *Th_I_Q*$_r$ model, an exponent curve is adopted to descript the relationship between *nz_per* threshold and re-quantization parameter, and an update process of parameters in model is also proposed in the thesis. The initial motion vectors of macroblock are calculated according to pre-coded frame, and they are not very precision, especially in the situation of re-quantization parameter is large. A new motion vector refinement method is proposed which adopts *nz_per* as criterion to calculate the refinement steps. In the proposal, with the increase of re-quantization parameter, the refinement steps increase as well. In the area with high *nz_per* value responding to high motion activity, a longer refinement steps is used. The *Th_P_Q*$_r$ model is also extended into temporal resolution transcoding in this thesis. According the experimental results, the proposed method achieves about 15-20 times improvement in the re-encode computational complexity comparing to full search algorithm, while the maximum PSNR is degraded by 1.1 dB, and about 35 times can be

improved by the proposal in macroblock mode selection part.

3. The classification method is introduced firstly into macroblock mode selection in this thesis. A fast mode decision scheme is proposed based on support vector machine. The features vectors used in training and classification stage of support vectors machine are distilled from incoming bitstream, including motion vectors, residual data, pre-coded macroblock modes, and quantization parameters etc. A H.264 video transcoding including spatial resolution transcoding, temporal resolution transcoding, and bit rate transcoding simultaneously is implemented based on classification method for the first time. The extensive experiments are performed including intra mode selection and inter mode selection. In intra mode selection part, the proposed method achieves about 15 times improvement in the computational complexity comparing to full search algorithm, while the maximum PSNR is degraded by 0.3dB; On the other hand, about 25-30 times can be speed up in inter mode selection, while the PSNR is degraded by 0.2-1.2dB depending on different sequences and bit rate.
4. In MCTF encoding scheme, we propose an adaptive group of picture structure selection scheme, in which the group of picture size and low-pass frame position are selected based on mutual information. Furthermore, the temporal decomposition process is determined adaptively according to the selected group of picture structure. A large amount of experimental work is carried out to compare the compression performance of proposed method with the conventional motion compensated temporal filtering encoding scheme and adaptive group of picture structure in standard scalable video coding model. The proposed low-pass frame selection can improve the compression quality by about 0.3-0.5db comparing to the conventional scheme in video sequences with high motion activities. In the scenes with un-even variation of motion activities, e.g. frequent shot cuts, the proposed adaptive group of picture size can achieve a better compression capability than conventional scheme. When comparing to adaptive group of picture in standard scalable video coding model, the proposed group of picture structure scheme can lead to about 0.2~0.8 dB improvements in sequences with high motion activities or shot cut, especially abrupt shot cut.

From the above mentioned, in the research of video transcoding, a H.264 video

transcoding including spatial resolution transcoding, temporal resolution transcoding, and bit rate transcoding simultaneously is implemented in the thesis for the first time. In the proposal, the incoming bitstream with H.264 format should be full decoded. After the change of image format (spatial resolution, temporal resolution, image quality), the outgoing bitstream is re-encoded with H.264 also. In scalable video coding part, an adaptive group of picture structure selection is proposed based in mutual information. In the final part, the opening issues are discussed, and the future directions are analyzed as well.

Keywords: H.264, Video Transcoding, Macroblock Mode Selection, Linear Regression, Mutual Information, Motion Compensated Temporal Filtering, Group of Picture.

第1章 绪论

视频的应用环境非常复杂，从传输的信道，存储介质，到播放终端等都各不相同。例如在互联网上，接入的网络有局域网、无线网、拨号上网等，各种网络也就具有不同的带宽等信道性质；具体到用户终端，有PC，PDA，手机，机顶盒等，这些不同的终端从CPU速度，内存大小，存储空间，显示能力等均有很大差异。为了能适应这些不同的信道和终端，视频信号就需要动态的调整，视频自适应技术（Video Adaptation）[1]为视频的复杂应用提供了各种解决方案，它根据用户端的功能和环境要求，对原始视频流转换处理，生成一个新格式的视频流从而满足用户的需求。视频自适应技术作为一个新兴的领域，目前还没有完整的明确定义[1]，它实际是上利用各种已有的视频处理技术来适应不用的应用环境，这些技术大致包括：视频转码技术（Video Transcoding），可分级编码技术（Scalable Video Coding），关键帧提取（Key Frame Selection），镜头分割（Shot Detection），视频描述（Video Description）等。本论文研究的范围包括视频转码技术和可分级编码技术，如在网络接入点，就可以利用视频转码技术生成接入网络所需要的比特流格式，从而适应了不同的网络；利用可分级编码技术，则只需要在编码端一次性编码高分辨率的比特流，不同的网络和终端根据本身的特点只接受部分码流解码即可。二者的不同之处在于，若利用视频转码技术，则在网络接入点（路由器，代理服务器等）需要额外的开销（配置转码模块），但可以生成不同标准格式的比特流，如H.263到H.264；可分级编码技术不需要额外的开销就可以完成视频流的转换，但是该技术无法转换视频流的标准格式。因此两者具有各自的优缺点，需要根据具体应用而定。

1.1 视频转码技术及国内外研究现状

视频转码的基本过程见图 1-1。视频转码的输入是一种比特流格式（空间分辨率 S1，时间分辨率 T1，码率 R1，标准 C1 等），经过转码模块，可以得到另一个比特流格式（空间分辨率 S2，时间分辨率 T2，码率 R2，标准 C2 等）。从不同的角度分类，视频转码可以划分为不同的类别。根据输入和输出比特流格式，视频转码通常分为标准间转码和标准内转码两种。标准间转码指输入比特流和输

出比特流属于不同的标准格式，就如图 1-1 中输入标准为 C1，输出标准为 C2。标准内转码指输入和输出比特流属于同一类标准，转码的目的主要是降低输出码率，从而适应不同的带宽，又常分为空间分辨率转码[2]-[11]，时间分辨率转码[12]-[13]，比特率转码[27]-[31]三个方面。

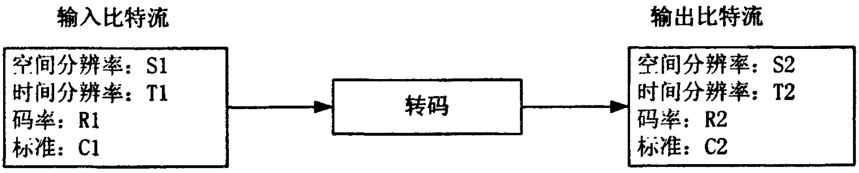


图 1-1 视频转码概述

空间分辨率转码是通过改变视频图像的尺寸达到降低比特流的目的，图像缩放因子分为整数比例[3]-[6]和任意比例[7]-[9]两种。在空间分辨率转码中，一个常见的问题是运动矢量合并。假设图像缩放因子为 2 的情况，当前帧（降低尺寸后的图像）的一个宏块（16x16 像素块）对应着原始图像（转码系统的输入帧，即高分辨率下的数据）的四个宏块，问题在于如何将四个宏块的运动矢量合并到一个宏块中，如下图所示。

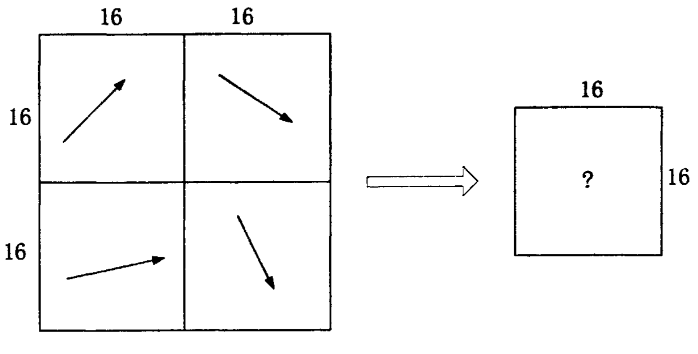


图 1-2 空间分辨率转码中运动矢量的计算

常用的方法有任选其一[1]，选择最大直流系数残差[19]，选择最多方向的运动矢量[14]，平均值[1]，中间值[1], [14]等等。根据文献[19]，中间值方法的效果较好；在任意比例因子情况下，当前帧中一个宏块对应到原始图像的区域通常会叠加到几个宏块上，而且宏块边界不会完全对齐，常用的运动矢量合并方法则有

加权平均[6]、[7]、[10]，加权中间值[8]、[10]等，常见的权重为运动矢量所占的图像面积，对应块的残差能量等。

文献[66]还提出了一种在任意比例因子下转换域的图像缩放方法。缩小图像尺寸通过截断部分高频离散余弦变换（Discrete Cosine Transform, DCT）系数完成，放大图像则通过为 DCT 系数高频分量补零完成。截断和补零操作最后通过一个快速的正反 DCT 变换来实现。

在基于多种块类型（如 H.264）的转码中，另外一个研究方向是在降低图像尺寸后如何选择当前宏块类型[3]-[5]，[16]-[18]。事实上，宏块类型选择与运动矢量合并是密切相关的，文献[5]就利用原始图像中的运动矢量和残差数据的特征，在转码方案中同时考虑了降低尺寸后图像的宏块类型和运动矢量。

时间分辨率转码中通过改变帧率（丢弃部分帧）达到降低比特流的目的。在时间分辨率转码中，由于部分帧需要被舍弃，如果某帧以丢弃帧作为参考帧，则它的运动矢量将不复存在。如图 1-3，F_n 为当前帧，F_{n-1} 被丢弃，则 F_n 中指向 F_{n-1} 中的运动矢量就要重新计算，并指向 F_{n-2}。

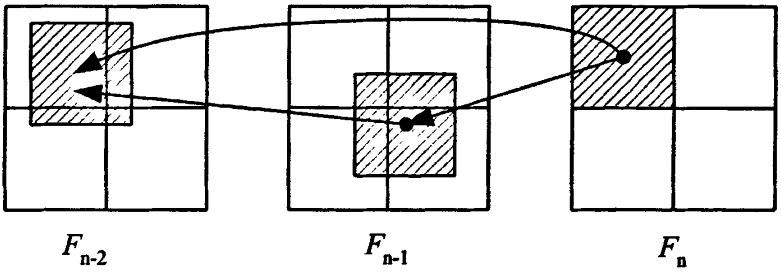


图 1-3 跳帧中运动矢量的计算

常用的计算方法有 FDVS(Forward Dominant Vector Selection)[63]，ADVS(Activity Dominant Vector Selection)[13]，加权中间值[10]，加权平均值[10]等。FDVS[63]方法选择覆盖区域最大的块的运动矢量作为重新估计后的结果；ADVS[13]则选择覆盖区域的四个块中活动最为剧烈的块的运动矢量，活动剧烈程度通常用非零系数个数来度量；加权中间值[10]方法则将每个运动矢量加权，再从中选择中间值，常用权重为覆盖面积或非零系数个数；而加权平均值[10]则选择加权后的平均值

比特率转码指在不改变图像尺寸和帧率的前提下降低码率，通常由调整变换

系数[68]，[69]和重新量化[27]-[31]来实现。由于变换系数的能量集中在低频部分，因此可以丢弃部分高频分量来达到降低比特率的目的，而不至于降低太多的图像质量；如果使用重新量化实现，不同的量化步长，宏块类型的使用也有所不同。比如在 H.264 帧内预测中，当量化步长较小时，会使用较多的 I4MB 的宏块类型；而随着量化步长逐渐增加，会使用越来越多的 I16MB。在文献[27]中，当前宏块所占用的比特流长度被作为衡量标准，来选择在重新量化后的宏块类型。另外，如果选择开环的视频转码系统，重新量化会带来误差，并且逐帧积累，形成漂移效应。文献[28]和[29]引入了一个与宏块类型相关的矩阵来对重新量化误差进行了补偿。文献[22]专门分析了 H.264 重新量化转码中存在的问题，并指出现在 MPEG-2 中常用的像素域开环的转码系统不适用于 H.264。

另外根据视频转码系统实现的结构，还可分为开环和闭环两种结构，开环结构转码速度较快，但会带来“漂移”效应，而闭环结构可以避免“漂移”效应，代价是速度较慢，图 1-4 和图 1-5 以重新量化转码为例分别给出了这两种结构的示意。

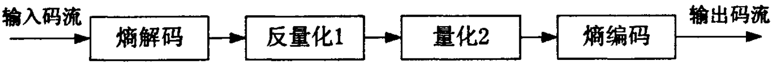


图 1-4 开环结构 (open-loop transcoding)

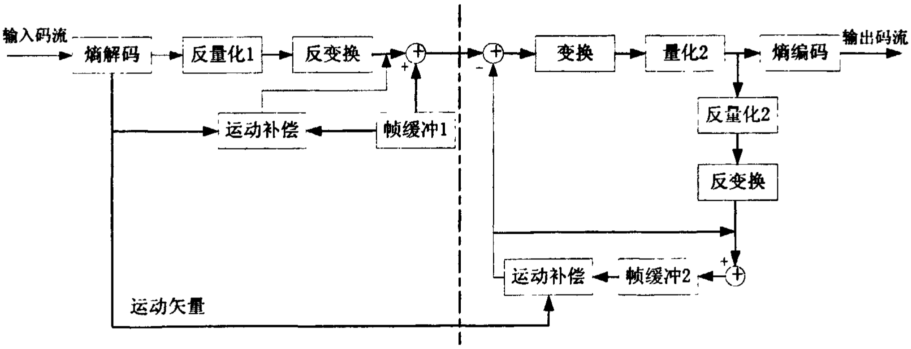


图 1-5 闭环结构 (close-loop transcoding)

根据不同的数据操作域，视频转码还可分为像素域转码和变换域转码。如上述开环结构即为变换域转码，而闭环结构则为像素域转码。在像素域转码中，输入比特流需要完全解码，得到像素域的帧数据；而在变换域转码中，不需要完全

解码，直接对变换域的变换系数进行操作，相对于前者，它的运算速度较快，由于没有对输入比特流进行完全解码，从而会造成“漂移”效应。文献[21]专门讨论了像素域运动补偿和变换域运动补偿所造成的偏差问题，并提出一个提升常量来补偿两者之间的偏差。

视频转码最容易实现的方式是将输入比特流完全解码，然后根据输出格式的要求进行重新编码（即像素域的级联转码,Cascaded Pixel-Domain Transcoding, CPDT），如图 1-6 所示。显然该方法也是运算复杂度最高的方法。为了提高重新编码速度，在视频转码中就需要充分利用解码得到的信息。

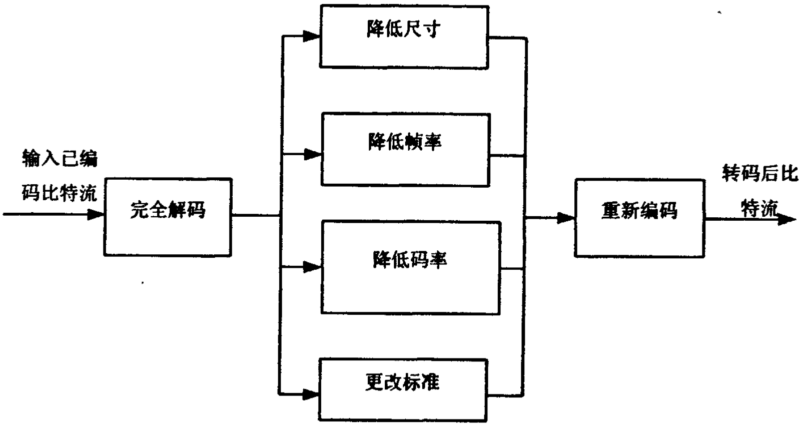


图 1-6 级联转码(Cascaded Pixel-Domain Transcoding, CPDT)

在基于 H.264 的视频转码研究中，大部分研究内容尚集中在标准间转码部分，而针对 H.264 的标准内转码的研究还不多见，本文针对 H.264 的标准内转码进行了深入的分析，并提出了一系列的解决方案。另外，对 H.264 的视频转码介绍见第 2 章。

1.2 可分级编码技术及国内外研究现状

为了能适应视频的复杂应用，常常需要编码不同的比特流，从而增加了编码端的负担。在视频可分级编码（Scalable Video Coding, SVC）中，编码端只需要一次性编码全分辨率下的比特流，不同应用的解码端只需要根据特定的应用环境接受部分码流进行解码即可，因此就减轻了编码端的负担。

在目前常用的运动补偿/变换混合编码方案中，由于采用了迭代的编码方式，

如果解码端接受数据有误，就会导致“漂移”效应。基于 MCTF（Motion Compensated Temporal Filtering ）的小波视频编码方案中完全抛弃了迭代编码方式，而采用了开环式系统，因此可以完全避免“漂移”效应。另外，该方案可以同时提供空间、时间和质量等三个方面的可分级编码。

事实上，MCTF 的使用甚至可以追溯到 20 年前[35]。Ohm[36]解决了在 MCTF 方案中由于运动估计造成的连接/不连接问题，并用镜像滤波（Quadrature Mirror Filters, QMF’s）对时间上的滤波过程进行了全面解释。有别于文献[36]，文献[37]提出了另一种滤波得到低通帧和高通帧的方法，并提出了一种 GOP 级（Group of Pictures, GOP）码率控制方案。提升方案的引入[38]-[40]解决了在 MCTF 中由于分数像素运动估计造成无法完全重构的问题。为了能有效的编码低通帧和高通帧，文献[41]，[42]提出了一种基于运动补偿的嵌入式零块编码器（Motion-compensated Embedded Zeroblocks Coder, MC-EZBC）。在低比特流情况下，运动矢量常常耗用较大的比特流，文献[37]提出了一种运动矢量的分层次编码方法，称为分层的变块尺寸匹配方案（Hierarchical Variable Size Block Matching, HVSBM）。为了能在空间上能得到更好的层次性，还有研究人员对小波域的 MCTF 编码方案进行了研究[43]-[45]。由于 MCTF 方案是目前基于小波的视频编码的核心技术之一[46]，对其在硬件上的性能分析可以在文献[47]中看到。

传统差分脉冲编码具有较高的压缩性能，而基于小波的 MCTF 技术可以实现可分级的编码方案。由 ITU-T（International Telecommunication Union Telecommunication Standardization Sector）的视频编码专家组（Video Coding Experts Group, VCEG）和 ISO/IEC（ISO-International Organization for Standardization, IEC-International Electrotechnical Commission）的活动图像专家组(Motion Picture Experts Group, MPEG) 组成的联合视频组（Joint Video Team, JVT）已经将 H.264 扩展到了 SVC [70]-[73]，并作为标准的一部分提出，从而融合了两者的优点。

在基于传统的 MCTF 的小波编码方案中，GOP 结构是固定的，因此无法处理视频序列中运动性质变化的情况，从而导致解码时视觉上的不连贯性以及压缩性能的损失，在目前的 MCTF 研究中，还尚未有文献对 MCTF 中 GOP 结构选择进行深入的分析。

1.3 本论文的研究内容和主要贡献

本论文的研究内容包括基于H.264的视频转码技术和基于MCTF的可分级编码技术两部分。

在基于H.264视频转码的研究中，本论文首次提出了一种基于H.264的同时包含空间、时间、质量三个方面的转码方案，论文尤其对重新编码阶段的宏块类型选择问题进行了深入的研究。在论文提出的基于H.264的视频转码方案中，输入和输出比特流均为H.264格式，输入的H.264比特流需要完全解码（像素域转码），在更改图像格式之后重新编码输出，其中图像格式的更改包含三个方面：空间分辨率，时间分辨率，图像质量。研究的主要内容包含以下几个方面：

1. 空间分辨率转码的帧内模式选择部分。本文基于图像缩放因子为2的情况，统计原始图像中4个对应宏块的非零系数比例（*nz_per*），并将统计结果与设定的阈值进行比较，进而选择当前宏块类型（I4MB/I16MB）。为了准确的确定*nz_per*阈值和重新量化参数（Q_r）的关系，本论文提出了一个$Th_I_Q_r$模型，该模型以指数曲线描述Q_r和*nz_per*阈值的关系。经过对指数模型进行线性化处理，得到一元线性回归模型，然后利用最小二乘法估计模型中的参数。为了能使得$Th_I_Q_r$模型能适应不同的视频序列，本论文提出了一种在实际转码过程中更新模型中参数的方法。$Th_I_Q_r$模型用于选择帧内宏块类型，不同的宏块类型对应着不同的帧内预测模式，本论文提出了一种快速的帧内预测模式选择方法。该方法充分利用输入原始图像中的宏块类型和帧内预测模式，进而大幅度降低了当前宏块的帧内预测模式选择时间。最后的实验结果表明，相对于全搜索法，在最大PSNR（Power Signal-to-Noise Ratio）损失约0.6dB前提下，本文方法的耗时仅为全搜索法的20%~25%，而重新编码时间约为全搜索法的70%左右。
2. 空间分辨率转码的帧间模式选择部分。与帧内模式选择类似，本文同样统计*nz_per*值，并用其作为划分当前宏块所在区域运动性质的准则，从而跳过部分候选宏块类型的测试，节省运算时间。在使用*nz_per*中，本文提出了$Th_P_Q_r$模型。与$Th_I_Q_r$模型类似，该模型同样使用指数曲线来描述Q_r和*nz_per*阈值的关系，并在实际转码过程中进行即时更新。根

据本文提出的方法，*Th_P_Q$_r$*模型可以快速的划分出当前宏块所在区域的运动性质，如对于运动缓慢的区域，就不再需要进行P8x8类型的测试，而对于运动剧烈的区域，就不再需要P16x16类型的测试，从而大幅度地节省了运算复杂度。另外，由于当前宏块的运动矢量是根据原始图像计算出来，这些初始计算的运动矢量并非一定精确，尤其是当Q_r较大时，因此需要运动矢量细化。本文提出了一种新的运动矢量细化方案，该方案中以*nz_per*作为运动矢量细化步长的准则，且随着Q_r的增加，运动矢量细化步长也逐步增加。从而保证了在运动较为剧烈的区域，运动矢量细化步长较长，而在运动平缓的区域，细化的步长较短。本文又进一步将*Th_P_Q$_r$*模型推广到了时间分辨率转码方面，并取得了较好的效果。最后的实验表明，相比于全搜索法，在最大PSNR损失约1.1dB前提下，本文方法可以将总编码速度提高15-20倍，若仅考虑选择宏块类型部分，则可以提高约35倍。

3. 本文首次提出基于分类方法在视频转码中快速选择宏块类型。并利用该方法，本文首次完成了基于H.264的同时包行空间、时间、质量三个方面的转码方案。该转码方案中从输入比特流中提取特征向量，主要内容有：原始图像中宏块类型、残差数据、运动矢量、量化参数等，并将这些特征向量输入到离线训练完毕的支持向量机模型，从而预测出目标宏块类型。通过最后的实验，相比于全搜索法，在最大PSNR损失约1.2dB前提下，本文方法可以将总编码速度提高12倍，若仅考虑选择宏块类型部分耗时，则可以提高约30倍。

在基于传统的 MCTF 的可分级编码技术中，GOP 尺寸是固定的，它与时间上的滤波器类型和时间上分解层次有关。但实际的视频序列是千差万别的，因此该方案无法适应实际序列中的运动性质的变换，针对该部分，本文的主要研究内容如下：

1. 根据视频中运动性质的变换，本文提出了一种类haar的MCTF编码方案。该方案包含了GOP结构选择和时间分解层次确定两部分，其中GOP结构根据帧间的互信息值自适应的确定，又分为GOP尺寸选择和低通帧选择两部分。在本文提出的方案中，同时利用GOP内平均互信息值和标准差来控制GOP尺寸，从而选定的GOP尺寸不仅能根据运动类型的变化自适

应的改变，而且同一个GOP内部的运动类型也能保持一致。在选定的GOP内部，本文首次提出了一种低通帧的选择方案，该方案基于互信息技术，从一个GOP内提取出与其余帧最具相关性的帧。进而当解码端时间上的解码层次较少时，本文方法可以解码得到的帧更能代表GOP内的运动特征，而且该方法还可以进一步提高压缩性能。另外，本文根据选择的GOP结构，提出了一种自适应的时间分解过程。该分解过程尽可能的降低帧对间的距离，从而减少运动预测残差，节省比特流。根据实验结果，对于运动性质有明显变化（尤其具有频繁的镜头切换）或运动较为剧烈的序列，本文的GOP结构选择方法能较大地提高压缩性能。

论文的章节安排如下：第 1 章为绪论，主要介绍了视频转码技术和可分级编码技术的概况，第 2 章讨论了基于 H.264 的视频转码技术，第 3 章主要内容为空间分辨率转码，其中包括帧内模式选择和帧间模式选择两部分，第 4 章中为时间分辨率转码部分，主要包括帧间模式选择部分，在第 5 章中，本文基于支持向量机首次提出了一种同时支持空间、时间、码率三个方面的 H.264 转码系统，第 6 章对基于 MCTF 的视频编码方案进行了讨论，并提出了一种帧组结构选择方案，最后第 7 章为全文的总结以及后续工作的展望。

第 2 章 基于 H.264 的视频转码

H.264 标准是 ITU-T 的 VCEG 和 ISO/IEC 的 MPEG 组成的 JVT 联合开发的标准,也称为 MPEG-4 AVC（Advanced Video Coding），并作为 MPEG-4 的第 10 部分颁布。为取得较高的压缩性能，H.264 使用的技术包括分数像素的运动估计，4x4 整数变换，内容自适应的变长编码（Context-Adaptive Variable Length Coding, CAVLC），内容自适应的二进制算术编码（Context-Adaptive Binary Arithmetic Coding, CABAC），帧内预测（intra prediction），帧间预测（inter prediction），多参考帧，环路滤波(loop filtering)等技术。

2.1 H.264 编码技术

H.264 中使用了 1/4 像素精度（亮度分量）和 1/8 像素精度（色度）的运动估计，并采用了 6 抽头的滤波器进行 1/2 像素的插值，使得运动预测残差更少，进而提高压缩性能；视频压缩编码中以往的常用单位为 8x8 块（如 H.263），在 H.264 中采用小尺寸的 4x4 块，由于变换块的尺寸变小了，运动物体的划分就更为精确。这种情况下，图像变换过程中的计算量小了，而且在运动物体边缘的衔接误差也大为减少；H.264 还采用了两种熵编码方式，CAVLC 和 CABAC，其中 CABAC 比 CAVLC 运算复杂度要高，同样压缩效果更好。在 H.264 中，可采用多个参数帧的运动估计，即在编码器的缓存中存有多个刚刚重构的参考帧，编码器从其中选择一个给出更好的编码效果的作为参考帧，并指出是哪个帧被用于预测，这样就可获得比只用一个参考帧更好的编码效果。在基于块的运动补偿中，在重构图像的块边缘上会有块效应，从而导致图像质量下降，在 H.264 中采用了环路滤波技术对块效应进行处理，从而提高了图像质量。下面对本论文主要的涉及部分帧内预测和帧间预测进行详细讨论。

2.1.1 帧内预测

帧内预测利用邻近像素来估计当前块的像素值，并对预测残差进行编码，从而充分利用了图像的空间相关性。在 H.264 的帧内预测模式中，宏块的亮度分量可以使用 I4MB、I8MB、I16MB 等三种类型。其中 I8MB 仅使用在 FREXT（Fidelity Range Extensions）中，本文不对其进行讨论。

如果宏块使用了 I4MB 类型，整个宏块将被划分成 16 个 4x4 子块，每个 4x4 子块有各自的预测模式，预测模式共有 9 种候选类别。图 2-1 给出了一个 4x4 子块（像素：a、b、c、…、n、o、p）及其相邻像素（X、A、B、…、K、L），图 2-2 给出了所有 9 种亮度预测模式。

X	A	B	C	D	E	F	G	H
I	a	b	c	d				
J	e	f	g	h				
K	i	j	k	l				
L	m	n	o	p				

图 2-1 4×4 子块及其相邻像素

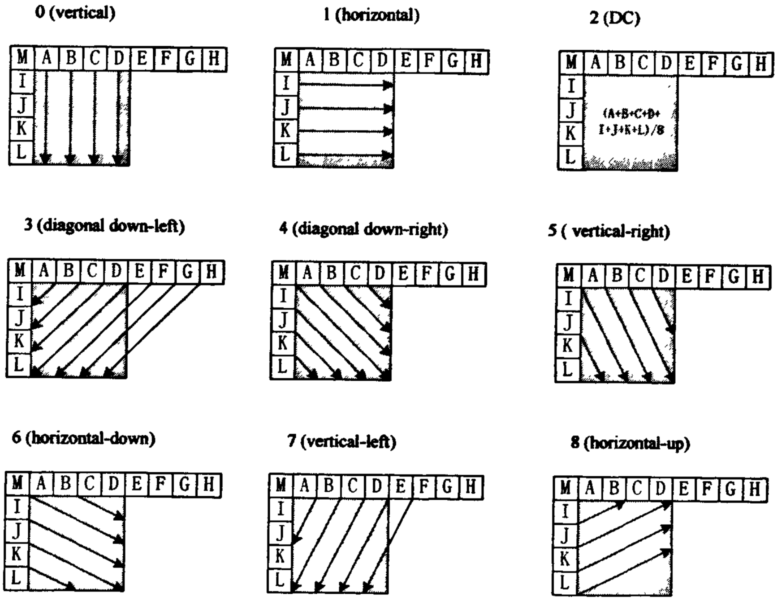


图 2-2 4×4 亮度预测模式

如果宏块使用 I16MB 类型，则整个宏块会使用同一个预测模式。针对 I16MB 类型，H.264 提供了四种帧内预测模式，它们分别是竖直、水平 、DC 和 PLANE，其中前三种模式与图 2-2 类似，PLANE 模式见下图：

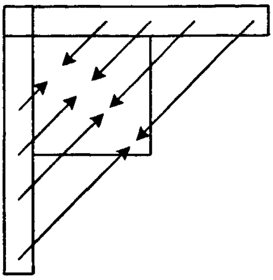


图 2-3 I16MB PLANE 亮度预测模式

另外，色度分量具有与 I16MB 宏块相同的帧内预测模式，不同之处在于色度分量在 H.264 基本层（baseline profile）中的尺寸为 8x8。

2.1.2 帧间预测

帧间预测中利用参考帧来补偿当前帧，并对预测残差进行编码，它充分利用了视频中时间上的相关性，具体的编解码过程见图 2-4。

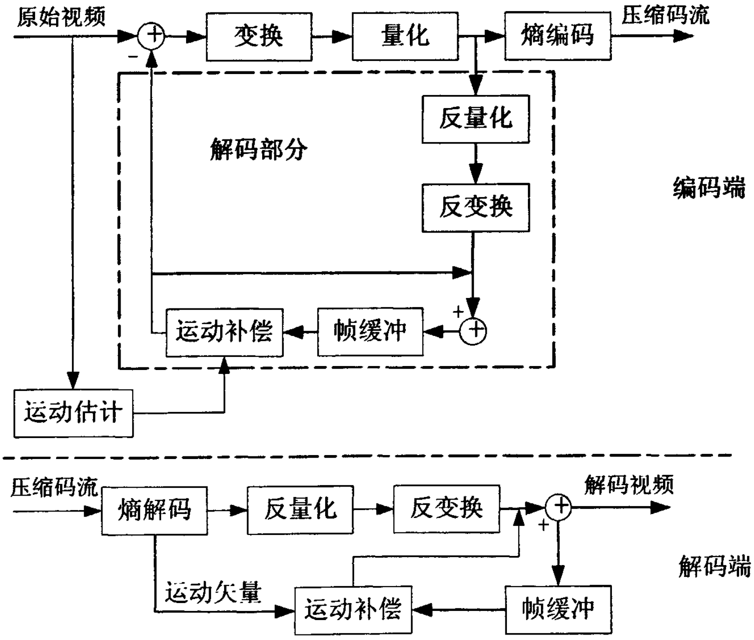


图 2-4 运动预测/变换编解码过程

H.264 中使用了变块大小的运动预测，宏块可以划分为 16x16（P16x16）、16x8（P16x8）、8x16（P8x16）、和 8x8（P8x8）；如果使用 P8x8 的宏块类型，则每个 8x8 的块还可以进一步划分为 8x4、4x8、或 4x4（见图 2-5）。另外，H.264 还使

用了 SKIP 宏块类型，该类型比较适用于运动非常平稳的区域。

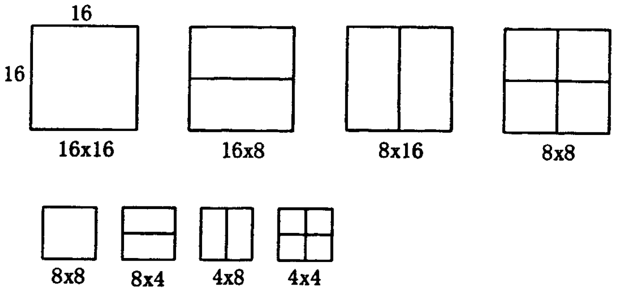


图 2-5 宏块的尺寸划分

在 H.264 中，I 帧（Intra frame）仅使用帧内预测宏块类型（I4MB 或 I16MB），P（Inter frame）帧则可能会用到所有的宏块类型（I4MB、I16MB、SKIP、P16x16、P16x8、P8x16、P8x8 等），有关帧内/帧间预测模式的细节讨论请参考文献[32]、[33]。

2.2 基于 H.264 的视频转码及研究现状

H.264 使用了可变的宏块类型，从而提高了压缩性能，但也是 H.264 编码器中最主要的耗时部分。如在图 2-4 中运动预测部分，如果使用全搜索法，需要对每种帧间宏块类型进行运动估计，进而得到每种宏块类型的预测残差，并选择预测残差最小者为最终类型。因此，宏块类型选择也是基于 H.264 的视频编码的研究热点之一[23]-[26]。在视频转码中同样需要重新选择宏块类型，如在重新量化转码中，随着量化参数的增大，原有的宏块类型不再合适，因此需要重新选择。例如，在量化参数较小时（高比特率），会使用较多的 P8x8 类型，而随着量化参数的递增，会使用越来越多的 P16x16 类型；在空间分辨率转码中，如图像缩放因子为 2，当前图像中一个宏块对应着原始图像中 4 个宏块，就需要利用这 4 个宏块的信息为当前宏块选择类型；在时间分辨率转码中，有些帧需要被丢弃，间接导致了保留帧之间的运动加剧，因此保留帧中的宏块类型不再合适，需要重新选择。转码中的宏块类型选择与编码中的宏块类型选择的最大的不同点在于，在转码中，可以利用丰富的解码信息来快速选择宏块类型过程，从而节省计算量。

已有一些研究人员针对基于 H.264 的转码技术进行了研究，文献[74]分析了 MEPG-2 到 H.264 的变换域转码中，由于插值和量化步长的不同导致的漂移效应，并推导了插值误差的理论形式，进而论证了插值误差是导致漂移效应的主因。文

献[75]提出了一种帧内模式选择方案，该方案基于 MPEG-2 到 H.264 的变换域转码。方案中提出了一种 DCT 系数到整数变换系数的直接转换方法，并利用时间上帧间的关联性在变换域预测宏块类型。文献[76]和[79]则利用比特率转码中全搜索法中宏块类型的分布特征，另外[76]中还提出了一种帧间类型选择方法，该方案中通过控制运动矢量细化的步长，尽可能的在运算复杂度和压缩性能之间取得平衡。文献[78]提出的是一种从 MPEG-2 到 H.264 的变换域转码方法，该方法利用输入比特流中的 DCT 系数来预测输出的帧内宏块类型。输出 H.264 的帧内宏块类型（I16MB/I4MB）是由输入 DCT 系数的方差来决定。文献[79]则对 H.263 到 H.264 的转码中帧内/帧间宏块类型选择进行了讨论，在帧内宏块类型选择中利用 H.263 的预测残差估计 H.264 的结果，另外为了节省计算量，文献在运动矢量细化中中只使用了 1/4 像素的步长。

现有基于 H.264 的视频转码研究中，大部分内容尚集中在标准间转码部分，如 MPEG-4 到 H.264，H.263 到 H.264 等。而针对 H.264 的标准内转码的研究还并不多见，本文针对 H.264 的标准内转码进行比较全面的研究，尤其重新编码阶段的宏块类型选择进行了深入的分析，对并提出了一系列的解决方案。

如绪论所述，根据操作数据的性质，视频转码可分为像素域转码和变换域转码。其中后者的运算速度较快，但会带来“漂移”效应，不少研究人员提出了控制“漂移”效应的方案[100]-[102]，其中一个核心就是变换域的运动补偿（DCT domain Motion Compensation , DCT-MC）[101]，如下图所示：

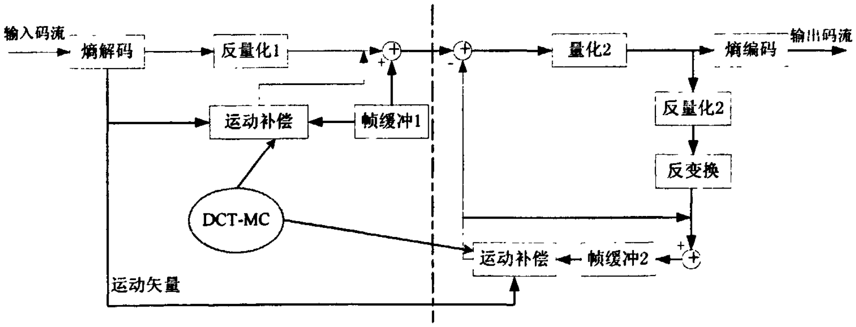


图 2-6 变换域转码

但是目前所提出的变换域的运动补偿方法仅限于整数像素或线性插值的半像素运动估计[100]，[134]，[135]，而 H.264 采用了 1/4 像素的运动估计，在计

算 1/2 像素值更是使用了 6 抽头的插值滤波器，从而极大地增大了变换域的运动补偿的难度和可行性。另外 H.264 还采用了复杂的环路滤波技术，来消除运动补偿所造成的块效应，而环路滤波的操作数据对象也是像素值，如果使用变换域转码，必然进一步扩大“漂移”效应，从而恶化图像质量。

基于上述讨论，在本文提出的基于 H.264 的视频转码中，输入和输出比特流均为 H.264 格式，输入的 H.264 比特流需要完全解码（像素域转码），在更改图像格式之后重新编码输出，其中图像格式的更改包含三个方面：空间分辨率，时间分辨率，比特率。其中比特率转码由重新量化实现。由于本文在空间分辨率转码和时间分辨率转码均考虑了重新量化的因素，因此不对重新量化转码单独讨论，第 3 章到第 5 章分别讨论空间分辨率转码，时间分辨率转码，时间-空间分辨率转码三部分。

2.3 本章小结

本章介绍了 H.264 中所使用的各种技术，运动估计，多参考帧，环路滤波等。并详细介绍了帧内预测和帧间预测两部分内容，另外，本章还讨论了基于 H.264 的视频转码技术及研究现状，对基于 H.264 的视频转码中所存在的问题进行了分析，并指出完全的变换域转码并不适合 H.264 转码，因此本文采用了像素域的视频转码，最后并简要介绍了本论文中视频转码的研究范围。

第 3 章 空间分辨率转码

3.1 引言

空间分辨率转码也就是图像尺寸转码，图像尺寸缩放因子一般分为整数和任意比例两种。在文献[4]提出的基于 H.264 的空间分辨率转码中，宏块类型的选择是基于常见的率失真最小化代价函数的准则。方案中首先为每种宏块类型计算各自的候选运动矢量，这些候选运动矢量用来计算率失真代价，并选择具有最小代价的宏块类型。文献[3]则使用了重构图像的 DCT 系数的能量作为选择宏块类型的依据。

本文提出的空间分辨率转码中包含了比特率转码，并通过重新量化实现。不同的量化步长，宏块类型的使用也有所不同。比如在 H.264 帧内预测中，当量化步长较小时，会使用较多的 I4MB 宏块类型；而随着量化步长逐渐增加，会使用越来越多的 I16MB。在文献[27]中，当前宏块所占用的比特流长度被作为衡量标准，来选择在重新量化后的宏块类型。另外，如果选择开环的视频转码系统，重新量化会带来误差，并且逐帧积累，形成漂移效应。文献[28]和[29]引入了一个与宏块类型相关的矩阵来对重新量化误差进行了补偿。

本文使用的缩放因子为 2，即降低尺寸后图像中的一个宏块对应着原始图像中的四个宏块。本文将称“降低尺寸后图像”为“当前图像”，称当前图像的待编码宏块为“当前宏块”，称当前宏块所对应的原始图像中的四个宏块为“四个对应宏块”。在转码中的需要为当前宏块选择类型，如图 3-1。

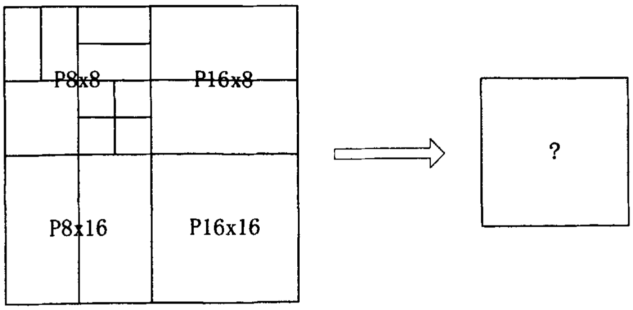


图 3-1 当前宏块与四个对应宏块

当前宏块在转码中重新编码端需要选择合适的宏块类型、运动矢量等参数。四个对应宏块是已编码宏块，在完全解码后可以提取相应的宏块类型、运动矢量、残差数据等信息。其中残差数据可以由所有非零系数（经过 DCT 变换及量化后的结果）的比例来表征，本文称之为"*nz_per*"。*nz_per* 的值越大，意味着残差数据越多，该区域细节较为丰富（帧内模式）或者运动较为剧烈（帧间模式）。另外由于考虑了重新量化，故转码系统存在两个不同的量化参数，其中一个是输入原始图像中的量化参数，简称为 Q_i，另一个是在重新编码过程中的量化参数，简称为 Q_r。

本文利用 *nz_per* 作为选择帧内/帧间模式的准则，并提出了 Th_Q_r 模型，它分别应用到帧内模式选择部分（$Th\_I\text{-}Q_r$ 模型）和帧间模式选择部分（$Th\_P\text{-}Q_r$ 模型）。$Th\_I\text{-}Q_r$ 模型是用来区分当前宏块的类型（I16MB/I4MB）；而 $Th\_P\text{-}Q_r$ 模型是用来区分当前宏块所在区域的运动性质。下面将对其分别进行讨论。

3.2 $Th\_I\text{-}Q_r$ 模型

在帧内模式选择中，首先需要为当前宏块选择类型（I16MB/I4MB），然后根据不同的宏块类型选择预测模式，具体的帧内预测模式见图 2-2 和图 2-3。在图 3-2 所示的实验中，输入的量化参数 Q_i=20，所有帧都使用帧内预测模式；当前图像选择具有最小率失真代价的最佳宏块类型（为了与帧间模式选择保持一致，本文简称为全搜索法），实验中选择了多个重新量化参数进行测试 Q_r={25，30，35，40，45}。针对当前图像所选择的每种宏块类型（I4MB 或 I16MB），本文统计了其对应原始图像中的 *nz_per* 的平均值。图 3-2 中横坐标是 Q_r，纵坐标是 *nz_per*，左边的型条表示 I16MB 类型的比例，右边的型条表示 I4MB 类型的比例。从该实验结果可以看出，不论 Q_r 大小，当前图像中类型为 I16MB 的宏块对应的 *nz_per* 平均值都远远小于 I4MB 宏块所对应的 *nz_per* 的平均值。图 3-2 给出的实验结果是多个视频序列的平均（除非特别说明，本文给出的实验结果都是经过了大量的实验，综合多个视频序列给出的平均结果）。

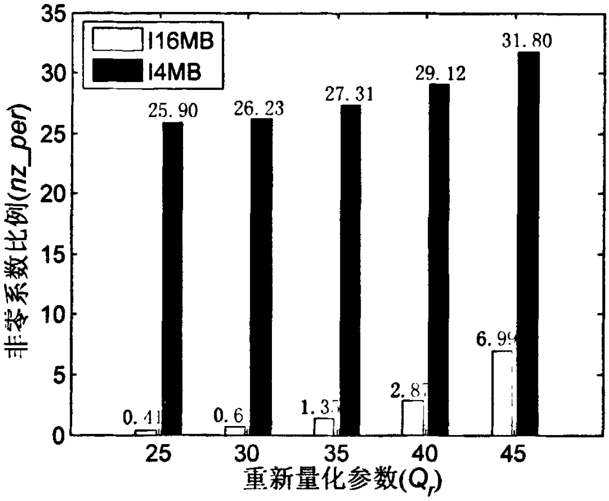


图 3-2 *nz_per*、*Q_r* 和 I16MB/I4MB 的比例

根据上图实验结果，可以说当前宏块对应的 *nz_per* 值较小时，它选择为 I16MB 的可能性较大；当前宏块对应的 *nz_per* 值较大时，则选择 I4MB 的概率更高。下面使用另一个实验来进一步证明该结论。在图 3-3 的实验中，针对原始图像中每个 *nz_per* 值，都统计出使用全搜索法所选择的当前宏块类型，并给出了三个 Q_r 值的结果 Q_r={30，40，50}。图中横坐标是每个 *nz_per* 值，纵坐标是当前宏块选择 I16MB 类型的概率。从这个实验结果可以很明显的看出，当 *nz_per* 值较小时，当前宏块使用 I16MB 的概率就非常高，而 *nz_per* 值较大时，当前宏块使用 I4MB 的概率很高。另外，随着 Q_r 的递增，相同的 *nz_per* 值对应的当前宏块使用 I16MB 的概率不断递增。比如当 *nz_per*=10（如图点划线所示），在 Q_r=30 时，使用 I16MB 的概率极低；当 Q_r=40 时，使用 I16MB 的概率上升到了 80%左右；而当 Q_r=50 时，概率值几乎达 100%。

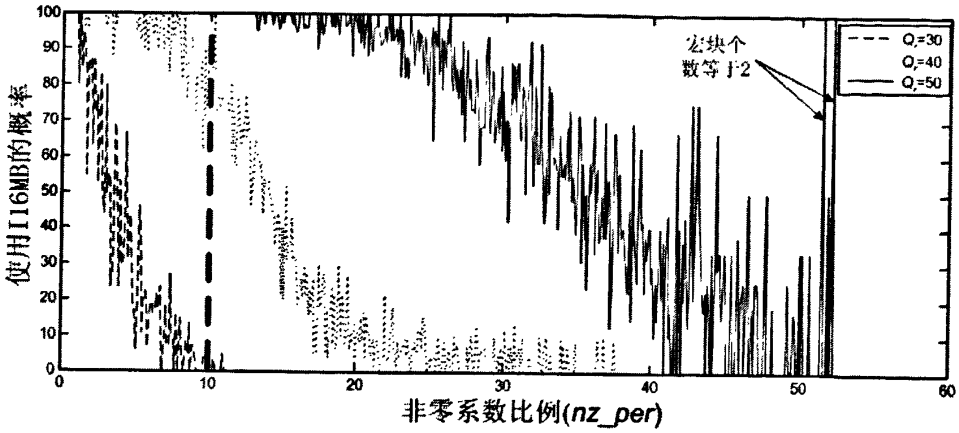


图 3-3 不同的 Q_r 与选择 I16MB 的概率

3.2.1 *Th_I*-Q_r 模型的引入

从上述分析可以看出，*nz_per* 值可以作为划分 I16MB/I4MB 的依据，如果当前宏块对应的 *nz_per* 值小于某个阈值（本文称之为 *Th_I*），就可以直接选择为 I16MB 类型，否则选择 I4MB。但是如何确定这个阈值？从图 3-3 结果可以看出，该阈值显然与 Q_r 相关。图 3-4 给出了部分序列的实验结果，实验中为每个 Q_r 选择一个固定的阈值 *Th_I*，如果当前宏块对应的 *nz_per* 值小于这个固定的阈值，就使用 I16MB 宏块类型，否则使用 I4MB 类型。图中右半部分表示了针对每个 Q_r 所选择的固定阈值 *Th_I*，左半部分表示这种固定的阈值方法与全搜索法的比较，两者压缩性能几乎完全相同。

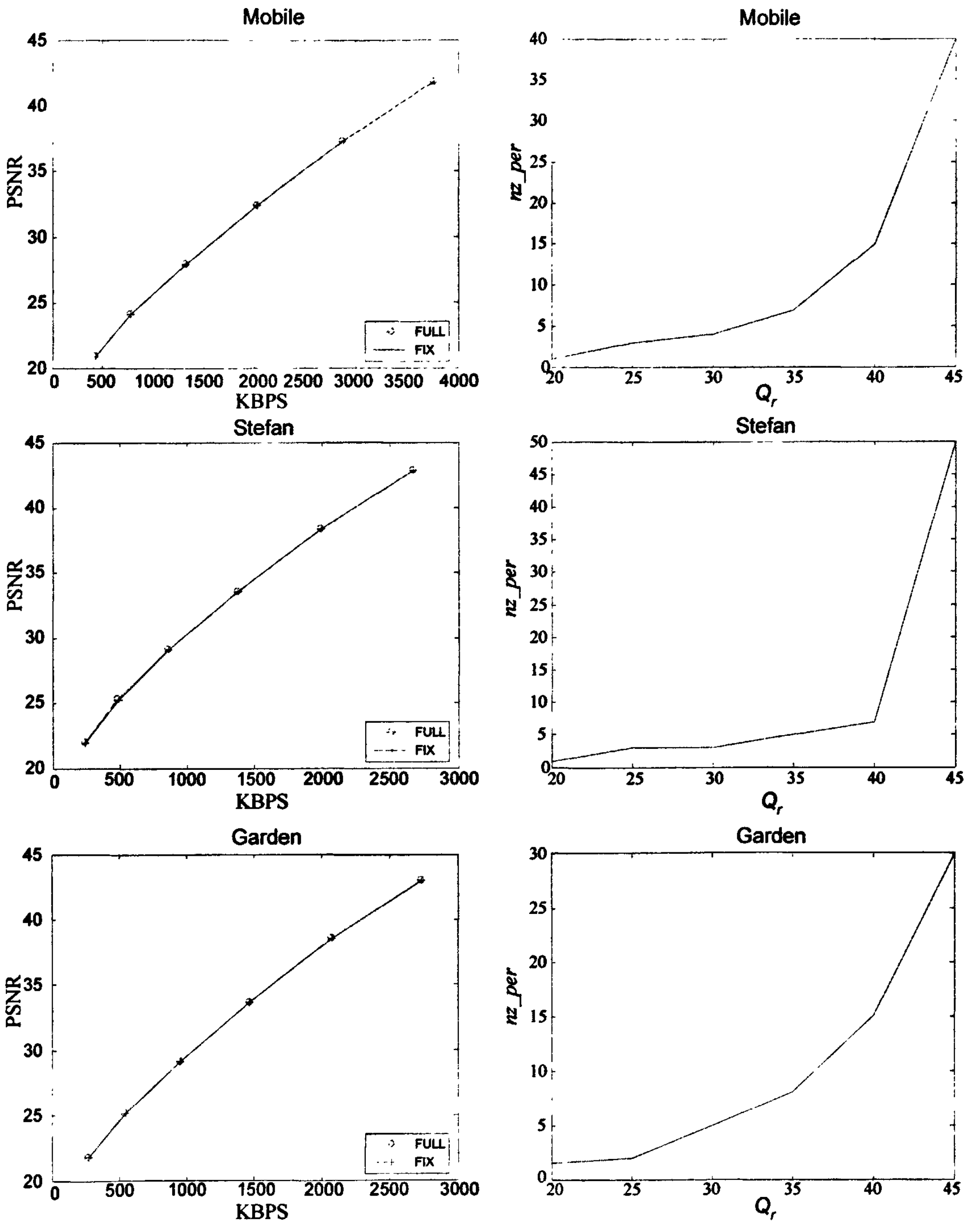


图 3-4 部分序列 Q_r 与 Th_I 的关系

从这个实验结果可以看出，在保证了压缩性能不受损失的前提下，Q_r 与 Th_I 的关系很接近于一个指数曲线。本文使用指数曲线来描述 Q_r 与 Th_I 的关系，并称之为 Th_I-Q_r 模型。数学模型见等式(3.1)，其中 a 和 b 是指数曲线的参数。

$$Th\_I = ae^{bQ_r} \qquad (3.1)$$

3.2.2 参数 a 和 b 的估计

首先对等式(3.1)进行线性化处理，得到一元线性回归模型[34]，然后利用最小二乘法估计模型中的参数 a 和 b。对等式(3.1)两边同时取对数就可以得到等式(3.2)。

$$\ln Th\_I = \ln a + bQ_r \tag{3.2}$$

分别用 y 和 c 来代替 *Th_I* 和 a，替代等式如下：

$$y = \ln Th\_I \tag{3.3}$$

$$c = \ln a \tag{3.4}$$

这样等式(3.1)就可以线性化为：

$$y = c + bQ_r \tag{3.5}$$

根据最小二乘法估计，参数 c 和 b 可以由下式得出：

$$b = \frac{n\sum_{i=1}^{n} Q_{ri} y_i - \sum_{i=1}^{n} Q_{ri} \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} Q_{ri}^2 - (\sum_{i=1}^{n} Q_{ri})^2} \tag{3.6}$$

$$c = \frac{\sum_{i=1}^{n} y_i - b\sum_{i=1}^{n} Q_{ri}}{n} \tag{3.7}$$

将等式(3.3)和(3.4)代入等式(3.6)和(3.7)，可以得到参数 a 和 b 的估计值如下：

$$b = \frac{n\sum_{i=1}^{n} Q_{ri} \ln Th\_I_i - \sum_{i=1}^{n} Q_{ri} \sum_{i=1}^{n} \ln Th\_I_i}{n\sum_{i=1}^{n} Q_{ri}^2 - (\sum_{i=1}^{n} Q_{ri})^2} \tag{3.8}$$

$$a = \exp\left(\frac{\sum_{i=1}^{n} \ln Th\_I_i - b\sum_{i=1}^{n} Q_{ri}}{n}\right) \tag{3.9}$$

其中

n 是总的帧数目。

Q_{ri} 是第 i 帧的量化参数的平均值。

Th_I_i是第 i 帧的阈值 nz_per 的平均值。

下面给出一个具体的例子。假设原始的 Th_I-Q_r集合为 Q_r = [20, 25, 30, 35, 40, 45]，Th_I = [1.07, 2.15, 4.10, 7.52, 12.99, 33.69]。利用等式(3.8)和(3.9)，a 和 b 的估计值分别是 a = 0.7932，b= 0.1320。将 a 和 b 的估计值代入等式(3.1)，就可以得到完整的 Th_I-Q_r曲线。实验结果见图 3-5，其中实线为真实的 Th_I-Q_r曲线，虚线由参数 a 和 b 的估计值计算得到。由此可以看出，Th_I-Q_r模型比较准确地描述了 Th_I 和 Q_r的关系。

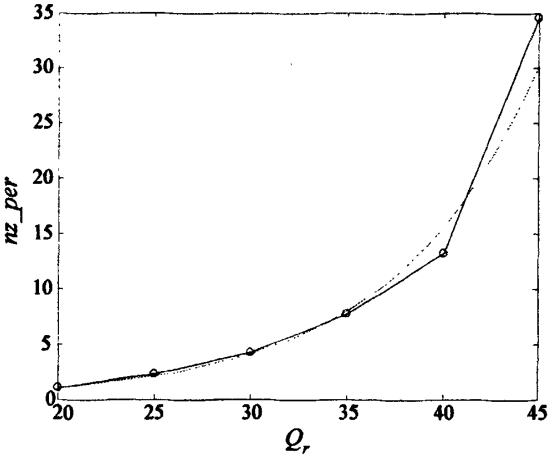


图 3-5 参数 a 和 b 的估计

3.2.3 参数 a 和 b 的更新

系统在转码前无法预知适合当前视频序列的最佳 Th_I-Q_r 集合。因此，需要针对大量的视频序列进行测试，得到一个初始的 Th_I-Q_r集合，根据等式(3.8)和(3.9)估计出参数 a 和 b 的初始值，并在转码的过程中即时更新。下面给出计算和更新参数 a 和 b 的伪代码。

```
a:  输入初始 Th_I-$Q_r$集合;
b:  使用当前 Th_I-$Q_r$集合，根据等式(3.8)和(3.9)估计参数 a 和 b;
c:  使用参数 a 和 b 根据等式(3.1)计算 Th_I;
d:  计算阈值 Th_low 和 Th_high:
        Th_low    =  0.9×Th_I;
        Th_high   =  1.1×Th_I;
e:  for(当前帧所有宏块)
```

```
    {
        为当前宏块统计原始图像中的 nz_per 值；
        if(nz_per < Th_low)          选择 I16MB 的宏类型；
        else if (nz_per > Th_high)   选择 I4MB 的宏类型；
        else
        {
            比较 I4MB 和 I16MB，选择最佳宏块类型；
            if(最佳宏块类型是 I4MB)    num_4++;
            else                       num_16++
        }
    }
    if(num_16 > num_4)
        将 Th_high 和 $Q_r$ 添加到 $Th\_I\text{-}Q_r$ 集合；
    else
        将 Th_low 和 $Q_r$ 添加到 $Th\_I\text{-}Q_r$ 集合；
f:  go to b，转码下一帧；
```

从上述伪代码可以看出，为了能更新参数 a 和 b，实际使用的阈值分别为 *Th_low* 和 *Th_high*。如果 *nz_per* 大于 *Th_high*，则直接选择为 I4MB，如果 *nz_per* 值小于 *Th_low*，直接选择为 I16MB，如果 *nz_per* 值处在这两者之间，则会在 I16MB 和 I4MB 中选择较好的宏块类型。当前帧编码结束时，统计当 *nz_per* 值处在两个阈值之间时最终选择的宏块类型的情况，如果选择的 I16MB 较多（num_16 > num_4），意味着当前阈值过低，可以适当调高（将 *Th_high* 和 Q_r 添加到 $Th\_I\text{-}Q_r$ 集合）；否则，可以适当调低。

3.3 $Th_P\text{-}Q_r$模型

在 H.264 中，P 帧中允许使用的宏块类型包括 I4MB、I16MB、SKIP、P16x16、P16x8、P8x16、P8x8 等，其中 P16x16 和 SKIP 类型常出现在运动较为平缓的区域，P8x8 和 I4MB 宏块常出现在运动较为剧烈的区域。

下面考虑当前宏块所对应的 *nz_per* 值与其使用不同宏块类型的概率。实验结果如图 3-6 所示，横坐标是当前宏块所对应的 *nz_per* 的值，纵坐标是当前宏块

使用 P16x16 或 SKIP 类型的概率（由全搜索法得出）。三条曲线分别表示不同的重新量化参数 Q_r。从图中很明显可以看出，随着 *nz_per* 的值递增，当前宏块使用 P16x16 或 SKIP 的概率递减，另外，随着量化参数的增大，相同的 *nz_per* 值所对应的宏块使用 P16x16 或 SKIP 的概率也在递增。该性质与 3.2 中所述的 $Th\_I\text{-}Q_r$ 极为类似。图 3-7 则给出了 P8x8 或 I4MB 类型与 *nz_per* 值的关系，根据该图，当 *nz_per* 值大于某个阈值时，当前宏块可以直接选择为 P8x8 或 I4MB 类型，另外，该阈值显然随着 Q_r 的增大而递增。

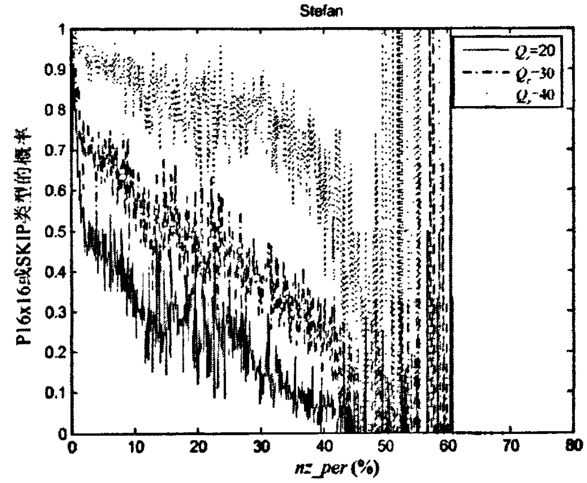


图 3-6 当前宏块对应的 *nz_per* 值与宏块类型（P16x16 或 SKIP）的关系

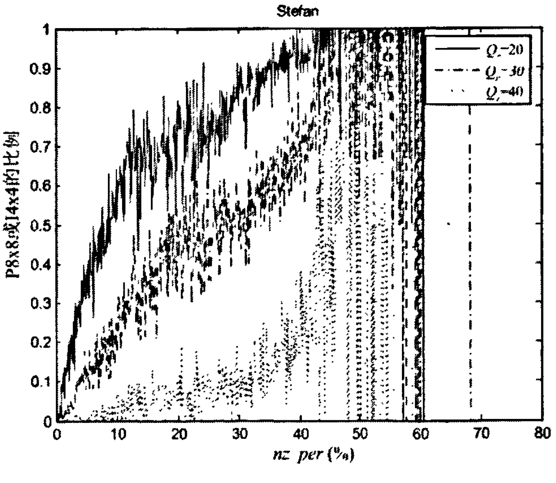


图 3-7 当前宏块对应的 *nz_per* 值与宏块类型（P8x8 或 I4MB）的关系

注：图中当 *nz_per* 值较大时（如 50%-60%），三条曲线出现了混叠，而且宏块类型的比例不再符合上述规律，原因是出在这个区域的宏块个数较少，因此不具备统计规律。

根据上述分析，给出下个实验，见图 3-8 和图 3-9。图 3-8 实验中为每个 Q_r 选择一个固定的阈值（*Th_L*），如果当前宏块对应的 *nz_per* 值小于这个固定的阈值，就使用 P16x16 宏块类型（SKIP 类型的使用在 P16x16 编码时进行），否则使用全搜索法选择最佳的宏块类型。图的右半部分表示为每个 Q_r 与所选择的固定阈值 *Th_L*；对应左半部分表示这种固定的阈值方法与全搜索法的压缩性能比较，两者性能几乎完全相同。图 3-9 中的实验则为 P8x8 和 I4MB 选择一个固定阈值（*Th_H*），当 *nz_per* 值大于这个阈值时，就直接选择为 P8x8 或 I4MB，否则使用全搜索法选择。

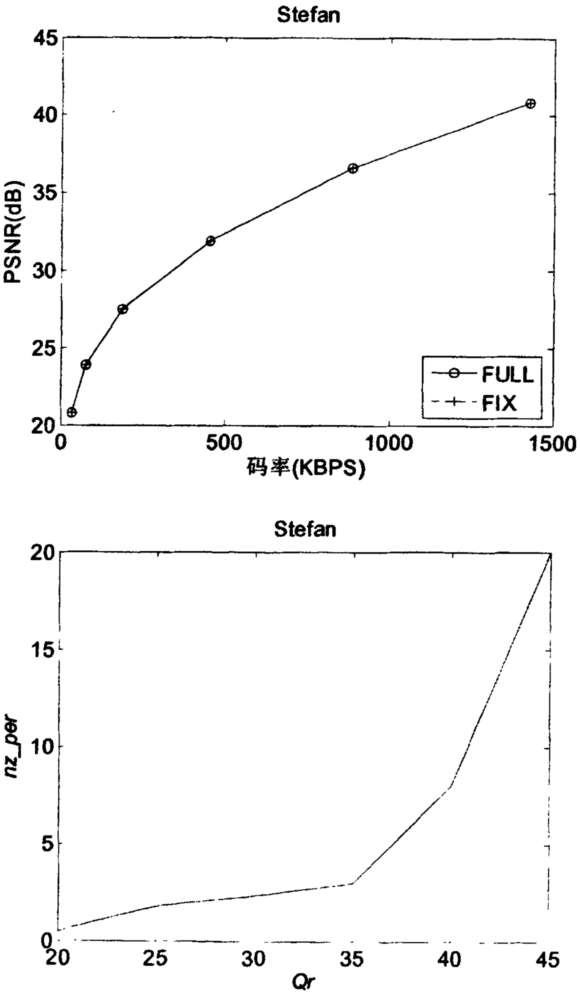


图 3-8 Q_r 和 *Th_L* 的关系

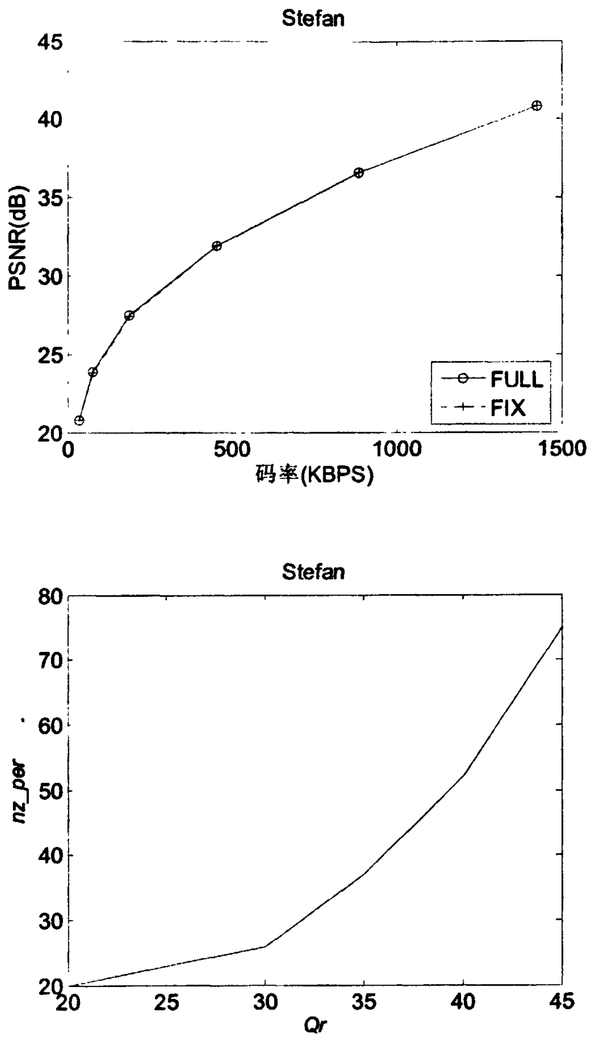


图 3-9 Q_r 和 Th_H 的关系

从上述两个实验显然可以看出，在保证压缩性能不受损失前提下，阈值 Th_L 和 Th_H 与 Q_r 的关系都近似指数曲线，这与 Th_I 与 Q_r 的关系类似。因此本文也使用指数曲线来描述 $Th\_P\text{-}Q_r$ （包含 $Th\_L\text{-}Q_r$ 集合和 $Th\_H\text{-}Q_r$ 集合两部分）模型：

$$Th\_L = ae^{bQ_r} \tag{3.10}$$

$$Th\_H = ae^{bQ_r} \tag{3.11}$$

与 $Th\_I\text{-}Q_r$ 模型类似，参数 a 和 b 也可以使用等式(3.8)和(3.9)估计。

3.3.1 参数 a 和 b 的更新

$Th_P\text{-}Q_r$模型中参数的更新方法与$Th_I\text{-}Q_r$模型极为类似，也需要输入初始的$Th_H\text{-}Q_r$和$Th_L\text{-}Q_r$集合用来估计等式(3.10)和(3.11)中参数 a 和 b 的初值，在编码的过程中对 a 和 b 更新。下面给出更新参数 a 和 b 的伪代码。

```
a:  输入初始的 $Th\_H\text{-}Q_r$ 和 $Th\_L\text{-}Q_r$ 集合；
b:  利用当前 $Th\_H\text{-}Q_r$ 集合和 $Th\_L\text{-}Q_r$ 集合计算各自的参数 a 和 b；
c:  使用等式(3.10)计算 $Th\_L$；使用等式(3.11)计算 $Th\_H$；
e:  for(当前帧所有宏块)
    {
        使用 3.5.1 所描述的过程选择宏块类型；
        if($nz\_per > 0.9\times Th\_H$ 并且 $nz\_per < Th\_H$)
        {
            if (选择的宏块类型是 P8x8)        num_high_8++;
            else                              num_low_8++;
        }
        if($nz\_per < 1.1\times Th\_L$ 并且 $nz\_per > Th\_L$)
        {
            if(如果选择的宏块类型是 P16x16)   num_high_16++;
            else                              num_low_16++;
        }
    }
    percentage_1 = num_high_8/(num_high_8+num_low_8);
    percentage_2 = num_high_16/(num_high_16+num_low_16);
    if(percentage_1 < 0.10)   将 $1.1\times Th\_H$ 和 $Q_r$ 添加到 $Th\_H\text{-}Q_r$ 集合；
    else                      将 $0.9\times Th\_H$ 和 $Q_r$ 添加到 $Th\_H\text{-}Q_r$ 集合；
    if(percentage_2 > 0.90)   将 $1.1\times Th\_L$ 和 $Q_r$ 添加到 $Th\_L\text{-}Q_r$ 集合；
    else                      将 $0.9\times Th\_L$ 和 $Q_r$ 添加到 $Th\_L\text{-}Q_r$ 集合；
f:  go to b，转码下一帧；
```

为了能更新 $Th\_H\text{-}Q_r$ 集合中的参数 a 和 b，系统会记录当前宏块所对应的 *nz_per* 值处在阈值附近（大于 0.9×*Th_H* 且小于 *Th_H*）时所选择的类型，如果选择的类型中只有极少数为 P8x8 类型（percentage_1 < 0.10），则意味着当前阈值可能偏低，需要适当调高（将 1.1×*Th_H* 和 Q_r 添加到 $Th\_H\text{-}Q_r$ 集合）。另外，$Th\_L\text{-}Q_r$ 集合的更新方法与 $Th\_H\text{-}Q_r$ 集合类似。

3.4 帧内模式选择

利用 3.2 节所述的 $Th\_I\text{-}Q_r$ 模型，可以初步选择宏块类型。如果选择的宏块类型是 I16MB，在本文中以 SAD（Sum of Absolute Difference）为准则测试 4 种候选预测模式（见 2. 1. 1 帧内预测），并选择具有最小 SAD 的预测模式；如果选择的宏块类型是 I4MB，整个宏块将被分成 16 个 4x4 的子块，每个子块最多有 9 种候选预测模式（见图 2-2）。如果将这 9 种候选模式全部测试，势必导致大量的计算。本文中图像的缩放因子是 2，当前图像中每个 4x4 子块都对应着原始图像中 4 个 4x4 子块，这些 4x4 的块可能位于 I16MB 宏块内，也可能位于 I4MB 宏块内，见图 3-10。图中原始图像只有左上角一个宏块是 I16MB，其余的都是 I4MB，而在当前图像中，也只有{B00，B01，B10，B11}所对应的 4x4 块位于 I16MB 宏块内，下面对这两种情况分别讨论。

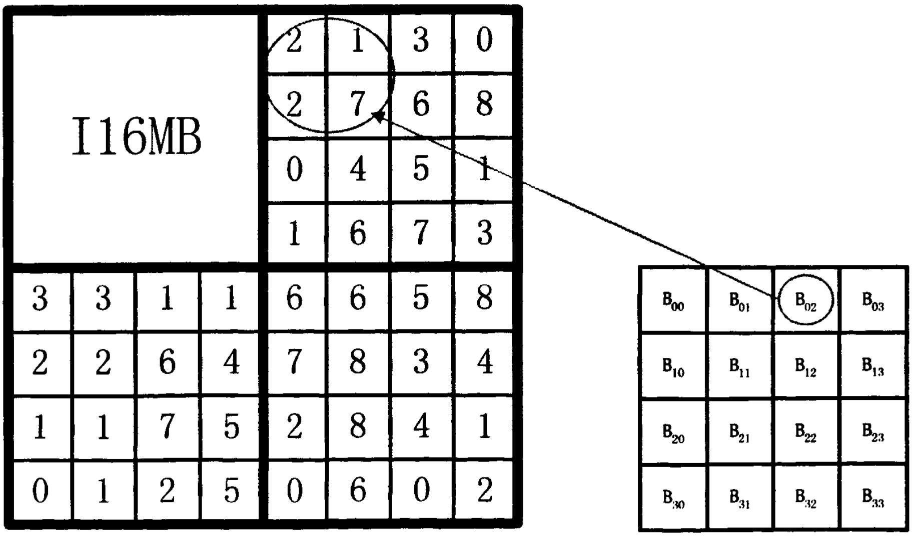


图 3-10 帧内预测模式的对应

情况 1：对应的 4 个 4x4 子块位于 I16MB 宏块内，如图 3-10 中{B00，B01， B10，B11}。图 3-11 给出这种情况下由全搜索法得出的各个预测模式的使用比例，图中的横坐标表示 9 种候选预测模式，纵坐标表示各预测模式的使用比例。从实验结果可以看出，模式 0、1、2 的使用概率（总和为 87.5%）要远远超过其他 6 种模式。因此在情况 1 下，就可以省略其他 6 种模式，直接从模式 0、1、2 中选择具有最小 SAD 的模式。

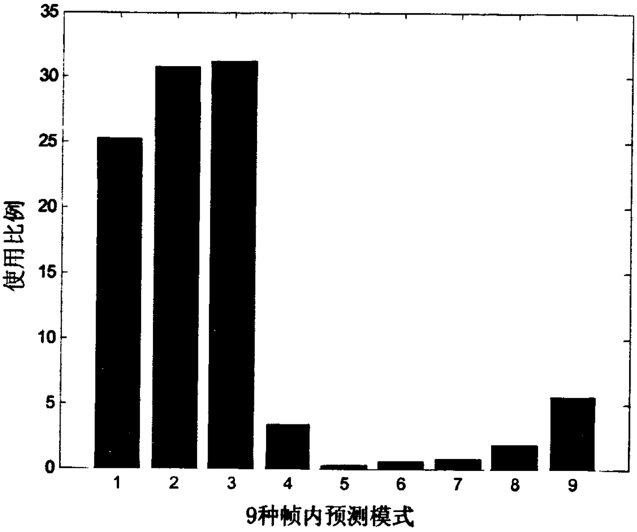


图 3-11 在情况 1 下各个预测模式的使用比例

情况 2-1：对应的 4 个 4x4 子块位于 I4MB 宏块内，如图 3-10 中{B02，B03，B12，B13}。在这种情况下，原始图像中的每一个 4x4 的子块都有各自的预测模式，比如，块 B02 对应的原始图像中的 4 个预测模式分别为{2, 1, 2, 7}。在下面的实验中将统计由全搜索法得出的 B02 使用这 4 个预测模式之一的概率。图中左边的型条表示 B02 的最优预测模式是这 4 个预测模式之一的概率，右边型条表示了 B02 的最优或次优预测模式是这 4 个预测模式之一的概率，也就是说，B02 使用这 4 个预测模式之一的概率很高。因此本文只考虑这 4 种预测模式作为候选模式，这样至少为每个 4x4 的子块省略 5 种候选模式，从而大幅度节省计算量。

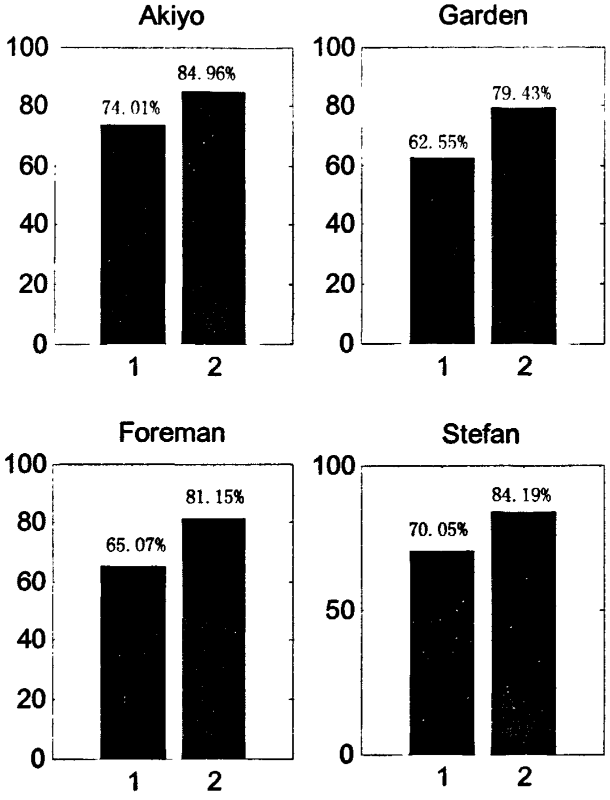


图 3-12 降低图象尺寸前后帧内预测模式的关系

情况2-2: 在图 3-13 的实验中，横坐标为重新量化参数 Q_r。纵坐标是选择 9 种预测模式的概率（由全搜索法得出），针对每个 Q_r 都有 9 个型条来表示每个预测模式的概率。通过这个实验可以看出，随着 Q_r 的递增，预测模式 0、1、2 的使用概率会越来越高（Q_r=20: 60.04%, Q_r=30: 65.16%, Q_r=40: 74.57%）。也就是说，在情况 2下，随着 Q_r 的增大，需要将 0、1、2 三种模式作为另外的候选模式。本文为 Q_r 设置了一个硬阈值，当 Q_r 大于该阈值时，在原有四个候选模式上增加 0、1、2 作为候选模式，并从中选择具有最小 SAD 的模式。

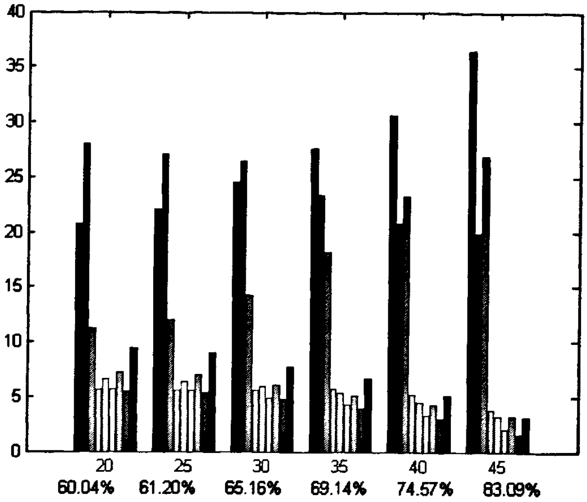


图 3-13 Q_r 与模式 0、1、2 的关系

3.5 帧间模式选择

3.5.1 宏块类型选择的过程

本文提出的整个帧间宏块类型选择的过程见图 3-14。

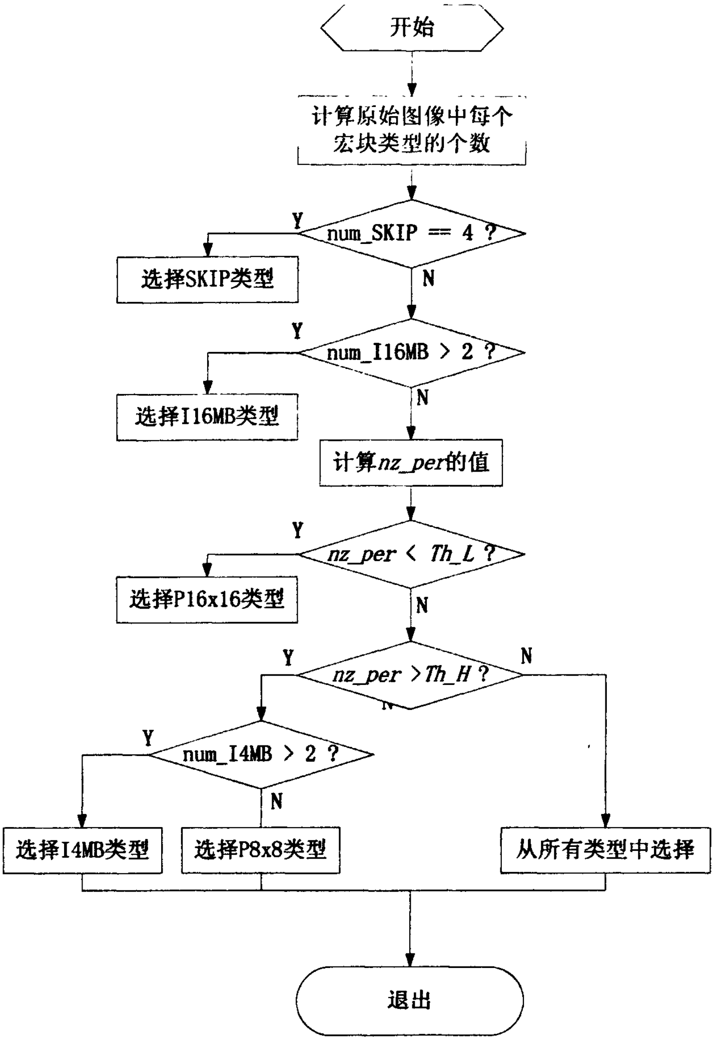


图 3-14 帧间模式选择流程

为了加速宏块类型选择的过程，本文对一些特殊类型宏块进行了单独处理。如果 4 个对应宏块全为 SKIP 类型，则当前宏块直接选择为 SKIP 类型。如果 4 个对应宏块中有 3 个以上是 I16MB 类型，则当前宏块选择为 I16MB。

如果 *nz_per* 值小于 *Th_L*，则直接选择为 P16x16 类型。在编码 P16x16 时，如果没有任何残差数据，并且当前运动矢量与预测值相同，则调整为 SKIP 类型。如果 *nz_per* 值大于 *Th_H*，则统计对应四个宏块类型，如果超过半数为 I4MB，则选择为 I4MB，否则使用 P8x8。如果 *nz_per* 处在两个阈值之间，则需要从所有类型中选择，选择的准则见 3. 5. 2 宏块类型选择的准则，另外还需要统计相关信息来更新 $Th\_H\text{-}Q_r$ 模型和 $Th\_L\text{-}Q_r$ 模型，具体方法见 3. 3. 1 参数 a 和 b 的更新。

3.5.2 宏块类型选择的准则

在本文中，宏块类型选择的准则是计算下式的最小值：

$$COST = SAD + \lambda MV_{bits} \tag{3.12}$$

其中：

SAD 是预测值和参考值间的差值的绝对值的和(Sum of Absolute Difference)。预测值根据初始运动矢量计算得到。初始运动矢量是根据原始图像中的运动矢量（通过解码可以得到）计算得到，具体计算方法见 3. 5. 3 初始运动矢量的计算。

λ 是与量化参数相关的拉格朗日算子。

MV_{bits} 是运动矢量所消耗的比特长度，这里只对运动矢量的残差进行编码。

在宏块类型选择中，所有候选宏块类型都以等式(3.12)为准则来比较，并选择其中最小值作为最终的宏块类型。

3.5.3 初始运动矢量的计算

本文使用的图像缩放因子为 2，H.264 中运动估计所用的最小子块为 4x4。若以 4x4 块为单位，则当前图像中一个宏块对应着原始图像中的 64 个运动矢量，见图 3-15。

mv_{00}	mv_{01}	mv_{02}	mv_{03}	mv_{04}	mv_{05}	mv_{06}	mv_{07}
mv_{10}	mv_{11}	mv_{12}	mv_{13}	mv_{14}	mv_{15}	mv_{16}	mv_{17}
mv_{20}	mv_{21}	mv_{22}	mv_{23}	mv_{24}	mv_{25}	mv_{26}	mv_{27}
mv_{30}	mv_{31}	mv_{32}	mv_{33}	mv_{34}	mv_{35}	mv_{36}	mv_{37}
mv_{40}	mv_{41}	mv_{42}	mv_{43}	mv_{44}	mv_{45}	mv_{46}	mv_{47}
mv_{50}	mv_{51}	mv_{52}	mv_{53}	mv_{54}	mv_{55}	mv_{56}	mv_{57}
mv_{60}	mv_{61}	mv_{62}	mv_{63}	mv_{64}	mv_{65}	mv_{66}	mv_{67}
mv_{70}	mv_{71}	mv_{72}	mv_{73}	mv_{74}	mv_{75}	mv_{76}	mv_{77}

图 3-15 原始图像中的运动矢量

当前宏块中每个类型中的初始运动矢量都由这 64 个运动矢量计算得出，具体计算方法见表 3-1。表中所列的方法中使用中间值方法为 P8x8，P8x16，P16x8，P16x16 计算初始运动矢量，其中计算 P8x8，P8x16，P16x8 的输入数据为原始图

像中的运动矢量，为了节省计算量，计算 P16x16 时使用了 P8x8 的初始运动矢量作为输入数据。

表 3-1 初始运动矢量的计算

宏块类型	初始运动矢量的计算
P8x8 $\{mv88_{i,j},\ i,j{=}0,1\}$	$mv88_{i,j}=\text{median}\{mv_{4\times i,4\times j}, mv_{4\times i+1,4\times j},\ldots,mv_{4\times i+3,4\times j+3}\}$
P8x16 $\{mv816_{j},\ j{=}0,1\}$	$mv816_{j}=\text{median}\{mv_{0,4\times j}, mv_{1,4\times j},\ldots,mv_{7,4\times j+3}\}$
P16x8 $\{mv168_{i},\ i{=}0,1\}$	$mv168_{i}=\text{median}\{mv_{4\times i,0}, mv_{4\times i,1},\ldots,mv_{4\times i+3,7}\}$
P16x16 $\{mv1616\}$	$mv1616=\text{median}\{mv88_{i,j}\ \ i,j{=}0,1\}$

3.5.4 运动矢量的细化

仅仅由原始图像计算出来的初始运动矢量并非一定精确，尤其是当 Q_r 较大时。因此，本文提出的方法中，选择了最终的宏块类型以后，需要对运动矢量进行细化。由前面的讨论得知，*nz_per* 值较大的区域意味着运动较为剧烈，因此需要较长的运动矢量细化步长。本文提出以 *nz_per* 作为运动矢量细化步长的准则，且随着 Q_r 的增加，运动矢量细化步长也逐步增加。通过对大量序列的统计结果，表 3-2 所列为最大的细化步长与 Q_r 的关系。

表 3-2 最大细化步长和 Q_r 的关系

Q_r	最大步长
$Q_r <= 10$	1
$10 < Q_r <= 20$	2
$20 < Q_r <= 40$	3
$40 < Q_r$	4

在本文提出的方法中，运动矢量的细化步长根据 *nz_per* 值自适应的改变，具体细化步长计算方法见等式(3.13)：

$$step=\min(\text{SR\_TAB}[Q_r],(\text{SR\_TAB}[Q_r]\cdot nz\_per/Th\_H))\qquad(3.13)$$

其中变量 SR_TAB 的数据即为表 3-2。根据本文的方法，如果 *nz_per* 的值大于 *Th_H* 时，就会采用由表 3-2 定义的最大细化步长，其余情况下，细化步长都随着 *nz_per* 的值自适应的变化，从而保证了在残差数据较大时（*nz_per* 较大，运动较为剧烈），使用较长的运动矢量细化步长；而对于运动较为平缓的区域，使用较短的细化步长即可，进而节省计算量。

3.6 实验结果

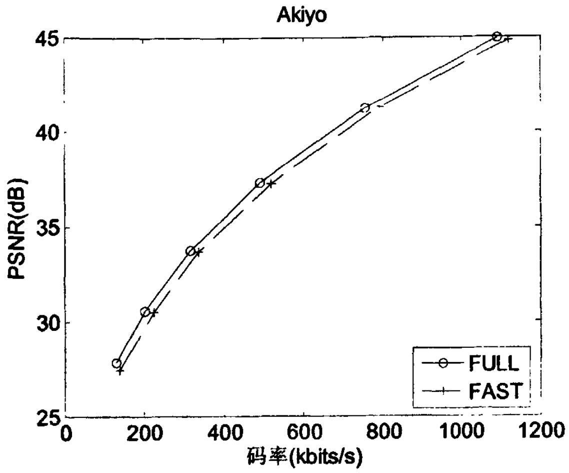
本文实验的运行环境为 Intel Pentium-IV 2.66GHz，512M 内存，Microsoft windows 操作系统。在下面的实验中，输入的原始图像尺寸为 CIF（352×288），帧率为 30 帧/秒，输入的量化参数为 20。降低图像尺寸的比例因子为 2，即降低尺寸后图像大小为 QCIF（176×144）。降低尺寸后图像中的像素值是原始图像中对应的 4 个像素的平均值。输入的原始图像由最常用的 H.264 参考软件 JM12.1 编码，部分编码参数见表 3-3。符号“FULL”表示使用全搜索法重新进行宏块类型选择得到的结果。“FAST”代表本文提出的方法。整个转码系统所消耗的时间包括解码时间、降低图像尺寸时间和重新编码时间三部分，由于本文讨论的内容集中在重新编码部分，这里没有给出前两部分时间消耗。“重新编码时间”表示整个重新编码过程总的耗时；由于本文的方法与全搜索法的差别是选择宏块类型，因此本文单独比较了两者在选择宏块类型上的耗时，用“选择宏块类型时间”来表示。

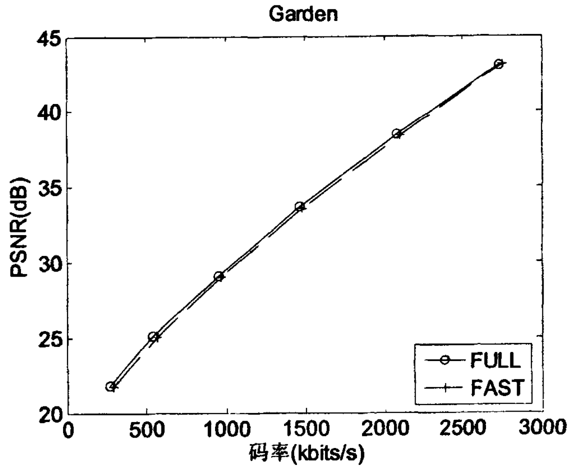
表 3-3 参考软件 JM 部分编码参数

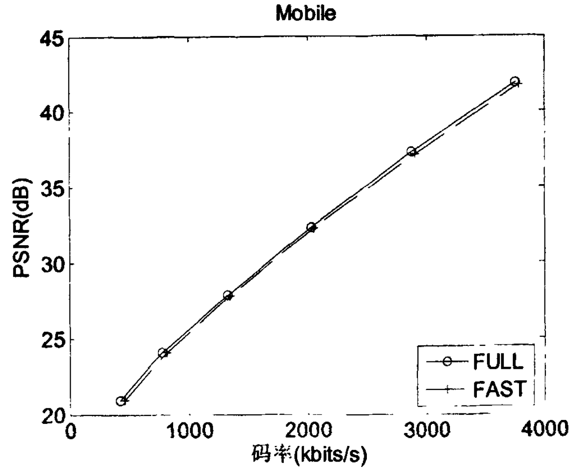
条件	结果
帧率	30
ProfileIDC	Baseline profile
运动预测	全搜索法
参考帧数目	1
码率控制	OFF
搜索步长	16
量化参数	20

3.6.1 帧内模式选择实验

在帧内模式选择实验中，所有帧都使用 I 帧进行编码。实验中输入的初始 $Th\_I\text{-}Q_r$ 集合是 Q_r =[20, 25, 30, 35, 40, 45]，Th_I = [1.07, 2.25, 4.10, 7.52, 12.99, 33.69]。本章 3.3 节中提到的硬阈值设为 30。图 3-16 给出了部分视频序列的实验结果，图中比较了全搜索法和本文方法的压缩性能。从实验结果可以看出，两种方法的压缩性能差别不大，比如在序列‘Akiyo’中最大 PSNR 损失约 0.6dB，而在其它序列中基本上无损失。


Akiyo
PSNR(dB)
45
40
35
30
25
0
200
400
600
800
1000
1200
码率(kbits/s)
FULL
FAST


Garden
PSNR(dB)
45
40
35
30
25
20
0
500
1000
1500
2000
2500
3000
码率(kbits/s)
FULL
FAST


Mobile
PSNR(dB)
45
40
35
30
25
20
0
1000
2000
3000
4000
码率(kbits/s)
FULL
FAST

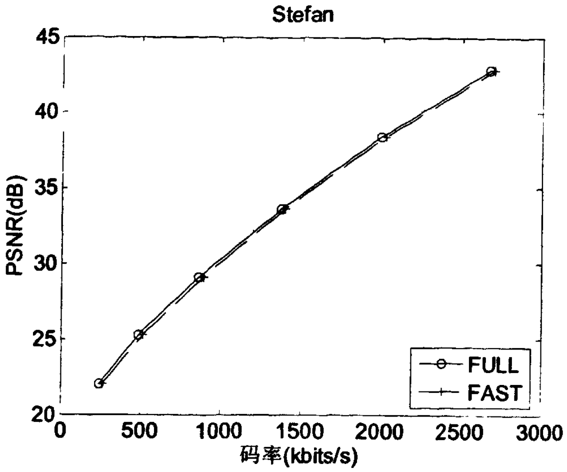


图 3-16 视频序列率失真比较

表 3-4 给出了全搜索法和本文提出的方法的消耗时间比较。根据表中的实验结果，本文提出的方法总的编码时间是全搜索法的 70%左右；如果仅考虑选择宏块类型部分的耗时，本文方法的耗时仅为全搜索法的 20%-25%。

表 3-4 "FAST" 和"FULL"耗时比较

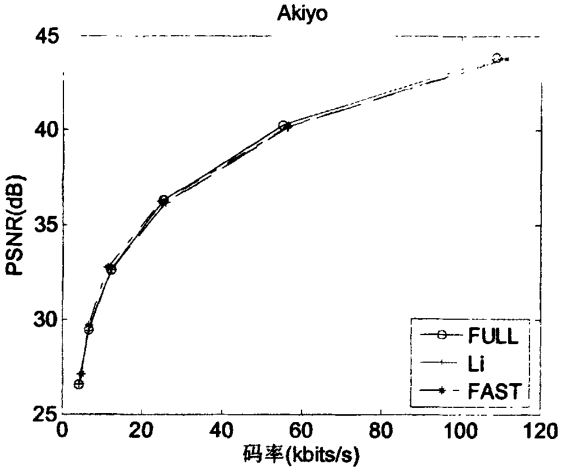
序列	总编码时间 (秒)		选择宏块类型时间 (秒)	
	FAST	FULL	FAST	FULL
Akiyo	3.218	4.690	0.478	2.344
Garden	3.557	4.898	0.469	1.989
Mobile	5.105	6.630	0.650	2.520
Stefan	4.210	5.833	0.578	2.377

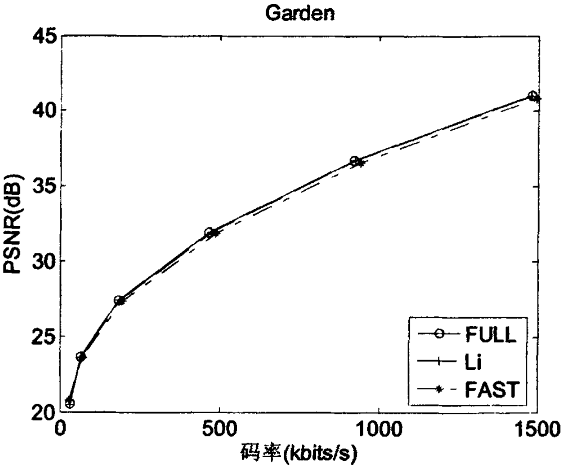
3.6.2 帧间模式选择实验

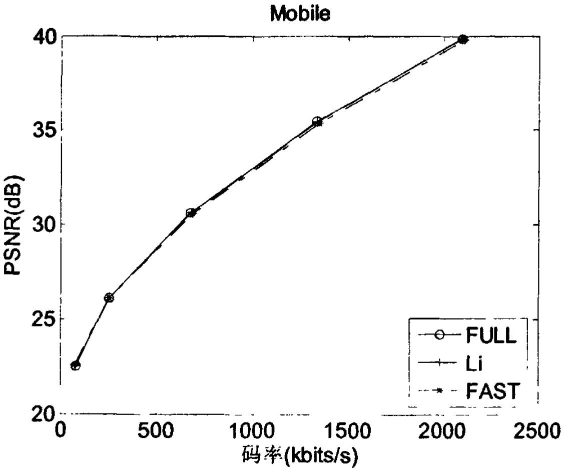
在帧间模式选择实验中，只有第一帧使用 I 帧，其余全部使用 P 帧编码。实验中输入的初始 *Th_H-Q*$_r$集合为 Q_r=[20, 25, 30, 35, 40, 45]，*Th_H* = [19.31, 22.36, 26.39, 36.24, 50.70, 73.24]；初始 *Th_L-Q*$_r$ 集合为 Q_r=[20, 25, 30, 35, 40, 45]，*Th_L* = [0.19, 0.49, 1.07, 2.15, 5.18, 14.55]。

在帧间模式选择实验中，全搜索法的运动预测的搜索步长为 16。另外，本文还实现了文献[4]提出的方法（称之为 Li）。图 3-17 给出的实验结果中比较了全搜索法，Li 的方法和本文的方法的压缩性能。本文的方法和 Li 的方法压缩性能差不多，在运动剧烈的序列中与全搜索法相比损失较多，如 Stefan，相差的

PSNR 最大值约为 1.0dB。在其它运动性质的序列中，两者都与全搜索法差别不大。







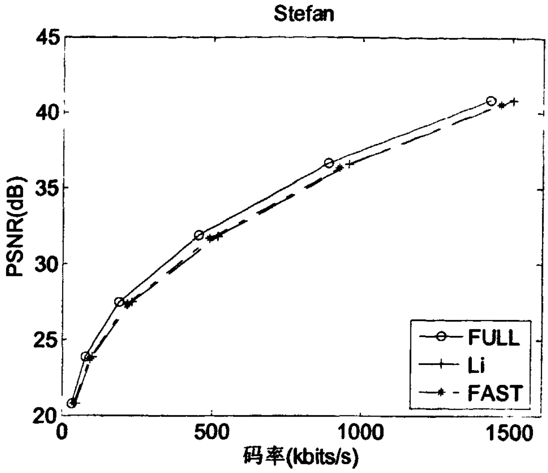


图 3-17 多个序列的率失真比较

表 3-5 给出了三种方法平均消耗时间的对比。相对于全搜索法，Li 的方法可以将总的编码时间平均可以提高约 5.5 倍，本文的方法平均可以提高约 15 倍；只考虑在选择宏块类型部分的耗时，相对于全搜索法，Li 的方法平均能将速度提高 6 到 7 倍，而本文提出的方法可以提高 35 倍左右，从而极大的节省了选择宏块类型的时间。

表 3-5 三种方法的平均耗时比较

序列	重新编码时间 (秒)			选择宏块类型时间 (秒)		
	FAST	Li	FULL	FAST	Li	FULL
Akiyo	2.19	6.48	37.05	0.65	4.54	35.29
Garden	4.09	9.58	58.37	1.86	7.13	56.18
Mobile	5.64	11.78	83.48	2.56	8.45	80.33
Stefan	5.13	20.48	75.45	2.22	17.51	72.86

3.7 本章小结

在基于 H.264 的空间分辨率转码系统中，需要为降低尺寸后图像中的宏块重新选择类型。本文提出了一种快速的宏块类型选择方法。该方法适用的图像尺寸缩放比例因子为 2。即一个降低尺寸后图像中的宏块对应了 4 个原始图像中的宏块。本文提出的方法包含了帧内模式选择和帧间模式选择两部分。在帧内模式选择部分，本文提出的方法利用原始图像中四个宏块的非零系数比例（*nz_per*）为准则选择宏块类型。在帧间模式选择部分，利用 *nz_per* 为准则划分当前宏块所

在区域的运动性质，根据不同的运动性质，跳过部分候选宏块类型的测试，从而节省计算时间。最后，针对帧间宏块类型，本文提出了一个自适应的运动矢量细化方法，该方法可以根据残差数据自动调整运动矢量细化步长。在使用 *nz_per* 作为判断准则时，本文使用指数模型来描述 *nz_per* 阈值与重新量化参数的关系（Th_Q_r 模型），并将其线性化为一元线性回归模型，进一步使用最小二乘法来估计数学模型中的参数。最后的实验结果表明，与全搜索法相比，在压缩性能相差不大的情况下，本文提出的方法可以大幅度的节省宏块类型选择的计算量，从而提高重新编码的速度。

第 4 章 时间分辨率转码

4.1 引言

在时间分辨率转码中，由于部分帧需要被舍弃，如果某帧以丢弃帧作为参考帧，则它的运动矢量将不复存在，因此需要对运动矢量作重新估计，常用方法有 FDVS[63]，ADVS[13]，加权中间值[10]，加权平均值[10]等。另外，由于部分帧被丢弃，导致了原有序列帧间变化加大，从而运动加剧，因此需要为保留帧中的宏块重新选择类型。本章讨论的时间分辨率转码不包含空间分辨率转码部分，包含了比特率转码部分。

从第 3 章的分析得知，当前宏块类型的选择与输入原始图像中的残差数据关系密切，并提出了 Th_Q_r 模型来估计残差阈值与重新量化参数 Q_r 的关系，本章将进一步讨论该模型，并将其推广到时间分辨率转码方面。

请参考图 1-3 给出的时间分辨率转码中的示范图例，F_n 为当前帧，$F_{n\text{-}1}$ 被丢弃，则 F_n 中指向 $F_{n\text{-}1}$ 中的运动矢量就要重新计算，并指向 $F_{n\text{-}2}$。常用的计算方法有 FDVS[63]，ADVS[13]等。本文统计当前宏块运动路径上（图 1-3 中 $F_{n\text{-}1}$ 及 F_n 的阴影部分）非零系数的个数比例，为与空间分辨率转码统一，也称之为"*nz_per*"。从直观上讲，*nz_per* 越大，意味着在丢弃的帧中，当前宏块所经过区域的运动较为剧烈，因此当前宏块选择 P8x8 的概率会更高。

下面通过一个实验来说明上述直观结果。在图 4-1 的实验结果中统计了序列'Stefan'中当前宏块的 *nz_per* 值与宏块类型的关系，所选择的类型都是经过全搜索法得出。从实验结果很明显可以看出，当 *nz_per* 很小时，选择 P16x16 的概率很高；随着 *nz_per* 的增大，选择 P8x8 的概率逐步增加。这里与空间分辨率转码部分非常类似。因此可以参照空间分辨率转码部分，为 *nz_per* 设置两个阈值，分别为 *Th_L* 和 *Th_H*。如果统计出的 *nz_per* 小于 *Th_L*，则当前宏块直接选择为 P16x16（SKIP）模式；如果 *nz_per* 大于 *Th_H*，则当前宏块直接选择 P8x8 类型；如果在两者之间，则从所有宏块类型中选择。

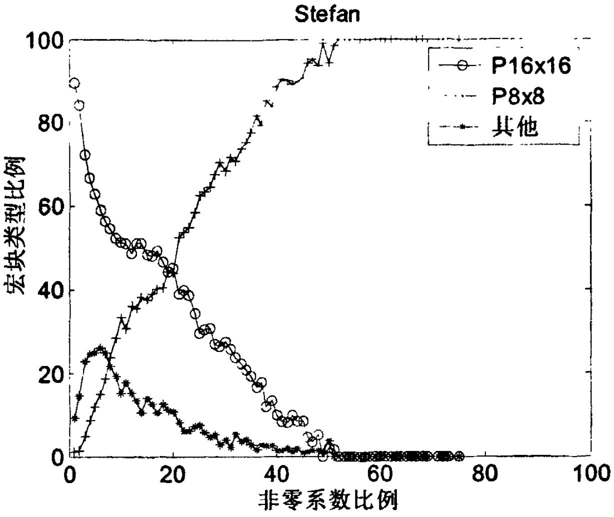


图 4-1 *nz_per* 与宏块类型的关系

4.2 *Th_Q$_r$_T* 模型

本章在时间分辨率转码过程中也考虑了重新量化的问题，由于宏块类型的选择与量化参数有关，阈值 *Th_L* 和 *Th_H* 必然也与 Q_r 有关。本节提出的 *Th_Q$_r$_T* 模型也分为 *Th_L_Q$_r$_T* 集合和 *Th_H_Q$_r$_T* 集合两部分，它们分别用于计算阈值 *Th_L* 和 *Th_H*。

图 4-2 和图 4-3 以序列 ‘Foreman’ 为例给出了一个实验结果，实验中为每个 Q_r 选择固定阈值 *Th_L* 和 *Th_H*，如果当前宏块对应的 *nz_per* 小于 *Th_L*，就使用 P16x16 宏块类型；如果 *nz_per* 大于 *Th_H*，则使用 P8x8 类型；否则使用全搜索法从所有宏块类型中选择最优类型。在保证压缩性能与全搜索法几乎相同的前提下，图 4-2 给出了 *Th_L* 与 Q_r 的关系，图 4-3 为 *Th_H* 与 Q_r 的关系。

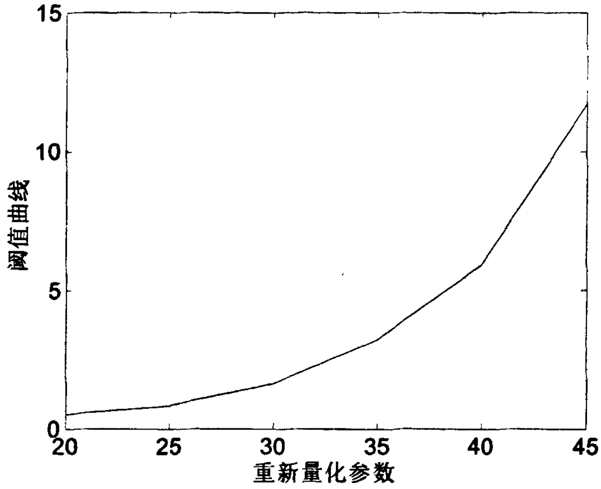


图 4-2 Q_r 与 Th_L 的关系

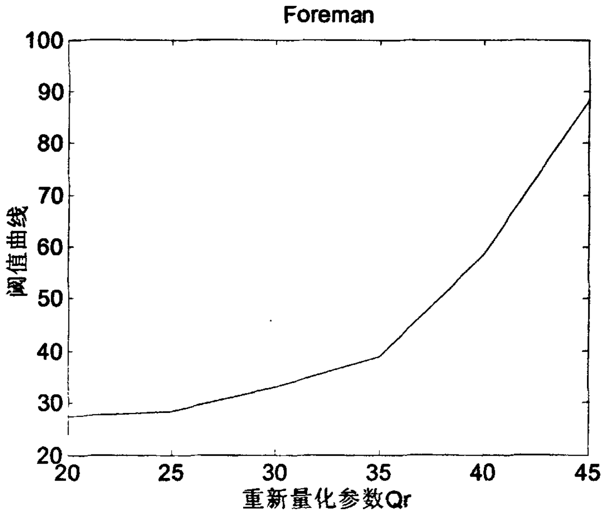


图 4-3 Q_r 与 Th_H 的关系

从这个实验结果可以看出，在保证了压缩性能不受损失的前提下，Q_r 与 Th_L 和 Th_H 的关系都很接近于一个指数曲线，该性质与空间分辨率转码中类似。因此，本章也用指数曲线来描述 Q_r 与 Th_L、Th_H 的关系，并称之为 $Th\_Q_r\_T$ 模型。数学模型见等式(4.1)和(4.2)，其中 a 和 b 是指数曲线的参数。

$$Th\_L = ae^{bQ_r} \tag{4.1}$$

$$Th\_H = ae^{bQ_r} \tag{4.2}$$

由于本章使用的 $Th\_Q_r\_T$ 模型与上章的 Th_Q_r 模型形式一致，因此参数的估计和更新均非常类似，故这里不再赘述。

4.3 宏块类型选择

本章提出的整个帧间宏块类型选择的过程见图 4-4。

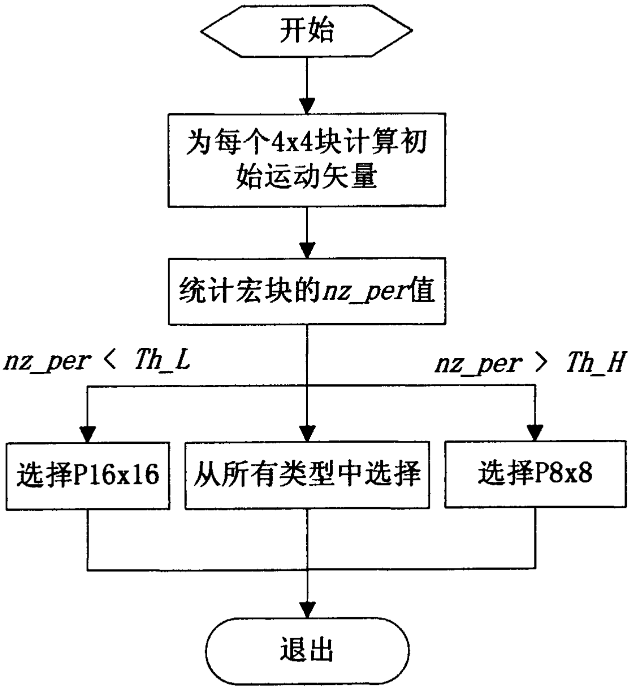


图 4-4 帧间模式选择流程

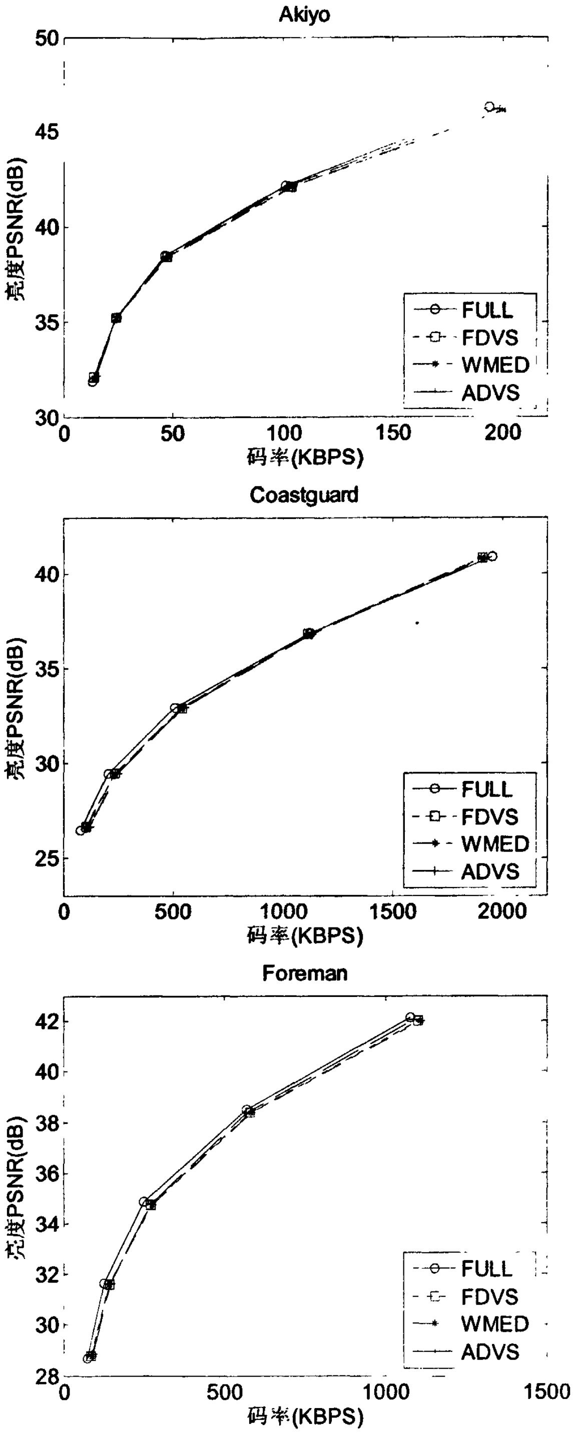
在选择当前宏块的类型前，需要统计其对应的 *nz_per* 值。如果 *nz_pe* 值小于 *Th_L*，则直接选择为 P16x16 类型。在编码 P16x16 时，如果没有任何残差数据，并且当前运动矢量与预测值相同，则调整为 SKIP 类型。宏块类型选择的准则见 3.5.2 部分。运动矢量细化方案也使用上章 3.5.4 提出的方法。

由于时间分辨率转码中牵涉到帧的丢弃，而计算 *nz_per* 时通过的区域并非一定宏块的边界（如图 1-3），也很难统计当前宏块所对应的丢弃帧中具体宏块类型，因此本文在时间分辨率转码中没有包括帧内预测模式（I4MB/I16MB），在下一步的研究中需要讨论如何将 I4MB/I16MB 类型引入到 *Th_Qr_T* 模型。

4.4 实验结果

在下面的实验中，输入的图像尺寸为 CIF（352×288），帧率为 30 帧/秒，输入的量化参数为 20。本文测试了 2，3，4 三种跳帧的比例因子，因此转码后输出的帧率分别为 15 帧/秒，10 帧/秒，7.5 帧/秒。输入的图像由最常用的 H.264 参考软件 JM12.1 编码，编码参数见表 3-3。符号“FULL”表示使用全搜索法重新进行宏块类型选择得到的结果，运动预测的搜索步长为 16。本文提出的方法测试了三种计算初始运动矢量的方法，分别是 FDVS，ADVS，和加权中间值（WMED）。整个转码系统所消耗的时间包括解码和重新编码时间两部分，由于本文讨论的内容集中在重新编码部分，这里没有给出解码部分时间消耗。“重新编码时间”表示整个重新编码过程总的耗时；由于本文的方法与全搜索法的差别是选择宏块类型，因此本文单独比较了两者在选择宏块类型上的耗时，用“选择宏块类型时间”表示。

在实验中只有第一帧使用 I 帧，其余全部使用 P 帧编码，在 P 帧中不使用 I16MB 和 I4MB 等帧内预测模式。实验中输入的初始 $Th\_L\text{-}Q_r$ 集合为 Q_r=[20, 25, 30, 35, 40, 45]，Th_L = [0.5, 0.8, 1.6, 3.2, 5.9, 11.7]；初始 $Th\_H\text{-}Q_r$ 集合为 Q_r=[20, 25, 30, 35, 40, 45]，Th_H = [27.5, 29.3, 33.2, 39.1, 58.6, 88.2]。图 4-5~图 4-7 给出了不同输出帧率下各个序列的实验结果。从结果可以看出，输出帧率较高时（如 15 帧/秒），本文提出的方法的压缩性能与全搜索法差别不大，在相同的比特率下，PSNR 相差最大不超过 0.5dB。随着输出帧率的降低，本文的方法的效率也有所下降，比如在帧率为 7.5 帧/秒时，相同比特率下，PSNR 最大相差约为 1.1dB。同时，本文测试的三种计算初始运动矢量的方法的压缩性能几乎没有差别。从实验结果还可以看出，对于运动平缓的序列（如 Akiyo），帧率的改变对本文提出的方法影响不大，并且与全搜索法的差别很小；而运动较为剧烈的序列（如 Stefan），随着帧率的降低，本文的方法的效率下降较快。


Akiyo
50
45
40
35
30
亮度PSNR(dB)
0
50
100
150
200
码率(KBPS)
FULL
FDVS
WMED
ADVS
Coastguard
40
35
30
25
亮度PSNR(dB)
0
500
1000
1500
2000
码率(KBPS)
FULL
FDVS
WMED
ADVS
Foreman
42
40
38
36
34
32
30
28
亮度PSNR(dB)
0
500
1000
1500
码率(KBPS)
FULL
FDVS
WMED
ADVS

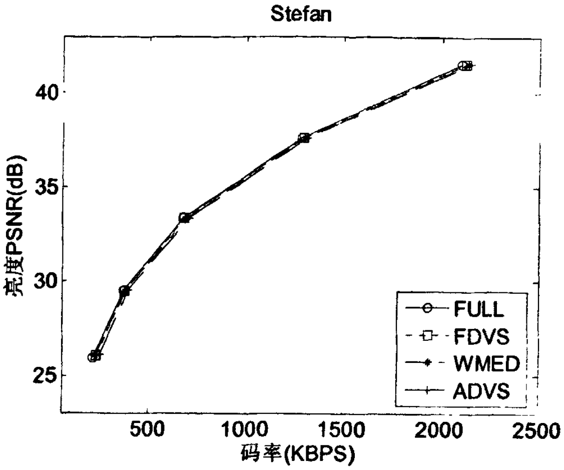
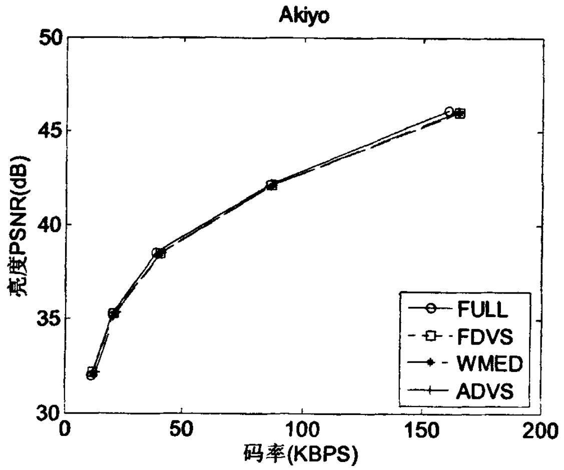
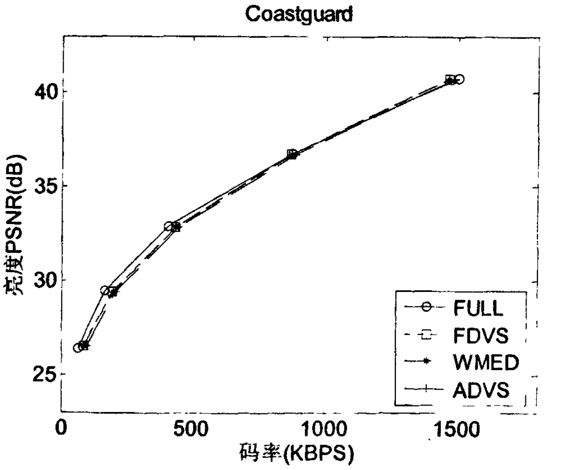
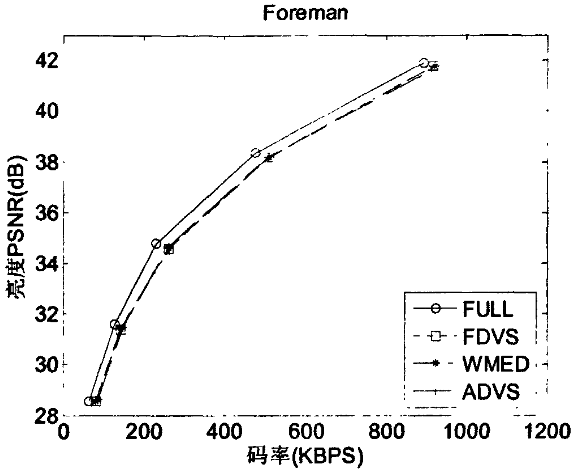

Stefan
亮度PSNR(dB)
40
35
30
25
500
1000
1500
2000
2500
码率(KBPS)
FULL
FDVS
WMED
ADVS

图 4-5 帧率：15 帧/秒


Akiyo
亮度PSNR(dB)
50
45
40
35
30
0
50
100
150
200
码率(KBPS)
FULL
FDVS
WMED
ADVS


Coastguard
亮度PSNR(dB)
40
35
30
25
0
500
1000
1500
码率(KBPS)
FULL
FDVS
WMED
ADVS


Foreman
42
40
38
36
34
32
30
28
0
200
400
600
800
1000
1200
码率(KBPS)
亮度PSNR(dB)
FULL
FDVS
WMED
ADVS

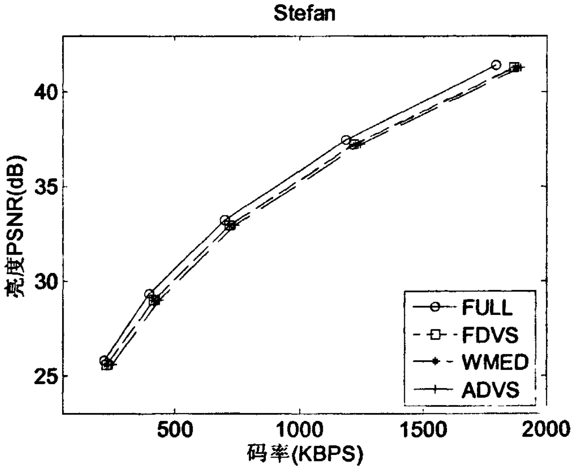
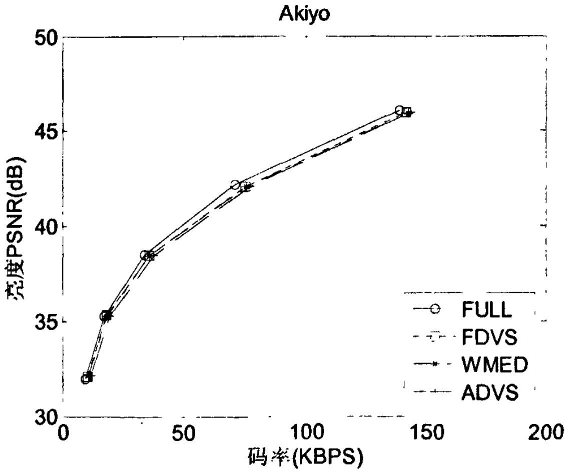

Stefan
40
35
30
25
500
1000
1500
2000
码率(KBPS)
亮度PSNR(dB)
FULL
FDVS
WMED
ADVS

图 4-6 帧率：10 帧/秒


Akiyo
50
45
40
35
30
0
50
100
150
200
码率(KBPS)
亮度PSNR(dB)
FULL
FDVS
WMED
ADVS

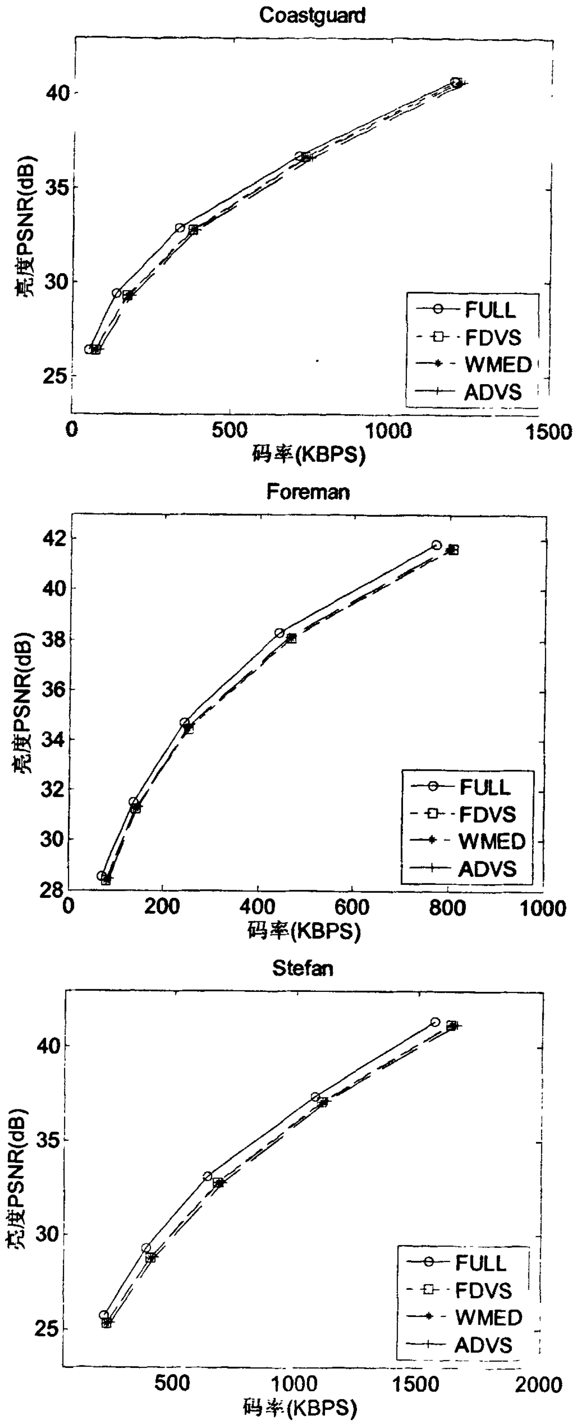


图 4-7 帧率：7.5 帧/秒

表 4-1 给出了几种方法运算复杂度的对比。表中所列数据为相对于全搜索法

所能提高的倍数，根据表 4-1，本文的方法的总体编码速度可以提高约 15 倍，最高可达 20 倍；若只考虑在选择宏块类型时间，则可以提高 35 倍左右，从而极大的节省了选择宏块类型的时间。另外，根据表中的结果，利用 FDVS 方法计算初始运动矢量最为省时，而加权中间值最为耗时。根据图 4-5~图 4-7 的结果，三种方法的压缩性能几乎没有差别，综合而论，FDVS 方法最为有效。

表 4-1 运算复杂度比较

序列	帧率（帧/秒）	重新编码时间比较（倍）			选择宏块类型时间（倍）		
		FDVS	ADVS	WMED	FDVS	ADVS	WMED
Akiyo	15	11.5	11.5	10.0	25.4	24.2	23.7
	10	11.4	10.9	10.4	24.0	22.5	20.6
	7.5	11.2	10.1	10.0	24.3	21.7	18.3
Coastguard	15	20.1	19.1	19.0	41.0	37.0	35.5
	10	18.7	19.5	19.0	37.4	36.4	35.3
	7.5	19.3	19.4	18.7	39.3	36.7	34.6
Foreman	15	17.3	17.0	16.7	33.3	32.5	31.6
	10	18.5	17.5	17.3	36.4	32.4	31.1
	7.5	19.4	18.9	17.9	37.3	34.3	31.8
Stefan	15	19.3	18.6	18.5	39.5	38.3	38.4
	10	18.6	18.7	17.5	38.6	37.1	36.1
	7.5	18.7	17.7	17.0	38.2	36.5	33.5

4.6 本章小结

在基于 H.264 的帧率转码系统中，需要为改变帧率后的图像中的宏块重新选择类型。本文提出了一种快速的宏块类型选择方法。本章提出的方法利用当前宏块的运动路径上非零系数个数比例（本文称为 *nz_per*）为准则划分当前宏块所在区域的运动性质，根据不同的运动性质，跳过部分候选宏块类型的测试，从而节省计算时间。本章借鉴了空间分辨率转码部分，将使用的阈值和重新量化参数模型化为一个指数曲线，并将其线性化为一元线性回归模型，最后使用最小二乘法对模型中的参数进行估计，在编码过程中对参数即时更新。最后的实验结果表明，与全搜索法相比，在压缩性能相差不大的情况下，本章提出的方法可以大幅度的提高宏块类型选择的速度，从而节省重新编码的运算时间。

第 5 章 时间-空间分辨率转码

5.1 引言

在本章中，帧内候选宏块类型包括：mode_I = {I4MB, I16MB}，帧间候选宏块类型包括：mode_P = {SKIP, P16x16, P16x8, P8x16, P8x8 }。

基于 H.264 的视频编码和视频转码均需要选择宏块类型，两者最大的不同点在于在视频转码中，可以利用丰富的解码信息来快速选择宏块类型，从而节省计算量。在转码中可以提取的信息包括：原始图像中宏块类型、残差数据、运动矢量、量化参数等，而宏块类型选择的目的就是从候选宏块类型集合（mode_I 或 mode_P）中选择其一。如果从解码信息中可以提取出部分特征向量，而目标宏块类型在很大程度上取决于这些特征向量，则就可以利用分类的方法来选择宏块类型。

5.2 支持向量机 Support vector machine (SVM)

支持向量机是一种非常常见的分类方法，在视频方面且已经成功地应用到了很多领域，如对象提取[80]、镜头边界检测[81]、车辆检测[82]等等。

给定特征向量 $x_i, i=1,...,N$,和目标类别 $y_i \in \{1,-1\}$ ，支持向量机的主要任务就是解决下式的优化问题：

$$\begin{aligned} \max_{\alpha_i} \quad & \sum_{\mathrm{i}=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j), \\ s.t. \quad & \sum_{\mathrm{i}=1}^{N}\alpha_i y_i = 0,\ 0 \le \alpha_{\mathrm{i}} \le C, \quad i=1,...,N. \end{aligned} \tag{5.1}$$

在分类阶段的判别函数为：

$$y = \mathrm{sign}(\sum_{i=1}^{N_s}\alpha_i K(x,x_i)+b) \tag{5.2}$$

其中 C 是惩罚因子，$K(x_i,x_j)$ 是核函数，它将特征向量影射到一个高维空间，N 是训练的特征向量的个数，N_s 是训练后支持向量机模型中支持向量的个数，x 是输入的待分类的特征向量，y 是分类结果。

在视频转码中，输入的特征向量x_i需要从解码信息中提取，而目标类别y_i就是需要确定的宏块类型，在训练阶段的目标类型可以通过全搜索法得到。

5.3 特征向量的选择

在视频转码中解码得到的信息包括：原始图像宏块类型，残差数据，运动矢量，量化参数。下面给出特征向量的概念公式：

$$F\_V = \{MV, ERROR, MODES, QP\} \tag{5.3}$$

其中MV代表运动矢量，$ERROR$表示残差数据，$MODES$表示原始图像中的宏块类型，QP表示量化参数。下面对每个特征逐一讨论。

5.3.1 帧内模式选择中的特征选择

时间分辨率转码中不存在帧内模式选择的问题，下面讨论空间分辨率转码中帧内模式选择问题。

本文中采用的图像缩放因子为2，因此当前宏块对应着原始图像中4个宏块，如图 5-1 所示，每个宏块都有各自的宏块类型。为了能尽量降低特征向量的维数，本文选择这4个宏块类型的和作为一个特征，该特征表示式(5.3)中的$MODES$。

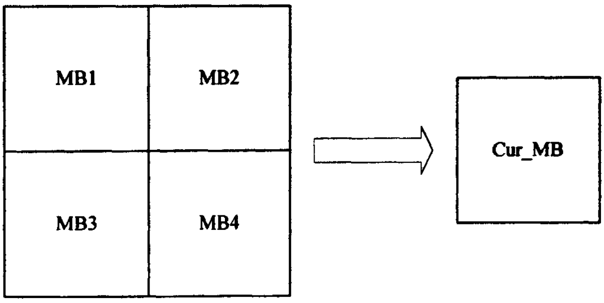


图 5-1 空间分辨率转码中的宏块

H.264 中采用的是 4x4 的整数变换，每个 4x4 子块都有各自的非零系数，该数据描述了当前 4x4 子块的性质（细节情况或运动性质），本文选择原始图像中4个宏块的非零系数的比例作为一个特征，该特征表示式(5.3)中的$ERROR$。

帧内模式选择中的量化参数特征的选取见 5.3.4。

5.3.2 空间分辨率转码中的特征选择

本节中，$ERROR$和$MODE$'的选择与帧内模式部分相同。

H.264 中以 4x4 子块为最小单位进行运动估计，因此，针对原始图像中 4 个宏块，无论哪种宏块类型下，每个 4x4 子块均有自身的运动矢量，则 4 个宏块在共有 64 个运动矢量(分为水平和竖直方向)。如果将这些运动矢量全部作为特征，势必导致巨量的计算。本文首先为当前宏块每个 4x4 子块计算出初始运动矢量，常用计算方法很多，如任选其一[1]，选择最大直流系数残差[19]，选择最多方向的运动矢量[14]，平均值[1]，中间值[1], [14]等等。根据文献[19]，中间值方法的效果较好，本文选择中间值方法。

这里定义这 16 个初始运动矢量集为 MVS，则它的元素 $\{mv_i \in MVS, i = 0,...,15\}$ 符合高斯分布：

$$\begin{aligned} P(mv_i) &= P(mv_i^x)P(mv_i^y) \\ &= \frac{1}{\sigma^x\sqrt{2\pi}}\exp\left\{\frac{-(mv_i^x-\theta^x)^2}{2(\sigma^x)^2}\right\}\bullet\frac{1}{\sigma^y\sqrt{2\pi}}\exp\left\{\frac{-(mv_i^y-\theta^y)^2}{2(\sigma^y)^2}\right\} \end{aligned} \tag{5.4}$$

这里假定水平和竖直方向的运动矢量相互独立，σ 和 θ 可由下式估计：

$$\begin{aligned} \theta^x &= \text{median}\left\{mv_i^x \in MVS, i = 0,....,15\right\} \\ \theta^y &= \text{median}\left\{mv_i^y \in MVS, i = 0,....,15\right\} \\ \sigma^x &= \frac{1}{16}\sum_{i=0}^{15}\left|\theta^x - mv_i^x\right| \\ \sigma^y &= \frac{1}{16}\sum_{i=0}^{15}\left|\theta^y - mv_i^y\right| \end{aligned} \tag{5.5}$$

下面给出一个实例说明，见图 5-2。图中横坐标为当前宏块的 16 个初始运动矢量值，纵坐标为对应的分布情况，其中实线为实际分布情况，而虚线表示根据上述模型计算的概率分布。

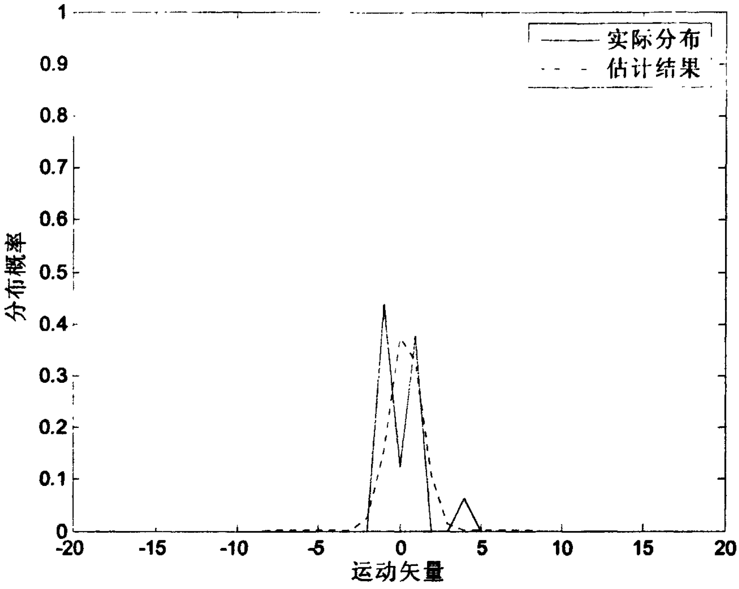


图 5-2 运动矢量的分布特征

因此利用该模型，原始的 64 个运动矢量可以简化为 2 个参数，从而极大地降低了特征向量的维数。为了能进一步节省计算量，本文使用下式作为运动矢量（式(5.3)中的 MV）的最终特征：

$$\begin{aligned} mv\_\sigma &= \left|\sigma^x\right| + \left|\sigma^y\right| \\ mv\_\theta &= \left|\theta^x\right| + \left|\theta^y\right| \end{aligned} \tag{5.6}$$

5.3.3 时间分辨率转码中的特征选择

在时间分辨率转码中，部分帧需要丢弃，因此，以丢弃帧作为参考帧的运动矢量需要重新计算，如图 1-3 给出的时间分辨率转码中的示范图例，F_n 为当前帧，$F_{n\text{-}1}$ 被丢弃，则 F_n 中指向 $F_{n\text{-}1}$ 中的运动矢量就要重新计算，并指向 $F_{n\text{-}2}$。常用的计算方法有 FDVS[63]，ADVS[13]等，根据本文第 4 章的实验结果，FDVS 效果最佳，因此本章直接采用 FDVS 方法合成运动矢量。

当时-空分辨率转码并存时，首先为跳帧之后的 64 个 4x4 子块单独计算运动矢量，然后使用中间值方法计算出 16 个初始运动矢量并提取特征。

在没有空间分辨率转码情况下，当前宏块只对应着原始图像中一个宏块，该

宏块的 16 个 4x4 子块的运动矢量就直接作为初始运动矢量来提取特征，该宏块的类型和非零系数比例即为 *MODES* 和 *ERROR*。

5.3.4 比特率转码中的特征选择

本文中比特率转码利用重新量化实现。输入量化参数称为 Q_i，重新量化参数称为 Q_r，Q_r 通常不小于 Q_i，为节省计算量，本文选择 Q_r-Q_i 作为量化参数的特征（式(5.3)中的 *QP*）。

5.4 帧内/帧间模式选择

用来预测宏块类型的支持向量机模型需要离线训练，训练中目标类型通过全搜索法得到。在预测宏块类型时，从解码信息中提取出特征向量需要利用判别函数与训练得到的支持向量进行比较。当支持向量的个数巨量时，会带来高昂的运算复杂度。为节省运算量，不少研究人员提出了精简的支持向量模型[83]-[85]。在这些精简模型中，训练得到原始支持向量由一些近似支持向量代替，从而大幅度精简了支持向量的个数。本文采用文献[84]提出的方法对训练得到的支持向量模型进行精简。

5.4.1 帧内模式选择

在帧内模式选择中，支持向量模型用来选择 I4MB 或 I16MB，需要进一步为 I4MB 和 I16MB 选择预测模式，具体选择方法见 3.4。

5.4.2 帧间模式选择

整个帧间模式选择的过程见下图：

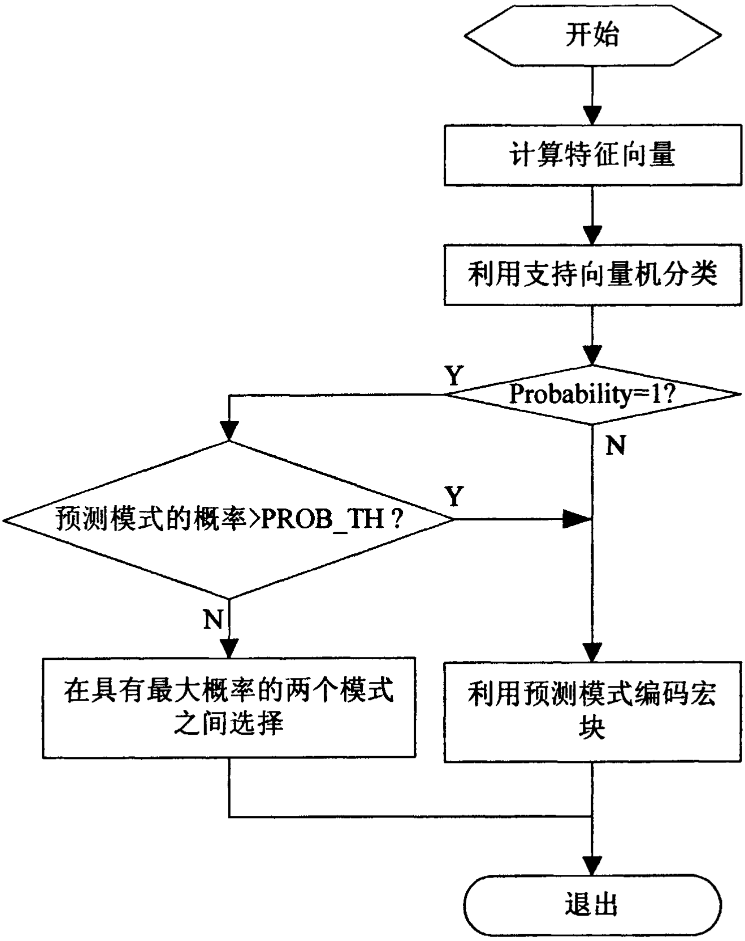


图 5-3 帧间类型选择流程图

在帧间类型选择中，首先从解码信息中提取特征向量，并引入到支持向量机模型来预测宏块类型。

由于 SKIP 类型可以视作无残差数据的 P16x16 类型，因此在本文中 SKIP 和 P16x16 合成一类处理，在编码 P16x16 类型时，如果没有残差数据，就自动调整为 SKIP 类型。

如果预测的是 P8x8 类型，所有 8x8 的子块（8x8, 8x4, 4x8, 4x4）都利用 3.5.2 中准则进行测试。

原始的支持向量模型不包含概率信息，但有时分类结果并非一定准确，文献[86]-[88]为支持向量机引入了概率模型，每个分类结果都对应着选择该类别的概率，进而更加准确的反映了分类结果。本文引用文献[86]提出的概率模型，如果

使用了概率模型（probability=1），且具有最大概率的类型的概率值如果小于阈值 PROB_TH，则从两个具有最大概率的类型中选择，选择准则见 3.5.2。

与上两章一样，在帧间类型选择中也需要运动矢量的细化，本章采用统一的运动矢量细化步长，细化步长与重新量化参数的关系见表 3-2。

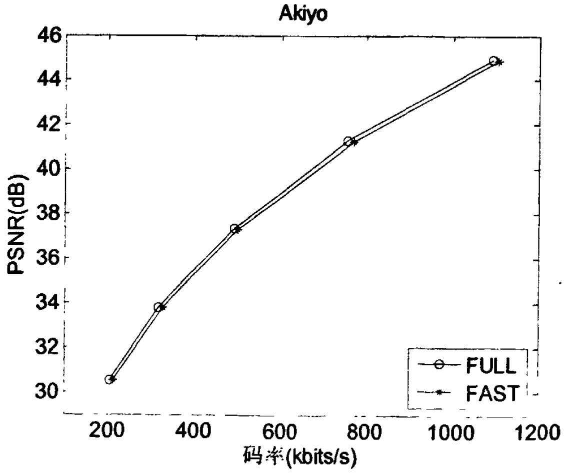
5.5 实验结果

实验中输入图像尺寸为 CIF (352x288)，帧率为 30 帧/秒，Q_i=20，Q_r={20, 25, 30, 35, 40}。编码原始视频序列的软件为常见 H.264 参考软件 JM12.1，具体参数见表 3-3。符号“FAST”表示提出的方法，“FULL”表示全搜索法。

另外，本文采用了从文献[88]下载的支持向量机软件，并选则径向基函数作为核函数$K(x_i,x_j)=\exp(-\gamma\|x_i-x_j\|^2),\gamma>0$，其中$\gamma=0.8$，惩罚因子 C=1。转码前，支持向量机模型需要利用大量的视频序列进行离线训练，并使用文献[84]提出的方法对训练得到的支持向量进行精简。

5.5.1 帧内模式选择实验

在帧内类型选择实验中，所有帧都使用 I 帧编码，图像缩放因子为 2，跳帧因子为 1，即无时间分辨率转码。各个序列的压缩性能见图 5-4。根据图中结果，本文提出的方法与全搜索法非常接近。如‘Akiyo’序列中，最大的 PSNR 损失为 0.3dB，其余三个序列几乎无任何损失。



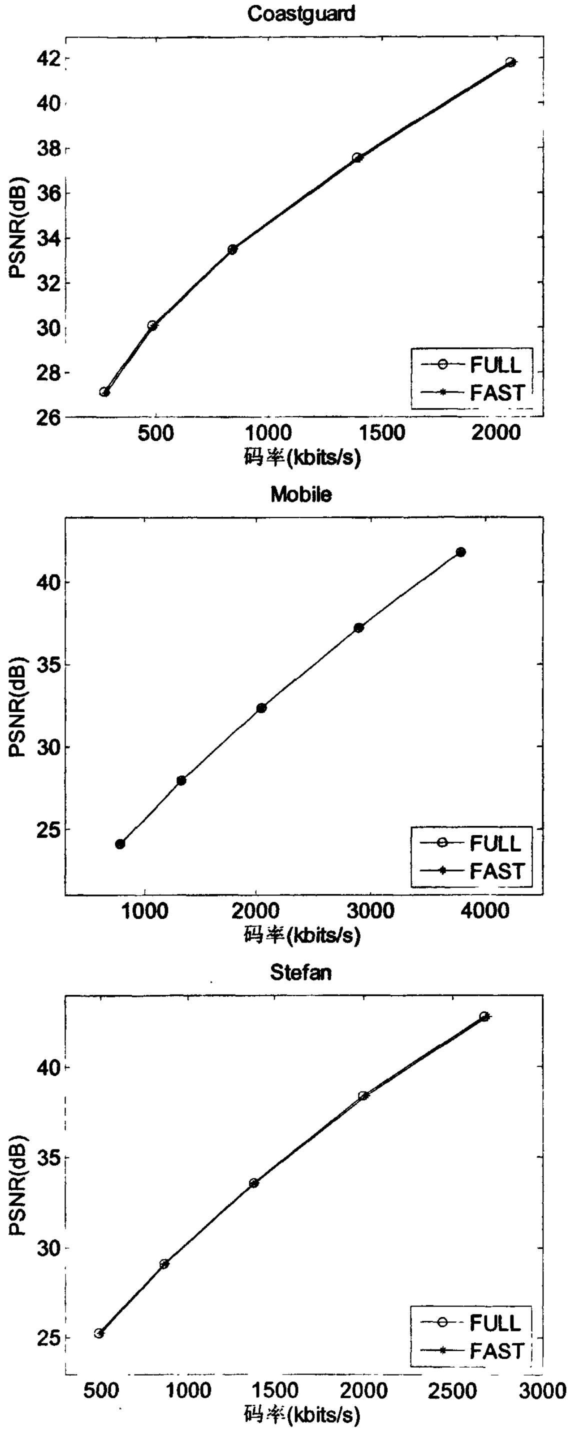


图 5-4 帧内类型选择中的压缩性能

下表给出了两种方法的运算耗时。从实验结果可以看出，相比于全搜索法，

本文方法在整体编码时间上能节省约 30%；若只考虑选择宏块类型时间，本文方法则可以提高约 15 倍，极大地节省了运算时间。

表 5-1 运算复杂度比较

序列	编码时间（秒）		选择宏块类型时间（秒）	
	FAST	FULL	FAST	FULL
Akiyo	3.71	5.31	0.20	2.46
Coastguard	4.44	6.31	0.18	2.71
Mobile	6.39	8.83	0.22	2.86
Stefan	4.63	6.75	0.21	2.76

5.5.2 不同特征向量的压缩性能比较

下面的实验结果中比较了不同的特征向量的性能。实验中图像缩放因子为 2，跳帧因子为 2 （帧率为 15 帧/秒），只有第一帧是 I 帧，其余帧均为 P 帧。使用前面分析的选择特征的方法可以组合出不同的特征向量，见表 5-2，由于本文考虑了比特率转码，因此每个特征向量均包含特征量化参数 *QP*，表中不再一一赘述。

表 5-2 不同的特征向量及其组成

向量名称	V_A	V_B	V_C	V_D	V_E	V_F	V_G
组成元素	*MV*	*ERROR*	*MODES*	*MV*, *ERROR*	*MV*, *MODES*	*ERROR*, *MODES*	全部

压缩性能的实验结果见表 5-3~表 5-5。表中的数据各种方法的 PSNR 值与全搜索法的差值。事实上，文中所提到的各个特征之间具有一定的重复性，比如当运动矢量的特征 mv_σ 和 mv_θ 都较小时，通常对应着 P16x16 或 SKIP 模式，这时特征 *MODES* 也较小，同时对应的残差数据也较小，因此这由三个特征组成的不同特征向量的压缩性能相差不多，而该结论也与下面的实验结果一致。

表 5-3 Akiyo

Bit rate(kbps)	V_A	V_B	V_C	V_D	V_E	V_F	V_G
5	+0.00	-0.14	-0.13	-0.10	-0.13	-0.09	-0.09
10	+0.02	-0.08	-0.05	+0.00	-0.04	+0.00	-0.04
20	+0.31	+0.26	+0.23	+0.29	+0.23	+0.26	+0.34
40	+0.44	+0.47	+0.48	+0.62	+0.60	+0.44	+0.48

表 5-4 Coastguard

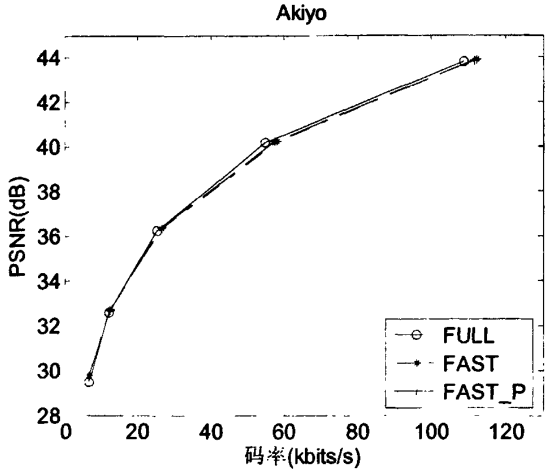
Bit rate(kbps)	V_A	V_B	V_C	V_D	V_E	V_F	V_G
50	+0.63	+0.54	+0.59	+0.24	+0.32	+0.26	+0.23
100	+0.42	+0.49	+0.47	+0.41	+0.36	+0.32	+0.31
200	+0.40	+0.41	+0.35	+0.42	+0.32	+0.32	+0.33
400	+0.42	+0.31	+0.24	+0.30	+0.23	+0.27	+0.28

表 5-5 Stefan

Bit rate(kbps)	V_A	V_B	V_C	V_D	V_E	V_F	V_G
80	+0.60	+0.57	+0.53	+0.58	+0.59	+0.59	+0.59
160	+0.71	+0.70	+0.73	+0.71	+0.73	+0.71	+0.71
320	+0.70	+0.66	+0.64	+0.72	+0.69	+0.68	+0.70
640	+0.50	+0.49	+0.51	+0.51	+0.50	+0.49	+0.50

5.5.3 空间分辨率转码实验

在空间分辨率转码实验中，只有第一帧使用帧内预测，其余帧均为 P 帧，图像缩放因子为 2。由于这里只考虑空间分辨率转码，因此，跳帧因子为 1，即无时间分辨率转码。采用特征向量‘V_F’，即 *ERROR* 和 *MODES*。图 5-5 给出了压缩性能比较，其中‘FAST’表示本节提出的方法，‘FAST_P’表示 5.4.2 中‘probability=1’的情况，硬阈值 PROB_TH=0.5，‘FULL’表示全搜索的结果。根据图中的实验结果，运动较为剧烈的序列，‘FAST’相对于‘FULL’的损失较高。如‘Stefan’序列中，最大的 PSNR 差值为 0.6dB。相对于‘probability=0’，当‘probability=1’时，压缩性能可以提高约 0.1dB。



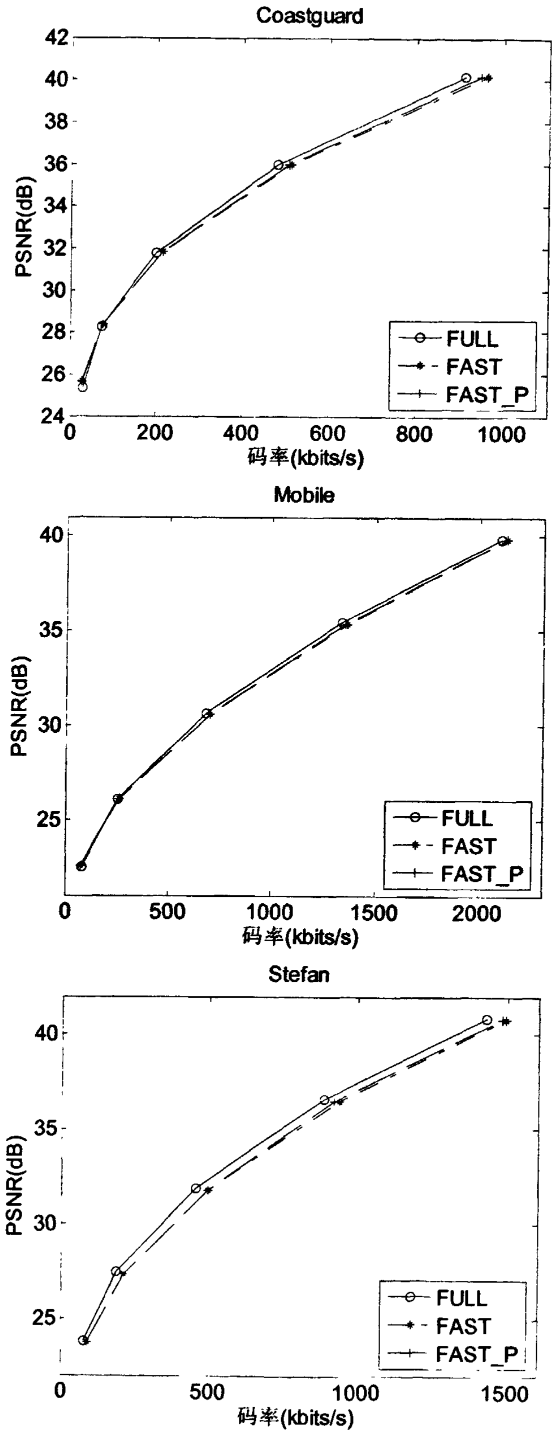


图 5-5 空间分辨率转码中压缩性能比较

下表给出了三种方法的运算复杂度的比较。根据表中的实验结果，相比于

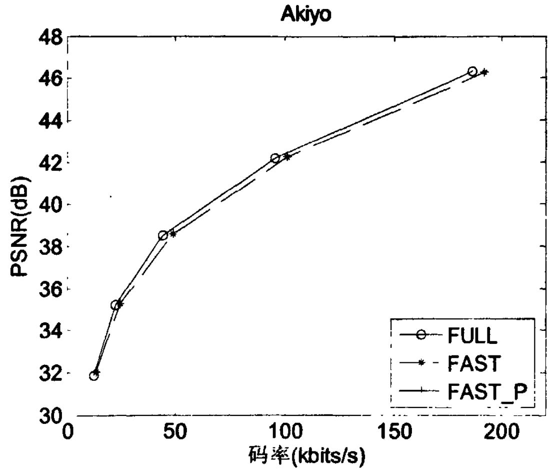
‘FULL’方法，‘FAST’方法在总的编码时间上可以提高约 10 倍；若仅考虑选择宏块类型的时间，则可以提高约 30 倍。‘FAST_P’方法比‘FAST’在宏块类型选择部分的运算耗时增加约 6%。

表 5-6 运算复杂度的比较

序列	重新编码时间 (秒)			选择宏块类型时间 (秒)		
	FAST	FAST_P	FULL	FAST	FAST_P	FULL
Akiyo	4.68	4.86	41.25	2.20	2.32	39.06
Coastguard	5.94	6.11	82.47	2.81	3.03	79.76
Mobile	6.78	7.19	94.69	2.86	3.04	91.01
Stefan	6.65	6.54	85.72	2.98	3.00	82.46

5.5.4 时间分辨率转码实验

本节中实验只考虑时间分辨率转码部分，图像缩放因子为 1，跳帧因子分别为 2（15 帧/秒），3（10 帧/秒）。实验结果分辨见图 9 和图 10。根据图中实验结果，与空间分辨率转码类似，运动较为剧烈的序列的压缩性能损失较高。另外，随着帧率的降低，压缩性能损失也递增。比如，序列‘Stefan’在帧率为 15 帧/秒时最大的 PSNR 损失为 0.5dB，而在帧率为 10 帧/秒时最大的损失为 1.1dB。在时间分辨率转码中，‘FAST_P’方法同样有效，能提高的 PSNR 值约为 0.2dB。



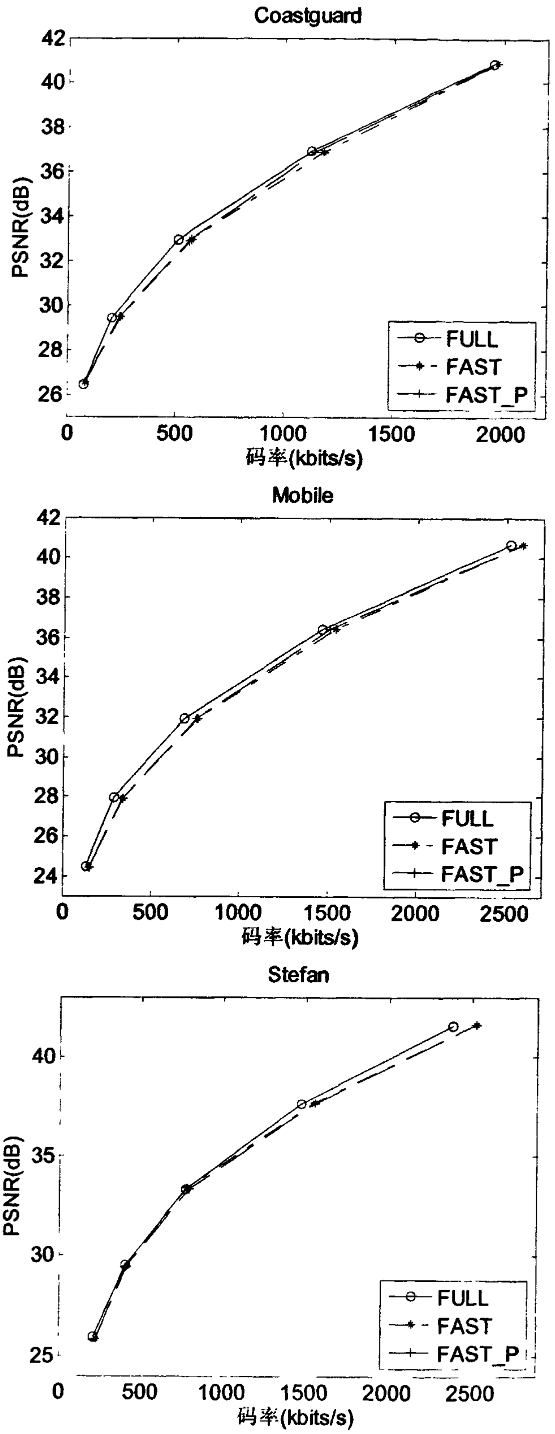
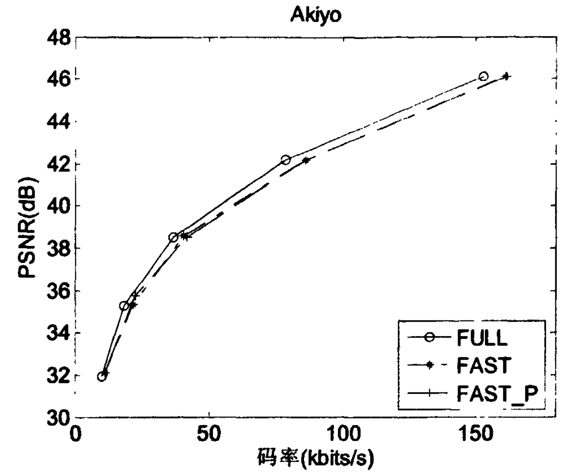
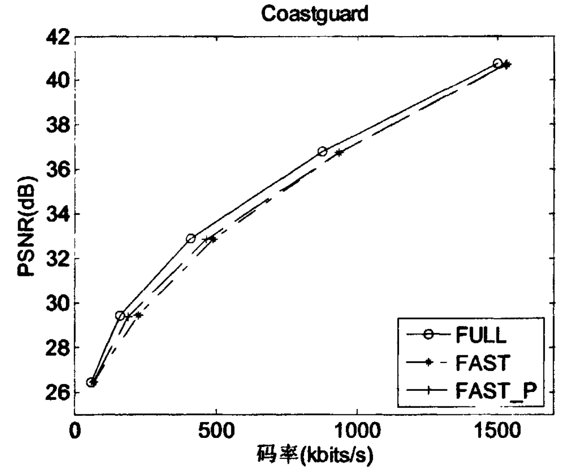
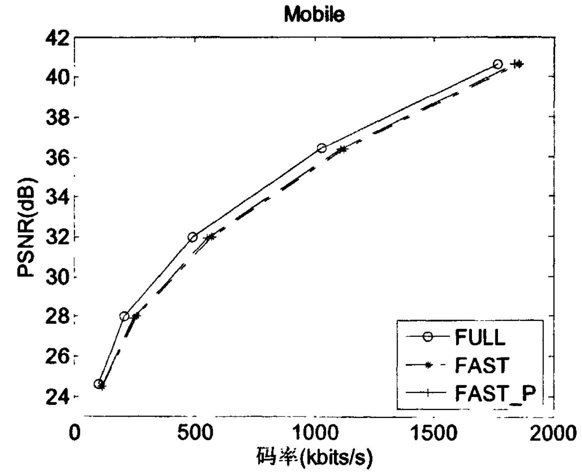


图 5-6 帧率为 15 帧/秒时的压缩性能比较


Akiyo


Coastguard


Mobile

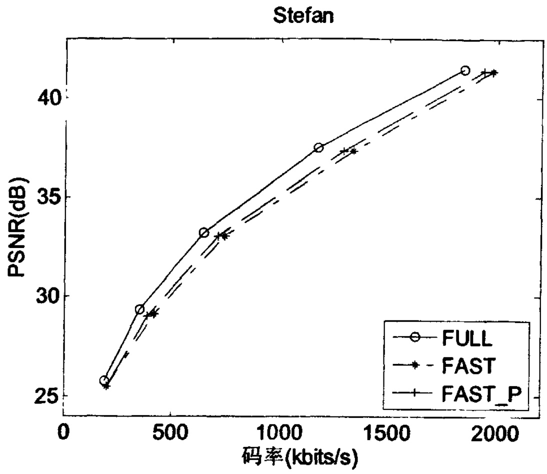


图 5-7 帧率为 10 帧/秒时的压缩性能比较

三种方法的计算复杂度见表 5-7 和表 5-8。根据表中的实验结果，总体编码时间上，'FAST'方法可以提高约 15 倍，选择宏块类型时间则能提约 25 倍。

表 5-7 帧率为 15 帧/秒时的计算复杂度比较

序列	重新编码时间 (秒)			选择宏块类型时间 (秒)		
	FAST	FAST_P	FULL	FAST	FAST_P	FULL
Akiyo	9.31	9.41	73.05	5.45	5.42	69.45
Coastguard	12.79	12.89	192.79	7.09	7.25	187.70
Mobile	12.59	12.61	174.08	6.45	6.70	168.56
Stefan	12.46	12.56	184.92	6.74	6.96	179.70

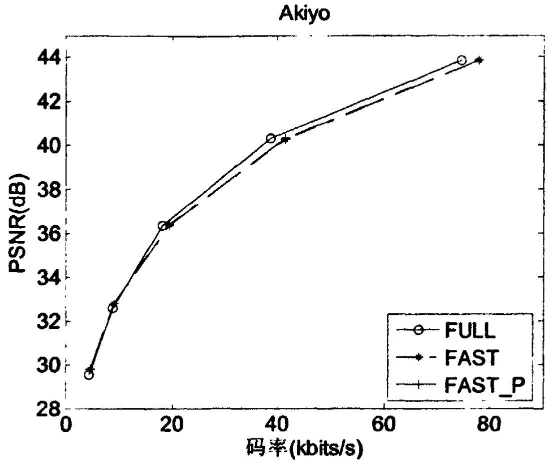
表 5-8 帧率为 10 帧/秒时的计算复杂度比较

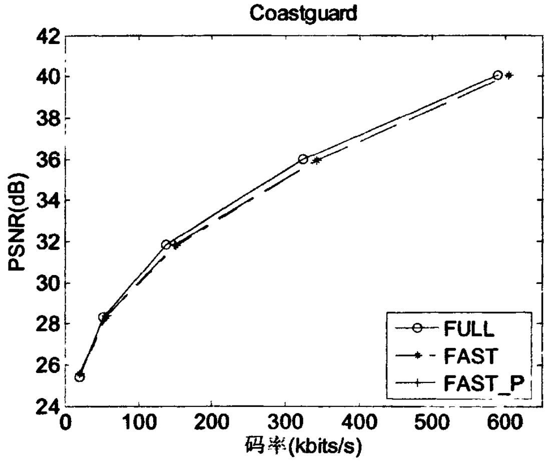
序列	重新编码时间 (秒)			选择宏块类型时间 (秒)		
	FAST	FAST_P	FULL	FAST	FAST_P	FULL
Akiyo	6.40	6.45	50.76	3.57	3.61	48.29
Coastguard	8.96	9.17	138.81	5.33	5.56	135.26
Mobile	8.64	8.72	115.98	4.55	4.84	112.32
Stefan	9.32	9.45	135.29	5.11	5.13	131.49

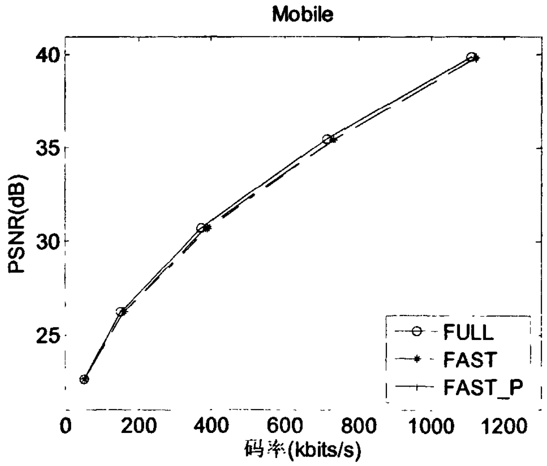
5.5.5 时-空分辨率转码实验

本节实验全面考虑了时间分辨率转码和空间分辨率转码。实验中的图像缩放因子为 2，跳帧因子分别为 2（15 帧/秒）和 3（10 帧/秒）。压缩性能见图 5-8 和图 5-9，根据图中实验结果，最大的 PSNR 损失约 1.2dB （'Stefan' 序列中

跳帧因子为 3（10 帧/秒）时）。运算复杂度比较见表 5-和表 5-10，本文方法可以将总的编码速度提高约 12 倍，选择宏块类型速度提高约 30 倍。







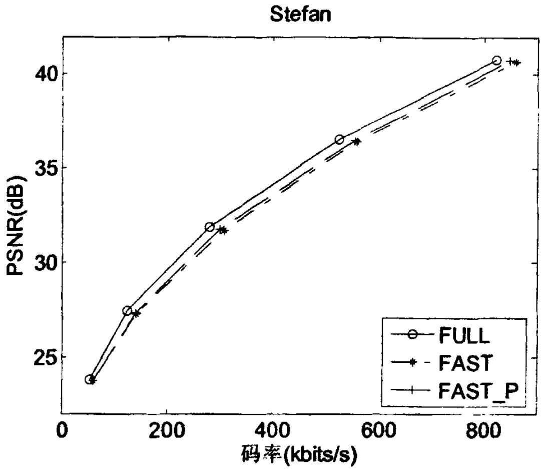
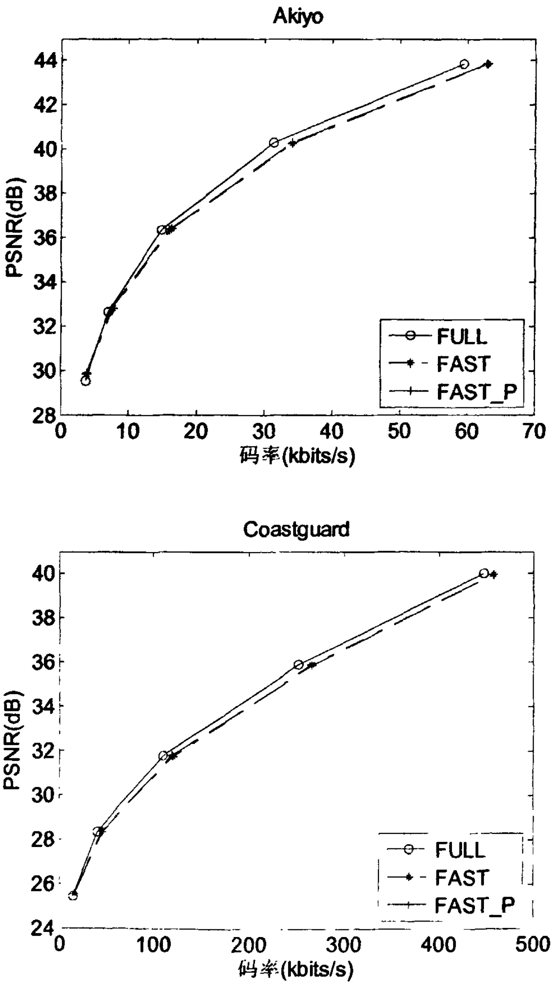


图 5-8 帧率为 15 帧/秒时的时-空分辨率转码



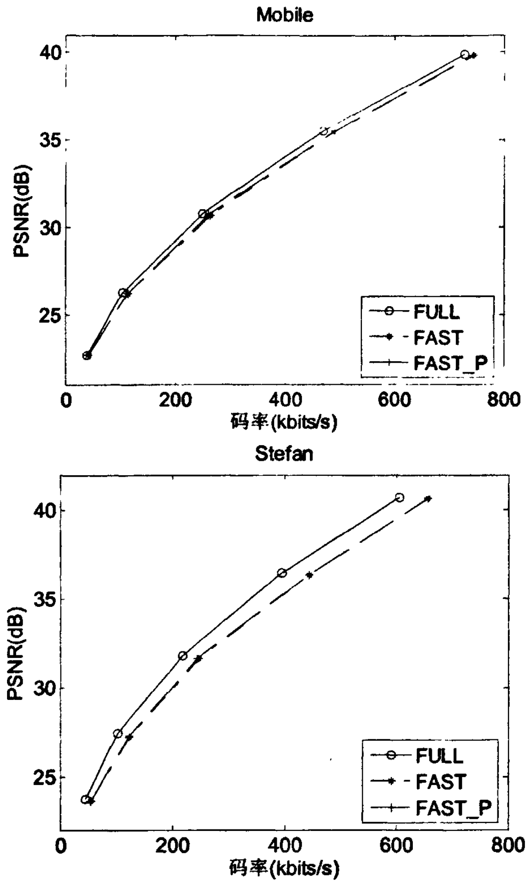


图 5-9 帧率为 10 帧/秒时的时-空分辨率转码

表 5-9 帧率为 15 帧/秒时的运算复杂度

序列	编码时间 (秒)			选择宏块类型时间 (秒)		
	FAST	FAST_P	FULL	FAST	FAST_P	FULL
Akiyo	2.50	2.62	20.72	1.17	1.21	19.67
Coastguard	3.21	3.37	45.63	1.59	1.62	44.33
Mobile	3.68	3.77	48.00	1.52	1.69	46.06
Stefan	3.41	3.60	46.95	1.53	1.65	45.29

表 5-10 帧率为 10 帧/秒时的运算复杂度

序列	编码时间 (秒)			选择宏块类型时间 (秒)		
	FAST	FAST_P	FULL	FAST	FAST_P	FULL
Akiyo	1.72	1.76	14.14	0.81	0.85	13.46
Coastguard	2.27	2.35	32.73	1.11	1.16	31.93
Mobile	2.41	2.52	31.12	0.94	1.09	30.50
Stefan	2.40	2.56	34.40	1.03	1.09	33.29

5.6 本章小结

本章首次提出以分类的方法在视频转码中来选择宏块类型。输入的比特流完全解码，并从解码信息中提取特征向量，包括运动矢量，原始图像中的宏块类型，量化参数，残差数据等。提取的特征向量引入到离线训练的支持向量机模型，来预测宏块类型。为检验本方法的有效性，本章给出了大量的实验结果。帧内模式选择部分，相比于全搜索法，在最大 PSNR 损失不超过 0.3dB 前提下，本章方法可以将选择宏块类型的速度提高约 15 倍。帧间模式选择部分，相比于全搜法，在最大 PSNR 损失不超过 1.2dB 前提下，可以将选择宏块类型的速度提高 25-30 倍。

第 6 章 基于 MCTF 的小波可分级编码技术

6.1 引言

在基于 MCTF 的小波编码方案中，当时间上滤波器类型和时间分解层次固定后，GOP 尺寸也就固定了。由于视频序列中的运动性质是随机的，故在固定的 GOP 尺寸中，有可能出现如下情况：运动比较平稳的一段视频被划分到不同的 GOP 内；而同一个 GOP 内包含了不同的运动性质。当解码端在时间上只重构很少的层次时，得到的视频流就非常的不连续，尤其当运动剧烈时，帧间的跳动就非常大。为了能得到更为灵活的 MCTF 编码方案，UMCTF （Unconstrained Motion Compensated Temporal Filtering）在文献[48]中被提出。该方案有一组控制参数，如：GOP 尺寸，时间分解层次，参考帧数目，高通（低通）帧数目等，用户可以根据不同的视频来进行设置。但文章并没有提到获取这些参数方法。

由于不同视频序列中运动性质各异，这就有必要使用自适应的 GOP 尺寸，且在 MPEG-x 及 H.26x 序列编码中已提出了一些方法。文献[49]，[50]就提出利用帧间差的色彩立方图 (Histogram of frame Difference, HOD) 来计算帧间的联系，另一个有关色彩立方图的方法是色彩立方图差(Difference Of Histogram, DOH) [49]. 针对 DOH 对局部运动信息不敏感的性质，有研究者提出了基于块的色彩立方图方法[50]，[51]。在文献[49]中使用了诸如当前帧和参考帧上所有宏块的 MAD （mean of absolute difference）、SAD（sum of absolute difference）和 SAD 的变化等参数来决定编码模式和 GOP 结构。在文献[52]中，划分 GOP 长度的标准是当前帧的帧内编码宏块个数。

在基于 MCTF 的小波视频编码方案中，自适应的 GOP 尺寸的使用见文献[41]和[54]。在时间上的当前分解层次下，非连接像素的个数作为评判的标准来决定是否进行下一层分解，并为此设定了一个阈值，因此利用该方法可以得到不同 GOP 结构，却无法改变 GOP 的尺寸。在标准 SVC 模型中也利用了自适应的 GOP 尺寸方式(Adaptive GOP Selection, AGS) [71]-[73]，在该方法中需要预定一个最大的 GOP 尺寸，如 16 帧，并以该尺寸下的所有子尺寸为单位分别进行预编码，即 16，8，4，2，并取重构之后效果最佳的组合，因此该方法的计算复杂度较高。

本文提出了一种自适应的选择 GOP 结构的方法，GOP 结构内包含 GOP 尺寸选择和低通帧选择两部分，进一步，MCTF 的编码方案根据选定的 GOP 尺寸和低通帧进行自适应的确定。在本文中使用的检测 GOP 结构的方法是互信息技术。

互信息可以度量帧间的变化量，因此，它可以作为评判镜头边界或关键帧提取的方法之一[55]-[57]。文献[55]使用互信息检测镜头切换、图像渐入、渐出等视频段，并且在分割得到各个帧簇中，使用互信息技术提取其中的关键帧。 而在文献[56]中，则提出了一种基于互信息的自动设定阈值的方法。摄像机的运动会导致帧间互信息的降低从而可能会造成错误检测，针对这个局限性，文献[57]应用一个特定的变换模型对互信息进行了最大化处理。

6.2 Haar MCTF（Motion Compensated Temporal Filtering）

在 MCTF 中采用了开环编码方法，故其可以完全避免“漂移”效应。在 MCTF 编码方案中，视频序列首先被划分成连续的 GOP，在一个 GOP 内部，原始的帧数据在时间上沿着运动轨迹进行滤波，如图 6-1 中时间滤波过程以 Haar 为例。如图所示，GOP 内每个帧对间使用一个双通道的 Haar 小波进行滤波，生成低通帧和高通帧，在下个层次上低通帧重复这一过程。如在第一层上，帧对（F0，F1）沿时间滤波，生成低通帧 L0 和高通帧 H0。在第二层上，第一层生成的低通帧 L0，L1，L2，L3 继续沿时间滤波，并再次生成低通帧 LL0，LL1 和高通帧 LH0，LH1。在最后一层，LL0 和 LL1 再次滤波，生成最后的 LLL0 和 LLH0。

在时间上完全分解后得到一个低通帧（LLL0）和若干个高通帧（LLH0，LH0，H0，H1，H2，H3）。最后这些滤波得到的低通帧和高通帧分别进行二维小波变换，小波系数使用嵌入式方法进行编码，如 MC-EZBC [41],[42]等。

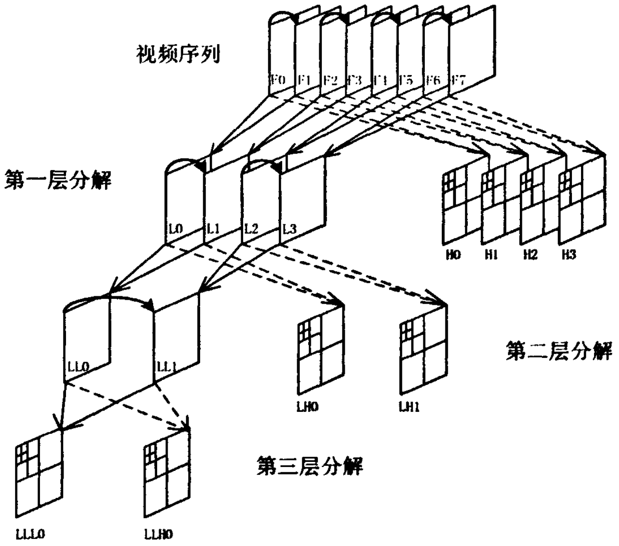


图 6-1 Motion Compensated Temporal Filtering (MCTF)

在时间上的分解过程中，为了进一步提高压缩性能，降低高通帧中的残差，常使用基于块的运动预测，如图 6-1 中帧对（F0，F1），F1 中每个块都以 F0 作为参考帧进行运动估计，并选择较好的匹配块来生成高通帧 H0，针对连接像素（connected pixels），低通帧（L）高通帧（H）有如下公式给出：

$$\begin{aligned} H[m,n] &= \frac{1}{\sqrt{2}} I_{2t+1}[m,n] - \frac{1}{\sqrt{2}} \tilde{I}_{2t}[m-d_m, n-d_n] \\ m-\bar{d}_m, n-\bar{d}_n] &= \tilde{H}[m-\bar{d}_m+d_m, n-\bar{d}_n+d_n] + \sqrt{2} I_{2t}[m-\bar{d}_m, n-\bar{d}_n] \end{aligned} \quad (6.1)$$

公式中 $I_{2t+1}[m,n]$ 代表帧 I_{2t+1} 中在位置 $[m,n]$ 上的像素值。如果使用分数像素的运动补偿，$\tilde{I}_{2t}[m-d_m,n-d_n]$ 则代表帧 I_{2t} 在位置 $[m,n]$ 上的插值结果，所使用的运动矢量为 (d_m,d_n)。$(\bar{d}_m,\bar{d}_n)$ 是帧 I_{2t} 反向的运动矢量，具体计算方法见文献[36]，[37]，$\tilde{H}[m-\bar{d}_m+d_m,n-\bar{d}_n+d_n]$ 是由高通帧（由公式(6.1)计算）的插值结果。同时，计算低通帧的过程又称为“预测”，而计算高通帧的过程称作“更新”，具体的细节讨论请参考文献[37]，[41]，[54]。

从图 6-1 可以看出，时间分解层次与 GOP 尺寸的关系如下:

$$n = 2^i \quad (i = 1,2,\ldots) \qquad (6.2)$$

这里 n 代表GOP尺寸, i 是时间上的分解层次。

如果其中一个参数固定，同时也就决定了另一个参数的值。在这种固定的模式下，视频序列中不同的运动性质处理方式一样。如果某个解码端只需要重构很低的时间层次，这时候较高时间层次的帧数据就要丢弃，则重构的视频序列可能非常不连续，尤其在运动较为剧烈的视频段。与此同时，在运动较为平缓的视频段，却能重构过多的帧而浪费了带宽。

6.3 互信息 (Mutual Information, MI)

互信息可以度量帧间的信息传递，因此可以用来检测镜头边界和关键帧提取[55]。帧间较大的变化对应着较小的互信息值，反之亦然。

设X是一个离散随机变量，其可能的取值范围$A_X=\{a1,a2,...,a_N\}$及每个值的概率为：$\{p_1,p_2,...,p_N\}$，$p_X(x=a_i)=p_i$，$p_i\geq 0$，则有$\sum_{x\in A_X} p_X(x)=1$。根据信息论，X的熵为：

$$H(X)=-\sum_{x\in A_X} p_X(x)\log p_X(x) \tag{6.3}$$

变量X，Y的联合熵为：

$$H(X,Y)=-\sum_{x,y\in A_X,A_Y} p_{XY}(x,y)\log(p_{XY}(x,y)) \tag{6.4}$$

变量X和Y的互信息为：

$$I(X,Y)=-\sum_{x,y\in A_X,A_Y} p_{XY}(x,y)\log\frac{p_{XY}(x,y)}{p_X(x)p_Y(y)} \tag{6.5}$$

互信息与联合熵的关系如下：

$$I(X,Y)=H(X)+H(Y)-H(X,Y) \tag{6.6}$$

在YUV色彩模型的视频序列中，亮度和色度的互信息值可以分别计算。设定视频序列的像素每个分量的取值范围为：0到N-1，其中N为最大灰度阶。针对亮度分量，$P^Y_{t,t+1}(i,j)$（$0\leq i,j\leq N-1$）是在帧F_t 中的某个像素的值为i而在帧F_{t+1}中改变为j的概率。因此就可以计算出这两帧间的亮度分量的互信息值为：

$$MI^Y_{t,t+1}=-\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}P^Y_{t,t+1}(i,j)\log\frac{P^Y_{t,t+1}(i,j)}{P^Y_t(i)P^Y_{t+1}(j)} \tag{6.7}$$

$$MI_{t,t+1}=MI^Y_{t,t+1}+MI^U_{t,t+1}+MI^V_{t,t+1} \tag{6.8}$$

同样，色度分量的互信息值$MI^U_{t,t+1}$,$MI^V_{t,t+1}$的计算方法同亮度一样。

众所周知，视频序列大部分能量集中在亮度分量上。因此，本文使用的运动

预测和互信息计算只考虑亮度分量。

互信息可以检测帧间的变化，因此它肯定也能检测帧间的运动性质，也就是说剧烈的运动肯定对应着较低的互信息值，而缓慢的运动则得出较高的互信息值。下面通过一个实验来具体说明。在实验中，使用拉格朗日算子[37]来平衡运动矢量占用比特流长度和残差数据，运动预测的目标就是试图获取下式的最小值：

$$COST = \lambda R_{mv} + D_{pred} \tag{6.9}$$

其中R_{mv}是运动矢量占用的比特流，D_{pred}是残差数据的能量，在本实验中拉格朗日算子λ=24。为能更明了的展示实验结果，我们对$COST$取倒数，并且互信息值和$1/COST$ 都经过了归一化处理。实验中还给出了另外两种常用的GOP尺寸划分方法DOH和HOD，结果见图 6-2。

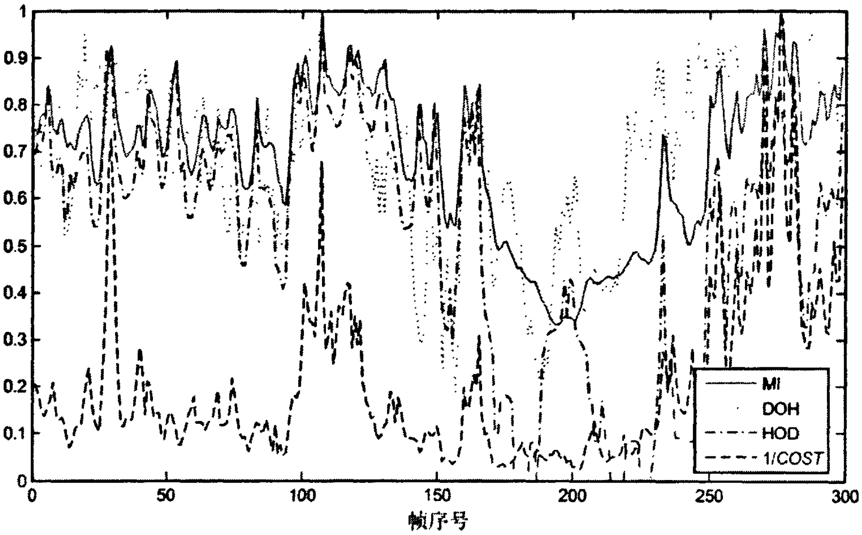
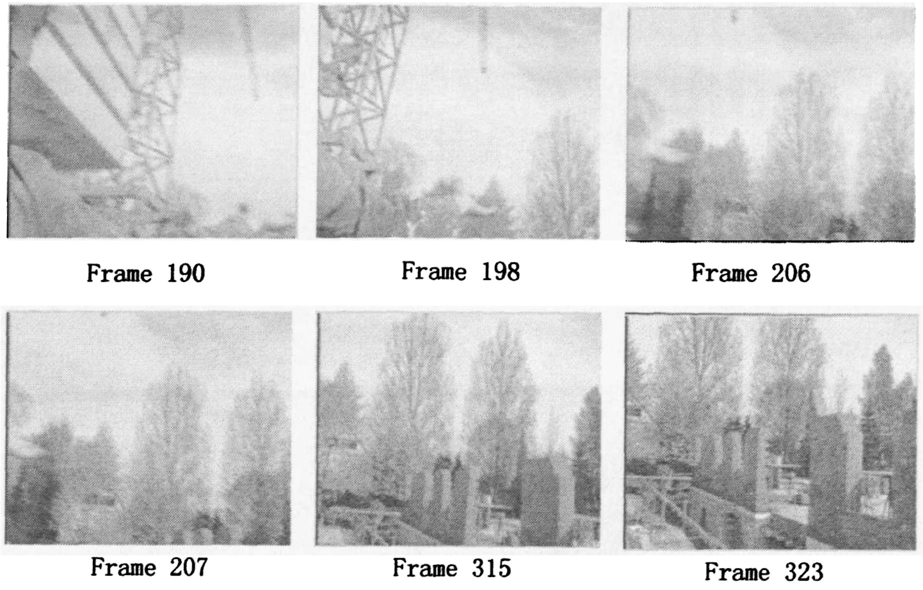


图 6-2 互信息，DOH，HOD 与运动预测的关系

从图 6-2的实验结果可以看出，当运动较为缓慢时（1/$COST$ 较高），就对应着较高的互信息值。相反，如果运动较为剧烈（1/$COST$ 较低），则对应着较低的互信息值。比如在第200帧前后，查看原始视频序列，不难发现第200前后对应着镜头切换部分，运动量较大，见图 6-3。



Frame 190　Frame 198　Frame 206

Frame 207　Frame 315　Frame 323

图 6-3 视频序列中镜头切换

根据图 6-2 的实验结果，DOH 和 HOD 的确也可以反映出序列的运动性质，但相比这两种方法，互信息方法与运动预测消耗曲线匹配的效果更好，比如在帧序号为 200 左右，DOH 和 HOD 方法均出现较大的波峰，而互信息方法则能更为准确地反映出镜头转换。

6.4 类 Haar 的 MCTF 编码方案

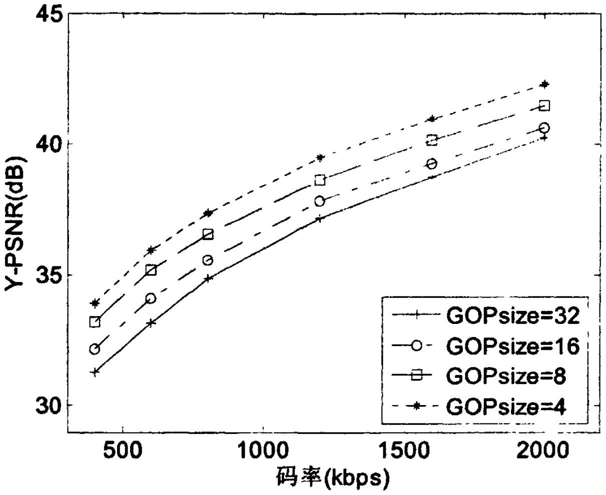
在本文提出的类 haar 的 MCTF 编码方案中，包含 GOP 结构选择和时间分解层次确定两部分，其中 GOP 结构根据互信息自适应的确定，又包含了 GOP 尺寸选择和低通帧选择两部分，最后，根据选定的 GOP 结构基础上来选择时间上的分解过程。

6.4.1 自适应的 GOP 尺寸

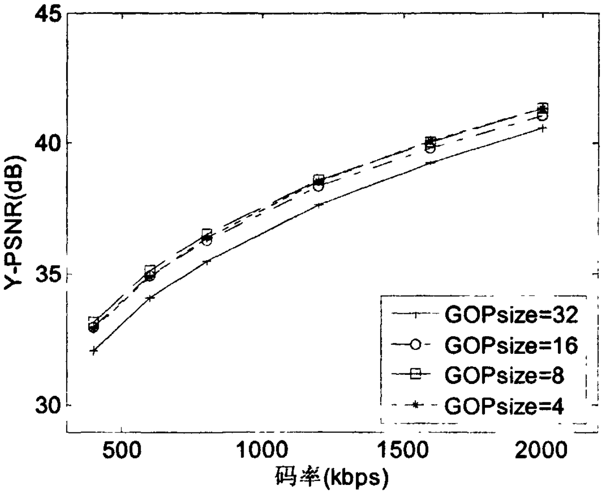
就如在6.3的描述，帧间剧烈的运动对应着较低的帧间互信息值，同时运动预测的*COST*也较高，反之亦然。因此可以利用帧间的平均互信息值来作为选择GOP尺寸的标准，当帧间平均MI较高时，可以选择较长的GOP尺寸，而较低的平均互信息值对应着较短的GOP尺寸。

从下面的实验还可以看出，在运动剧烈的视频段选择较短的GOP尺寸，而运动缓慢的视频段选择较长的GOP尺寸还可以在一定的程度上提高压缩性能。在实验中以视频序列“Foreman”为例，图像尺寸为CIF（352x288），帧率为30帧/秒。

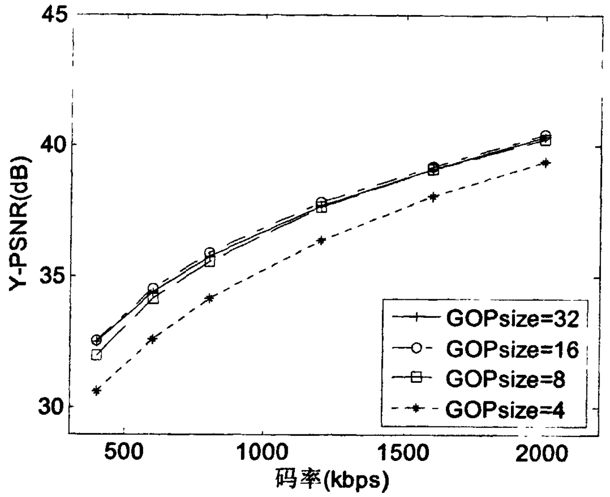
从实验结果（图 6-4）可以看出，在平均互信息值较低的视频段（帧编号：180-211），较短的GOP尺寸取得较好的压缩性能（GOP为4最好），同时，平均互信息值较高的视频段（帧编号：250-281），较长的GOP尺寸的压缩性能较好（GOP为32时效果最好）。



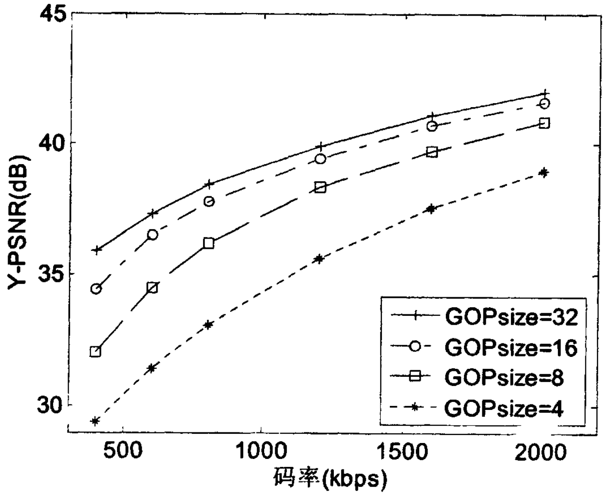
(a) 帧编号: 180-211 帧数目: 32 平均 MI=1.420751



(b) 帧编号: 197-228 帧数目: 32 平均 MI=1.538233



(c) 帧编号: 212-243 帧数目: 32 平均 MI=1.775050



(d) 帧编号: 250-281 帧数目: 32 平均MI=3.069295

图 6-4 不同运动类型及不同的 GOP 尺寸下的视频段的压缩性能

6.4.2GOP 尺寸的选择

因此，根据上述实验的结论，可以利用平均互信息值来选择GOP尺寸，经过大量的实验，下面给出平均互信息值和GOP尺寸的关系的经验数据，见表6-1，其中参数 *low_MI, median_MI* 和 *high_MI* 是用于控制GOP尺寸的阈值。

表 6-1 平均互信息值和 GOP 尺寸关系

平均互信息值 （*average_MI*）	GOP 尺寸
average_MI < *low_MI*	4
low_MI <= *average_MI* < *median_MI*	8
median_MI <= *average_MI* < *high_MI*	16
high_MI <= *average_MI*	32

如果某段视频下互信息值间的变化过大，这表明该视频段帧间的运动性质不一，不宜划分到同一个GOP内。因此，在利用平均互信息值来划分GOP的同时，本文还通过帧间各互信息值间的标准差来控制GOP的尺寸，以防止同一个GOP内运动变化过快，本文中所使用标准差阈值为*var_T*。

下面给出选择 GOP 尺寸的伪代码：

```
a:  初始化：
        n=0;
        设置标准差阈值 $var\_T$;
        读取第一帧数据 $F0$;
b:  决定 GOP 尺寸
        n++;
        读取一帧数据 $F_n$;
        计算帧$F_{n-1}$和$F_n$亮度分量的互信息值$MI_{n-1,n}$并且：
        计算互信息值集合$\{MI_{0,1},MI_{1,2},\ldots,MI_{n-1,n}\}$的平均互信息值，如下给出：
        $average\_MI = \sum_{i=0}^{n-1} MI_{i,i+1} / n$
        并且：
        if( ($average\_MI < low\_MI$) && (n >= 4) )
            encode_GOP();
        else if( ($low\_MI <= average\_MI < median\_MI$) && (n>=8) )
            encode_GOP();
        else if(($median\_MI <= average\_MI < high\_MI$) && (n>=16)
            encode_GOP();
        else if($average\_MI >= high\_MI$) &&(n>=32))
            encode_GOP();
```

```
else
        计算互信息值集合$\{MI_{0,1},MI_{1,2},\ldots,MI_{n-1,n}\}$的标准差$\sigma_n$
        if($\sigma_n$ >= $var\_T$) encode_GOP();
        else      goto b:
```

"encode_GOP()" 是本文提出的类haar的MCTF编码方法对一个GOP进行编码。帧间互信息值的亮度分量"$MI_{n-1,n}$"可由等式(6.7)计算。

在本文提出的方法中，平均互信息值和标准差同时用来控制GOP尺寸，选定的GOP尺寸不仅能根据运动类型的变化自适应的改变，而且同一个GOP内部的运动类型也能保持一致。

6.4.3 低通帧的选择

在传统的基于MCTF的小波编码方案中，低通帧的位置由时间上滤波类型决定。如图 6-1，在每个GOP内低通帧LLL0 都在位置$F0$。如果解码端只重构一层，则重构帧位置固定。但是，这些重构的帧时常并不是当前GOP的最佳代表。据笔者所知，目前还尚未有文献讨论MCTF中低通帧帧选择问题。关键帧提取可以从一个视频段中提取一个或多个帧作为当前视频段的代表，该方法常用在视频检索中[55]，[56]。因此，关键帧提取技术同样可以用于GOP内低通帧的选择。本文，引入文献[55]中从一个帧簇（frame clusters）中提取关键帧的方法从一个GOP内选择低通帧。在该方法中，某段视频段（GOP）内与其他帧间具有最大的互信息值的帧被选作为最具代表性的帧，见下式：

$$F_{key}=\max_{j}\left(\frac{1}{N}\sum_{\substack{i=0\\ j\neq i}}^{N-1}MI_{j,i}\right) \tag{6.10}$$

其中N 是视频段的帧数目，在本文中，就是选定的GOP。

$MI_{j,i}$ 是帧F_j和F_i亮度分量的互信息值，其中i和j是帧编号。

F_{key}是从GOP选定的最具代表帧。

在本文的方法中，我们可以看出，在以互信息值为选择标准下，选定的最具代表帧与其他所有帧保持着最大的关联。

6.4.4 时间分解过程确定

在选择了GOP结构后（GOP尺寸和低通帧位置），GOP尺寸可能不再满足式(6.2)，而低通帧的位置也不再是固定位置，从而新的GOP结构不再符合传统的基

于haar的MCTF编码过程，因此图 6-1所示的滤波过程不再适用。

在基于MCTF的视频编码方案中，时间上的分解过程是以帧对为单位进行的。随着分解层次的增加，帧对间的距离也逐渐递增。如图 6-1所示，在时间分解层次分别为1层，2层，3层上，帧对间的距离分别为1帧，2帧，4帧。如果帧对间的距离过长，势必造成帧间相差过大，从而运动预测残差较大，消耗很长的比特流。因此，在时间分解过程中当前层次上，距离低通帧最远的帧需要尽可能在较低的层次上分解完毕，从而尽量缩短剩余帧离低通帧的距离。

基于上述分析，本文提出了一个类haar的MCTF编码方法，它可以根据选择的GOP尺寸和低通帧位置，自动确定时间上的编码层次和每层上运动预测的方向。在本文方法中，处在低通帧前方和后方上最远的帧都尽可能的在较低的层次上分解，从而保证了下个层次中的剩余帧与低通帧保持较短的距离。

在本文中时间滤波器类型为haar，由于低通帧位置的自适应确定，如果采用传统基于haar小波的MCTF编码方案中的运动补偿的方向，势必导致帧对间的距离的增加。在本文的方法中，处在低通帧前方的帧使用后向运动预测，处在低通帧后方的帧使用前向运动预测，从而尽可能地保证剩余帧与低通帧的距离不止太远。

下面以GOP尺寸为14，低通帧位置是F8举例说明，见图 6-5。从图中明显可以看出，当GOP尺寸符合式(6.2)，并且低通帧位置为图 6-1中F_0时，该方法可以与传统的时间分解过程保持一致。

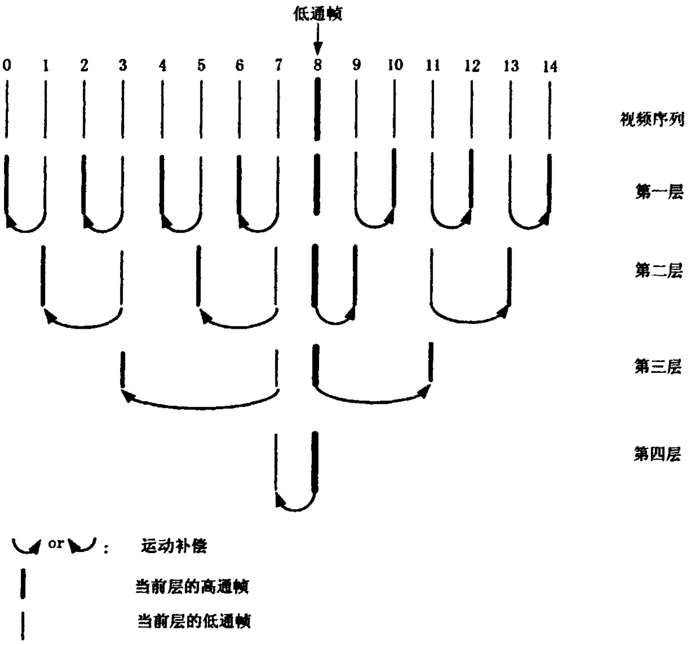


图 6-5 时间分解过程

6.5 实验结果

本文实验的运行环境为 Intel Pentium-IV 2.66GHz，512M 内存，Microsoft windows 操作系统。实验中选择不同运动类型的视频序列，见表 6-2。文中选择 HVSBM[37] 中的变块范围为 64x64 到 4x4。MCTF 得到低通帧和高通帧由 MC-EZBC[41]，[54] 进行编码。另外，在本实验中使用了部分出[55]下载的参考软件。

在实验中还给出了与标准 SVC 模型中的 GOP 尺寸选择方法 AGS 的比较，另外还给出了与传统的 H.264 编码结果的比较。输入的原始图像由最常用的 H.264 参考软件 JM12.1 编码，部分编码参数见表 3-3。

6.5.1 压缩性能比较

表 6-2 视频序列及其运动类型

视频序列	帧尺寸	帧数目	帧率	运动类型
Mobile	CIF（352x288）	300	30帧/秒	较缓
Foreman	CIF（352x288）	300	30帧/秒	中等，镜头转换
Stefan	CIF（352x288）	300	30帧/秒	较高
Football	SIF（352x240）	125	30帧/秒	较高
Tennis	SIF（352x240）	112	30帧/秒	镜头切换

表 6-4-表 6-8 给出了各个序列的实验结果，其中‘GOP8’和‘GOP16’表示GOP尺寸为 8 和 16 的传统 MCTF 编码方案的实验结果。实验中使用了两组 GOP 尺寸控制的阈值‘ADGOP1’和‘ADGOP2’，见表 6-3。

表 6-3 自适应选择 GOP 尺寸的参数集

名称	*low_MI*	*Median_MI*	*High_MI*	*var_T*
ADGOP1	1.5	2.0	3.0	0.15
ADGOP2	1.4	1.9	3.1	0.14

符号‘method+KF’ 表示方法“method”使用了低通帧选择

从实验结果可以看出，对于运动性质有明显变化的序列（Foreman），或者有镜头切换的序列（Tennis），尤其对后者，使用自适应的选择 GOP 尺寸的方法能取得较好的压缩性能，如在相同的比特率下，Tennis 序列约能提高 0.8-1.0 dB。而对运动性质变化不大的序列，如 Mobile，Stefan，Football，自适应选择 GOP 尺寸的方法在压缩性能上略有降低。

相比于传统的MCTF编码方案，针对较为剧烈的运动序列，如Stefan，Football，本文提出的自适应低通帧选择方法可以取得更好的压缩效果。在相同比特率下，图像质量大约能提高0.3-0.5 dB。

在具有镜头切换或者运动缓慢的序列，本文方法的压缩性能基本上与H.264持平，在其余的序列中，本文方法仍然与H.264的压缩性能有较大的差距，压缩性能也是基于MCTF的编码方式的一个研究课题。表 6-9 还给出了本文方法与H.264的运算复杂度的比较，从实验结果可以看出，本文方法的编码速度比H.264慢得多，这也是基于MCTF编码方案的另一个研究方向。

在与AGS方法的比较中，本文方法可以提高的PSNR值约0.25 dB。尤其是在镜头切换的序列，如‘Tennis’，本文方法可以提高的PSNR约0.9dB。如果仅考虑GOP尺寸选择部分，除去镜头切换的情况，其他序列中本文方法要比AGS损失的PSNR值约0.15dB。表 6-10 还给出了本文方法和AGS的选择GOP结构的运算复杂度的比较。由于在AGS方法中需要对各种子GOP尺寸进行预编码。如当最大GOP尺寸为16时，则需要对尺寸为16，8，4，2等四种方式进行预编码，该过程是非常耗时的部分。因此，本文方法的GOP结构选择的运算复杂度要远远少于AGS方法。

表 6-4 Mobile 的率失真比较

码率 (kbps)	PSNR(dB)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
400	24.26	26.13	26.13	26.87	25.92	25.94	26.13	26.12
600	26.99	28.98	28.98	28.49	28.68	28.68	28.98	28.95
800	29.00	30.69	30.69	29.67	30.51	30.42	30.69	30.64
1200	31.67	32.99	32.99	31.62	32.81	32.75	32.99	32.92
1600	33.52	34.54	34.54	33.14	34.40	34.37	34.54	34.41
2000	34.95	35.93	35.93	34.38	35.79	35.69	35.93	35.78

表 6-5 Foreman 的率失真比较

码率 (kbps)	PSNR(dB)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
400	32.93	33.29	33.61	35.64	33.52	34.01	33.50	34.00
600	34.94	35.18	35.51	37.21	35.35	35.72	35.36	35.80
800	36.29	36.41	36.78	38.36	36.48	37.13	36.59	37.03
1200	38.28	38.32	38.69	39.95	38.45	38.92	38.47	38.84
1600	39.68	39.67	40.05	41.13	39.79	40.31	39.83	40.17
2000	40.87	40.84	41.20	42.13	40.99	41.17	40.97	41.28

表 6-6 Stefan 的率失真比较

码率 (kbps)	PSNR(dB)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
600	28.45	28.63	28.76	31.50	28.51	28.96	28.55	29.14
800	30.30	30.40	30.56	32.75	30.33	30.76	30.33	30.89
1200	32.82	32.74	33.01	34.69	32.79	33.20	32.83	33.29
1600	34.58	34.46	34.76	36.22	34.64	35.13	34.61	35.07
2000	36.03	35.86	36.20	37.50	36.15	36.42	36.08	36.47

表 6-7 Football 的率失真比较

码率 (kbps)	PSNR(dB)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
600	25.12	24.66	25.34	27.39	25.18	25.62	25.10	25.58
800	26.43	25.99	26.58	28.77	26.42	26.92	26.38	26.86
1200	28.31	27.90	28.43	30.78	28.29	28.81	28.27	28.74

1600	29.93	29.56	30.00	32.34	29.89	30.37	29.88	30.31
2200	31.89	31.50	31.96	34.37	31.87	32.46	31.82	32.29
2600	33.10	32.73	33.18	35.47	32.98	33.61	33.02	33.50
3000	34.21	33.92	34.26	36.52	34.09	34.62	34.16	34.58

表 6-8 Tennis 的率失真比较

码率 (kbps)	PSNR(dB)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
400	29.84	30.38	30.54	31.52	31.31	31.39	31.31	31.40
600	31.61	32.10	32.29	33.18	33.18	33.25	33.11	33.15
800	32.96	33.51	33.67	34.51	34.51	34.61	34.48	34.57
1200	35.06	35.42	35.61	36.37	36.42	36.59	36.44	36.56
1600	36.71	37.05	37.23	37.74	37.92	38.10	37.98	38.01
2000	38.03	38.22	38.39	38.90	38.97	39.12	39.06	39.13

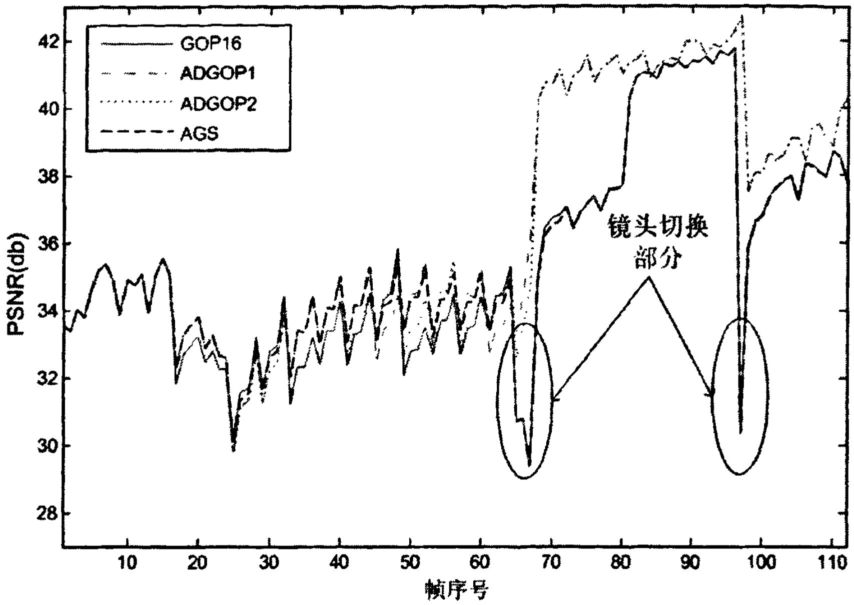
表 6-9 运算复杂度的比较

	Mobile	Foreman	Stefan	Football	Tennis
H.264 (sec)	335	315	328	125	101
ADGOP1 (sec)	6672	6486	7028	2873	2624

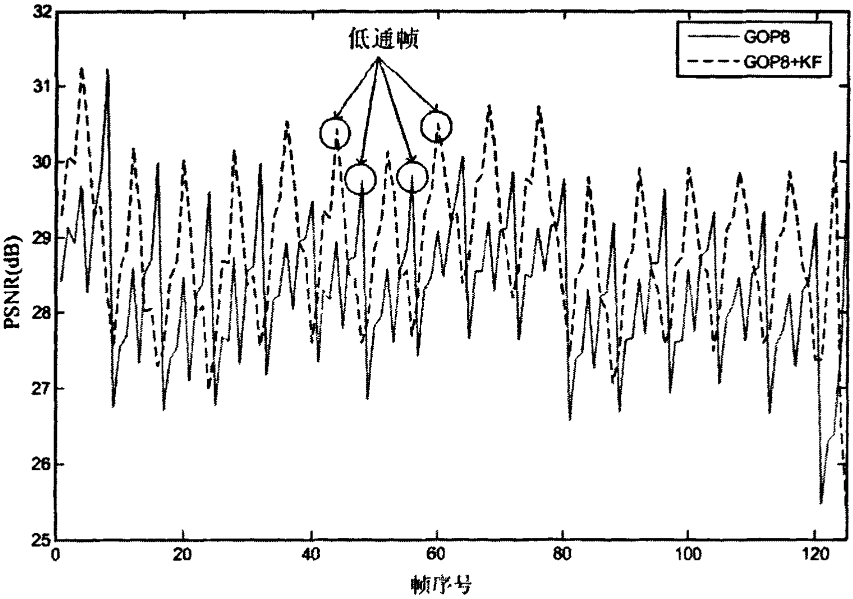
表 6-10 选择 GOP 结构运算复杂度的比较

	Mobile	Foreman	Stefan	Football	Tennis
AGS (sec)	17610	17700	17570	6093	5259
ADGOP1 (sec)	77	72	65	17	15

图 6-6给出了视频序列‘Tennis’和‘Football’在码率1200kbps情况下各帧的失真比较。在序列‘Tennis’中，帧编号为66和96时对应着两次场景变换，因此也对应两个非常大的图像质量的下降。在本文提出的自适应的选择GOP尺寸方法中，图像质量得到了明显的提高，见图 6-6-(a)箭头所指区域。另外，根据图中还给出的AGS方法的结果，该方法对镜头切换的检测性能较弱，因此在镜头切换部分的压缩性能没有明显的提高。在图 6-6-(b)中比较了GOP尺寸同为8时低通帧选择与固定低通帧方法的率失真情况，经过选择低通帧，整体图像质量提高了约0.5dB。



(a) 率失真比较tennis kbps=1200



(b) 率失真比较football kbps=1200

图 6-6 序列中率失真比较

6.5.2 时间上解码分析

在本文的方法中，GOP 的选择根据运动情况自适应而定。我们知道，运动剧烈的视频段含有较多的信息，因此使用较短 GOP 尺寸，这样当解码端只重构

较低的时间层次时，可以得到更多的帧数据。同样，运动平缓的视频端帧间相差不大，信息也较少，可以使用较长的 GOP 尺寸来编码，即使解码端重构的帧较少时，也可以重现原始视频序列的运动过程。下面给出序列“Foreman”的部分实验结果。使用表 6-3 给出的参数集‘ADGOP1’，得到“Foreman”的 GOP 划分情况见下表。

表 6-11 Foreman 的 GOP 划分

帧编号	GOP 尺寸	*var_T*	*Average_MI*
000-015	16	0.041909	2.956068
016-028	13	0.148091	2.920942
029-044	16	0.044507	2.948020
045-060	16	0.089403	2.918054
061-076	16	0.015122	2.951476
077-092	16	0.050900	2.639132
093-096	4	0.122002	2.730832
097-128	32	0.045612	3.454166
129-139	11	0.149865	3.002180
140-155	16	0.141933	2.584028
156-158	3	0.034239	2.358859
159-168	10	0.139096	3.053249
169-176	8	0.014505	1.901475
177-184	8	0.004491	1.568553
185-188	4	0.000806	1.438558
189-192	4	0.001846	1.305902
193-196	4	0.000872	1.273837
197-200	4	0.000066	1.306545
201-204	4	0.006877	1.491141
205-212	8	0.000553	1.536467
213-220	8	0.001397	1.571624
221-228	8	0.000801	1.662251
229-242	14	0.064574	1.982049
243-249	7	0.071476	2.012392
250-265	16	0.070580	2.820932
266-284	19	0.141868	3.219963
285-299	15	0.082141	2.844399

当解码端只重构时间上一个层次，在图6-7中给出重构的帧数目比较。

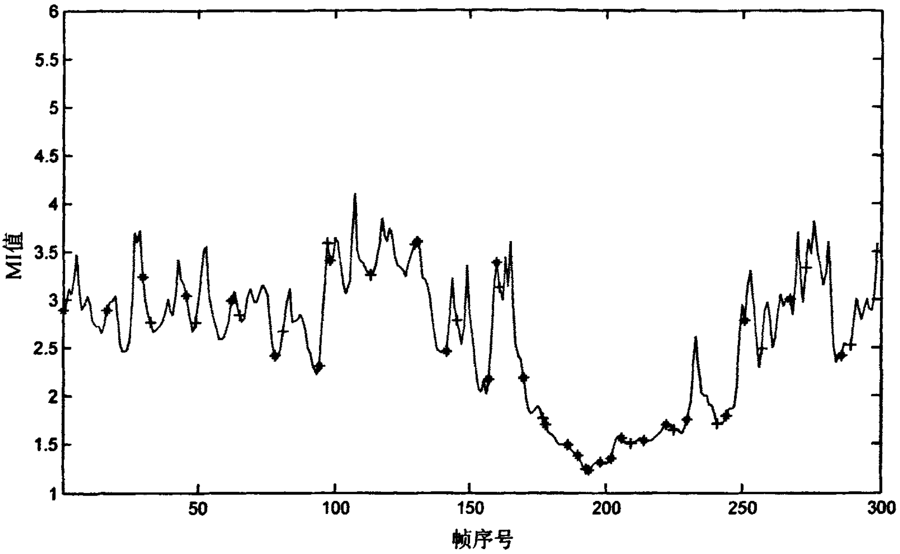


图 6-7 固定GOP尺寸和自适应GOP尺寸的比较

在图 6-7 中，曲线表示帧间的互信息值，即帧间的运动情况. ‘+’ 标记了在GOP尺寸固定为16时可以重构的帧编号，该方法重构的帧显然均匀分布在整个序列上。‘*’标记了自适应的GOP方法重构的帧，当然这些重构帧的分布是不均匀的。运动平稳时（帧编号110左右）重构帧较少；运动剧烈时（帧编号200左右）重构帧较多。运动平缓的视频段帧间相差不大，信息也较少，因此较少的重构帧就能体现出原始序列的运动情况；运动剧烈的视频段含有较多的信息，因此使用较短GOP尺寸，从而解码端可以在很低的时间层次上重构更多的帧数据，尽可能的重现原始序列的运动过程。因此，在解码端重构层次较低时，相对于固定GOP方法，自适应的GOP更能体现出原始视频序列的运动过程，不至于出现很大的帧间跳动，从而视觉效果更流畅。

6.6 本章小结

本章分析了传统MCTF编码方案中固定GOP结构中存在的问题，并提出了一种类haar的MCTF编码方案，GOP尺寸根据帧间的互信息的变化自适应的选择，从而与视频序列中的运动性质保持一致，在选定的GOP内部，根据互信息值来选择低通帧，最后，根据选定的GOP结构，来确定时间上的分解层次和运动预测方向。实验结果表明，在运动性质变化较大的序列中，本文提出的GOP

尺寸方法可以在一定程度上提高压缩性能，同时，当解码端时间上重构的层次较少时，本文的方法恢复的视频序列较传统方法更能体现出原始视频序列的运动情况。另外，在运动较为剧烈的视频序列中，本文提出的低通帧的选择方法较之传统方法可以得到更好的压缩性能。

第7章 总结及后续工作展望

视频转码技术和可分级编码技术均为视频序列的复杂应用提供了相应的解决方案。视频转码技术可以根据网络特点和用户终端来确定输出比特流的格式，而且兼容不同的视频编码标准。相对于完全重新编码，视频转码的最大优势在于它可以充分解码信息，进而提高重新编码的速度。在可分级编码中，视频源只需要一次性编码最高分辨率下的比特流，用户端根据自身的性能接受部分比特流解码即可，从而减轻了编码端的负担。基于MCTF的视频可分级编码中，完全抛弃了迭代编码方式，因此可以避免“漂移”效应。针对基于H.264的视频转码技术和基于MCTF的可分级编码技术，本文主要在以下几个方面进行了深入的研究：

1. 帧内模式选择部分。该部分主要应用在空间分辨率转码方面。本文统计了原始图像中非零系数比例（nz_per），与设定的阈值比较作为选择帧内宏块类型的准则。为了能使得设定的阈值适应不同的重新量化参数（Q_r），本文提出了一个$Th_I_Q_r$模型，该模型以指数曲线描述Q_r和nz_per阈值的关系。将该模型经过线性化处理，得到一元线性回归模型，然后利用最小二乘法估计模型中的参数。
2. $Th_I_Q_r$模型的初始参数可以通过大量的实验方法获取，但得到的是普遍模型，本文提出了一种在实际转码过程中对模型中的参数更新的方法，从而使得该模型能适应不同的视频序列。
3. 在使用$Th_I_Q_r$模型选择了帧内宏块类型（I4MB/I16MB）之后，不同的宏块类型对应不同的帧内预测模式，本文又提出了一种快速的帧内预测模式选择方法，该方法充分利用输入原始图像中宏块的类型和帧内预测模式，进而大幅度降低了当前宏块的帧内预测模式选择时间。
4. 在空间分辨率转码的帧间模式选择部分，本文利用nz_per划分出当前宏块所在区域的运动性质，从而跳过部分候选宏块类型的测试，并提出了$Th_P_Q_r$模型。与$Th_I_Q_r$模型类似，该模型同样使用指数曲线来描述Q_r和nz_per阈值的关系，并在实际转码过程中进行即时更新。
5. 本文将空间分辨率转码中的帧间模式选择方案推广到了时间分辨率转码，并取得了较好的效果。

6. 在帧间模式选择中，根据原始图像计算出来的运动矢量并非一定精确，尤其是当Q_r较大时。本文提出了一种新的运动矢量细化方案，该方案中以*nz_per*作为运动矢量细化步长的准则，且随着Q_r的增加，运动矢量细化步长也逐步增加。从而保证了在运动较为剧烈的区域，运动矢量细化步长较长，运动平缓的区域的具有较短的步长。
7. 本文首次提出基于分类方法在视频转码中快速选择宏块类型。利用该方法，本文首次完成了基于H.264的同时包行三个方面（空间、时间、质量）的转码方案。从输入比特流中提取特征向量，并将其输入到离线训练完毕的支持向量机模型，从而预测出目标宏块类型。在该方案中，可以同时选择三个方面的图像格式的改动（图像尺寸、帧率、重新量化参数）。
8. 基于互信息技术，结合实际视频序列中运行性质的变化，本文提出了一种GOP结构选择方案。该方案又包括GOP尺寸选择和低通帧选择两部分。本文同时利用GOP内平均互信息值和标准差来控制GOP尺寸，从而选定的GOP尺寸不仅能根据运动类型的变化自适应的改变，而且同一个GOP内部的运动类型也能保持一致。本文首次提出了一种低通帧的选择方案，该方案基于互信息技术，从一个GOP内提取出与其余帧最具相关性的帧。
9. 根据选定的GOP结构，本文提出了一种自适应的时间分解过程。该分解过程尽可能的使得每层帧对间的距离相差较小，进而降低运动预测参差。另外，该方案还可以与传统的MCTF方案保持兼容。

本文针对基于H.264的视频转码技术和基于MCTF的可分级编码技术进行了深入的研究，结合提出的各种方法的性能，认为以下几个方面还需要进一步的研究：

1. 在本文提出的Th_Q_r模型中，需要输入一些经验数据来估计数学模型中参数的初始值，如何修改现有的数学模型，从而避免输入经验数据，还需要做进一步的研究。
2. 很多研究人员已经提出了基于H.264的快速帧内/帧间模式选择的方法[23]-[26]，在下一步的研究中，可以探讨如何将这些选择方法应用到转码系统中。
3. 本文使用的支持向量机模型需要经过大量的视频序列进行离线训练，然后进行模型精简，在使用过程中并没有任何的更新。因此该模型是一个

具有普遍意义的模型，在后续研究中，可以进一步分析如何在使用中对支持向量机模型即时更新，使得它可以适应具体的视频序列。

4. 另外，本文利用大量的实验的方法对特征向量的选择进行了压缩性能的比较，在理论上探讨如何选择特征向量还需要进一步的研究。

5. 本文提出的基于分类方法的视频转码中宏块类型选择方案，仅限于帧内预测模式，不兼容I4MB和I16MB两种帧内预测模式，如何利用现有模型预测当前帧中I16MB和I4MB的使用还需要进一步的探讨。

6. 在本文提出的低通帧选择的方法中，需要计算GOP内各帧间的互信息值，计算量较大，如何减少计算量以及其他的低通帧选择方法也需要进行研究。

随着网络技术的飞速猛进，加上用户终端的千差万别，视频序列的应用环境复杂多变，一些视频标准[132]、[133]也对视频应用环境的变换有着不同程度的支持。相信视频转码技术和可分级编码技术作为两种解决方案会得到进一步的发展和研究。

参考文献

[1]. Shih-Fu Chang, Anthony vetro, "Video adaptation: concepts, technologies, and open issues," Proceeding of IEEE, vol. 93, pp. 148-158, Jan. 2005.

[2]. Niklas Bjork, Charilaos Christopoulo, "Transcoding architectures for video coding," IEEE Trans. Consumer Electronics, vol. 44, pp. 88–98, 1998.

[3]. Huifen Shen, Xiaoyan Sun, Feng Wu, Houqiang Li, Shipeng Li, "A fast downsizing video transcoding for H.264/AVC with rate-distortion optimal mode decision," IEEE International Conference on Multimedia and Expo (ICME 2006), pp. 2017-2020, Toronto, Canada, July, 2006.

[4]. Chi-Hung Li, Chung-Neng Wang, Tihao Chiang, "A fast downsizing video transcoding based on H.264/AVC standard," Springer Pacific Rim Conference on Multimedia (PCM 2004), pp. 215-223, Tokyo, Japan, Nov. 2004.

[5]. Kai-Tat Fung and Wan-Chi Siu, "Diversity and importance measures for video downscaling," IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005), vol. 2, pp. 1061-1064, Mar. 2005.

[6]. Bo Shen, Ishwar K. Sethi and Bhaskaran Vasudev, "Adaptive motion-vector resampling for compressed video downscaling," IEEE Trans. Circuits and system for video technology, vol. 9, pp. 929-936, Sept. 1999.

[7]. YongQing Liang, Lap-Pui Chau and Yap-Peng Tan, "Arbitrary downsizing video transcoding using fast motion vector reestimation," IEEE letters Signal Processing, vol. 9, pp. 352-355. Nov. 2002.

[8]. Yap-Peng Tan and Haiwei Sun, "Fast motion re-estimation for arbitrary downsizing video transcoding using h.264/AVC standard," IEEE Trans. Consumer Electronics, vol. 50, pp. 887-894, Aug. 2004.

[9]. Rajeev Kumar, Senior Member, IEEE, and Vasant Patil, "An Efficient Motion Vector Composition Scheme for Arbitrary Frame Down-Sampling Video Transcoding," IEEE Trans. Circuits and Systems for Video Technology, vol. 16, pp: 1148-1152, Sep. 2006.

[10]. Jun Xin, Ming-Ting Sun, byung-Sun Choi and Kang-Wook Chun, "An HDTV-to-SDTV spatial transcoding," IEEE Trans. Circuits and System for Video Technology, vol. 12, pp. 998-1008, Nov. 2002.

[11]. Kai-Tat Fung and Wan-Chi Siu, "DCT-based video downscaling transcoding using split and merge technique," IEEE Trans. Image Processing, vol. 15, pp. 394-403, Feb. 2006.

[12]. Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "New Architecture for Dynamic Frame-Skipping Transcoding," IEEE Trans. Image Processing, vol. 11, pp. 886-900, Aug. 2002.

[13]. Mei-Juan Chen, Ming-Chung Chu, Chih-Wei Pan, "Efficient motion-estimation algorithm for reduced frame-rate video transcoding," IEEE letters Circuits and System for Video Technology, vol. 12, pp. 269-275, 2002.

[14]. Tamer Shanableh and Mohammed Ghanbari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," IEEE Trans. Multimedia, vol. 2, pp. 101-110. Jun. 2000.

[15]. 杨高波, 余圣发, 张兆杨, "压缩域 H.264 视频转换编码及其关键技术分析," 通信学报, vol. 27, pp. 124-131, 2006.

[16]. Jae-Ho Hur, Hyouk-Kyun Kwon, Yung-Lyul Lee, " H.264/AVC baseline profile to MPEG-4 visual simple profile transcoding to reduce the spatial resolution," Wiley International Journal of Imaging System and Technology, vol. 16, pp. 24-33, 2006.

[17]. Gerardo Fernández-Escribano, Pedro Cuenca, Luís Orizco-Barbosa and Antonio Garrido, "A fast intra-frame prediction algorithm for MPEG-2/h.264 video transcodings," IEEE International Conference on Imaging Processing (ICIP 2005), vol. 3, pp. 684-687, Genova, Italy, Sept. 2005.

[18]. Sung-Eun Kim, Jong-Ki Han and Jae-Gon Kim, "Efficient motion estimation algorithm for MPEG-4 to h.264 transcoding," IEEE International Conference on Imaging Processing (ICIP 2005), vol. 3, pp. 656-659, Genova, Italy, Sept. 2005.

[19]. Ishfraq Ahmad, Xiaohui Wei, Yu Sun and Ya-Qin Zhang, "Video Transcoding: an overview of various techniques and research issues," IEEE Trans. Multimedia, vol. 7, pp.793-804. Oct. 2005.

[20]. Jun xin, Chia-Wen Lin and Ming-Ting Sun, "Digital video transcoding," invited paper, Proceeding of the IEEE, vol. 93, pp. 84-97, Jan. 2005.

[21]. Seung-Kyun Oh and Hyun Wook Park, "Analysis of IDCT and motion-compensation mismatches between spatial-domain and transform-domain motion-compensated coders," IEEE Trans. Circuit and System for Video Technology, vol. 15, pp. 835-843, Jul. 2005.

[22]. Damien Lefo, Dave Bull and Nishan Canagarajah, "Performance evaluation of transcoding algorithms for h.264," IEEE Trans. Consumer Electronics, vol. 52, pp. 215-222, Feb. 2006.

[23]. Tien-Ying Kuo and Chen-Hung Chan, "Fast variable block size motion estimation for h.264 using likelihood and correlation of motion field," IEEE Trans. Circuit and System for Video Technology, vol. 16, pp. 1185-1195, Oct. 2006.

[24]. Andy C. Yu, Ngan King Ngi and Graham R. Martinn, "Efficient intra- and inter-mode selection algorithms for h.264/AVC," Journal of Visual Communication and Image Representation, vol. 17, pp. 322-344, Aug. 2005.

[25]. 宋 彬,常义林,周宁兆, "基于 H.264 帧间预测的快速算法," 电子学报, vol. 34, pp. 31-34, 2006.

[26]. 李世平, 蒋刚毅, 郁梅, "快速帧内预测模式选择新方法," 电子学报, vol. 34, pp. 141-146, 2006.

[27]. Damien Lefol and Dave Bull, "Mode refinement algorithm for h.264 inter frame requantization," IEEE International Conference on Imagin Processing (ICIP 2006), pp. 845-848, Oct. 2006.

[28]. Jan De Cock, Stijn Notebasert, Peter Lambert, Davy De Schrijver and Rik Van de Walle, "Requantization transcoding in pixel and frequency domain for intra 16x16 in h.264/AVC," Advanced Concepts for Intelligent Vision Systems (ACIVS 2006), LNCS 4179, pp. 533-544, Belgium, Sept. 2006.

[29]. Stijn Notebasert, Jan De Cock, Koen De Wolf, and Rik Van de Walle, "Requantization transcoding of h.264/AVC bitstreams for intra 4x4 prediction modes," IEEE Pacific Rim Conference on Multimedia (PCM 2006), LNCS 4261, pp. 808-817, Hangzhou, China, Nov.2006.

[30]. Oliver Werner, "Requantization for transcoding of MPEG-2 intraframes," IEEE Trans.

Image Processing, vol. 8, pp. 179-191, Feb. 1999.

[31]. 修晓宇, 卓力, 沈兰荪, "一种基于 PID 控制器的 H.264 比特率转码方案," 电子学报, vol. 34, pp. 1062-1065, 2006.

[32]. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, JVT-K051, version 3 of h.264/AVC, 12th meeting: Redmond, WA, USA, 17-23 July, 2004.

[33]. Iain E. G. Richardson, "H.264 and MPEG-4 video compression – video coding for next-generation multimedia," John Wiley & Sons Ltd. The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England.

[34]. 王松桂, 陈敏, 陈立萍, "线性统计模型: 线性回归与方差分析," 高等教育出版社. 2004.

[35]. Eric Dubios, and Shaker Sabri, "Noise Reduction in Image Sequences Using Motion-Compensated Temporal Filtering", , IEEE Trans. Communications, vol. COM-32, no.7, pp. 826–831, July 1984.

[36]. Jens-Rainer.Ohm, "Three-dimensional subband coding with motion compensation," IEEE Trans. Image Processing, vol. 3, pp. 559–571, Sept. 1994.

[37]. S.-J. Choi and J. W. Woods, "Motion-compensated 3-D subband coding of video," IEEE Trans. Image Processing, vol. 8, pp. 155–167, Feb. 1999.

[38]. L. Luo, J. Li, S. Li, Z. Zhuang, and Y.-Q. Zhang, "Motion compensated lifting wavelet and its application in video coding," IEEE International Conference on Multimedia and Expo (ICME 2001), pp. 365–368, Tokyo, Japan, Aug 2001.

[39]. B. Pesquet-Popescu and V. Bottreau, "Three-dimensional lifting schemes for motion-compensated video compression," IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001), pp. 1793–1796, Salt Lake City, USA, May 2001

[40]. A. Secker and D. Taubman, "Motion-compensated highly-scalable video compression using an adaptive 3D wavelet transform based on lifting," IEEE International Conference on Image Processing (ICIP 2001), vol. 2, pp. 1029–1032, Thessaloniki, Greece, Oct 2001.

[41]. Peisong Chen, and John W. Woods, "Bidirectional MC-EZBC With Lifting Implementation", IEEE Trans. Circuits and System for video technology, vol. 14, pp. 982-993, Oct 2004.

[42]. Shih-Ta Hsiang, John W. Woods, "Embedded video coding using invertible motion compensated 3-D subband/wavelet filter bank", Signal Processing: Image Communication, vol 16, pp. 705–724, 2001.

[43]. Yonghui Wang, Suxia Cui, and James E. Fowler, "3-D Video Coding With Redundant-Wavelet Multihypothesis", IEEE Trans. Circuits Syst. Video Technol., vol. 16, pp. 166–177, Feb. 2006.

[44]. Yiannis Andreopoulos, Adrian Munteanu, Joeri Barbarien, Mihaela Van der Schaar, Jan Cornelis, Peter Schelkens, "In-band motion compensated temporal filtering", Signal Processing: Image Communication, vol. 19, pp. 653–673, Aug. 2004

[45]. Xin Li, "Scalable video compression via overcomplete motion compensated wavelet coding", Signal Processing: Image Communication, vol. 19, pp. 637–651, Aug. 2004.

[46]. Riccardo Leonardi, and Jens-Rainer Ohm, "Wavelet Video Coding – an Overview", MPEG Workgroup Video Subgroup, ISO/IEC JTC1/SC29/WG11 W7824, Bangkok, Thailand, Jan. 2006.

[47]. Ching-Yeh Chen, Chao-Tsung Huang, Yi-Hau Chen, Shoa-Yi Chien, and Liang-Gee Chen, "System Analysis of VLSI Architecture for 5/3 and 1/3 Motion-Compensated Temporal Filtering", IEEE Trans. Image Processing, vol. 54, pp. 4004–4014, Oct 2006.

[48]. D.S. Turaga, M. van der Schaar, Y. Andreopoulos, A. Munteanu, , P. Schelkens, "Unconstrained motion compensated emporal filtering (UMCTF) for efficient and flexible interframe wavelet video coding," Signal Processing, Image Communication, pp:1–19, 2005.

[49]. Hwangjun Song, Jongwon Kim, C.-C. Jay Kuo, "Real-time encoding frame rate control for H.263+ video over the internet," Signal Processing: Image Communication, vol. 15, pp. 127–148, 1999.

[50]. Jungwoo Lee, Bradley W. Dickinson, "Temporally adaptive motion interpolation exploiting temporal masking in visual perception", IEEE Trans. Image Processing. vol. 3, pp. 513-526, Sept. 1994.

[51]. Lee J. ; Shin I ; Park H, "Adaptive Intra-Frame Assignment and Bit-rate Estimation for Variable GOP Length in H.264," IEEE Trans. Circuits and Systems for Video Technology, vol, 16, pp. 1271-1279, Oct. 2006.

[52]. Yu-Lin Wang, Jing-Xin Wang, yen-Wen Lai, Alvin W. Y Su, "Dynamic GOP structure determination for real-time MEPG-4 advanced simple profile video encoder" IEEE Internationl Conference on Multimedia and Expo. (ICME 2005), pp: 293-296, Amsterdam, Netherlands, July, 2005.

[53]. L. Wang, "Rate control for MPEG video coding," Signal processing: Image communication, vol. 15, pp. 493-511, 2000.

[54]. Peisong Chen, "Fully scalable subband/wavelet coding," Doctoral Thesis, Rensselaer Polytechnic Institute Troy, New York. May, 2003.

[55]. Zuzana Cerneková, Ioannis Pitas, Christophoros Nikou, "Information Theory-Based Shot Cut/Fade Detection and Video Summarization," IEEE Trans. Circuits and System for Video Technology, vol. 16, pp. 82–91, Jan 2006.

[56]. Wengang Cheng, Yaniing Liu, De Xu, "Shot boundary detection based on the knowledge of information theory," IEEE International Conference on Neural Networks and Signal Processing (ICNNSP 2003), vol. 2, pp. 1237-1241, Nanjing, China, Dec. 2003.

[57]. T Butz, JP Thiran, "Shot boundary detection with mutual information," IEEE International Conference on Image Processing (ICIP 2001), vol. 3, pp. 421-424, Thessaloniki, Greece, Oct. 2001.

[58]. Peisong Chen, Software package of MC-EZBC wavelet coder is publicly available at ftp://ftp.cipr.rpi.edu/personal/chen.

[59]. Christophe Tillier, Béatrice Pesquet-Popescu, and Mihaela van der Schaar, "3-Band Motion-Compensated Temporal Structures for Scalable Video Coding," IEEE Trans. Image Processing, vol. 15, pp. 2545–2557, Sep. 2006.

[60]. Deepak S. Turaga, Mihaela van der Schaar, and Beatrice Pesquet-Popescu, "Complexity Scalable Motion Compensated Wavelet Video Encoding," IEEE Trans. Circuits and System for Video Technology, vol. 15, pp. 982-993, Aug. 2005.

[61]. Jens-Rainer Ohm, "Advances in Scalable Video Coding," Proceedings of the IEEE, vol. 93, Issue 1, pp. 42-56, Jan. 2005.

[62]. Hendrik Eeckhaut, Harald Devos, Benjamin Schrauwen, Mark Christiaens, Dirk Stroobandt,

"A Hardware-Friendly Wavelet Entropy Codec for Scalable Video," IEEE Design, Automation and Test in Europe Conference and Exhibition (DATE'05), vol. 3, pp. 14-19, 2005

[63]. Jeongnam Youn, Ming-Ting Sun, and Chia-Wen Lin, "Motion vector refinement for high-performance transcoding," IEEE Trans. Multimedia, vol. 1, pp. 30-40, Mar. 1999.

[64]. Mei-Juan Chen, Ming-Chung Chu and Chih-Wei Pan, "Efficient motion-estimation algorithm for reduced frame-rate video transcoding," IEEE letters Circuits and System for VideoTechnology, vol. 12, pp. 269-275. Apr. 2002.

[65]. Haiyan Shu and Lap-Pui Chau, "The Realization of Arbitrary Downsizing Video Transcoding," IEEE Trans. Circuits and Systems for Video Technology, vol. 16, pp. 540-546, Apr. 2006.

[66]. Young Seo Park and Hyun Wook Park, "Arbitrary-Ratio Image Resizing Using Fast DCT of Composite Length for DCT-Based Transcoding," IEEE Trans. Imaging Processing, vol. 15, pp. 494-500, Feb. 2006.

[67]. Haiyan Shu, and Lap-Pui Chau, "An Efficient Arbitrary Downsizing Algorithm for Video Transcoding," IEEE Trans. Circuits and Systems for Video Technology, vol.14, pp. 887-891, Jun. 2004.

[68]. T. Shanableh and M. Ghanbari, "Heterogeneous video transcoding to lower spatial-temporal resolutions and different encoding formats," IEEE Trans. Multimedia, vol. 2, no. 2, pp. 101–110, Jun. 2000.

[69]. H. Sun, W. Kwok, and J. W. Zdepski, "Architectures for MPEG compressed bitstream scaling," IEEE Trans. Circuits and Systems for Video Technology, vol. 6, no. 2, pp. 191–199, Apr. 1996.

[70]. Heiko Schwarz, Detlev Marpe, and Thomas Wiegand, "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard," IEEE Trans. Circuits Systems and Technology, vol. 17, no. 9, pp. 1103-1120, Setp. 2007.

[71]. G.H. Park, M.W. Park, S Jeong, J. Cha, K. Kim, and J. Hong, "Adaptive GOP structure for SVC," ISO/IEC/JTC1/SC29/WG11/MPEG/ M11563, Hong Kong, Jan. 2005.

[72]. G.H. Park, M.W. Park, S. Jeong, K. Kim, and J. Hong, "Improve SVC coding efficiency by adaptive GOP structure (SVC CE2)," Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) JVT-O018, Korea, Apr. 2005.

[73]. M.W. Park, G.H. Park, S. Jeong, Doug-Young Suh, and K. Kim, "Adaptive GOP Structure for Joint Scalable Video Coding," IEICE Trans. Communication, vol. E90–B, no.2, Feb. 2007.

[74]. Tuanjie Qian, Jun Sun, Dian Li, Xiaokang Yang, and Jia Wang, "Transform Domain Transcoding From MPEG-2 to H.264 With Interpolation Drift-Error Compensation," IEEE Trans. Circuits and systems for video technology, vol. 16, pp: 523-534, Apr. 2006.

[75]. Jun Xin, Anthony Vetro, Huifang Sun, and Yeping Su, "Efficient MPEG-2 to H.264/AVC Transcoding of Intra-Coded Video," Journal on Advances in Signal Processing, vol. 2007, Article ID 75310, 12 pages, 2007.

[76]. Damien Lefol, David Bull, Nishan Canagarajah, David Redmill, "An efficient complexity-scalable video transcoding with mode refinement," Signal Processing: Image Communication, vol. 22, pp. 421-433, 2007.

[77]. Yung-Ki Lee, Seong-Seon Lee, and Yung-Lyul Lee, "MPEG-4 to H.264 transcoding with

frame rate reduction," Springer Multimedia Tools and Application, vol. 35, pp. 147-162. Nov. 2007.

[78]. Hari Kalva, Branko Petljanski, "Exploiting the Directional Features in MPEG-2 for H.264 Intra Transcoding," IEEE Trans. Consumer Electronics, vol. 52, pp. 706-711, May 2006.

[79]. Jens Bialkowski, Marcus Barkowsky, André Kaup, "Fast video transcoding from H.263 to H.264/MPEG-4 AVC," Springer Multimedia Tools and Application, vol. 35, pp. 127-146, Nov. 2007.

[80]. Yi Liu, Yuan F. Zheng, "Soft SVM and Its Application in Video-Object Extraction," IEEE Trans. Signal Processing, vol. 55, pp. 3272-3282, Jul. 2007.

[81]. Jinhui Yuan, Huiyi Wang, Lan Xiao, Wujie Zheng, Jianmin Li, Fuzong Lin, and Bo Zhang, "A Formal Study of Shot Boundary Detection," IEEE Trans. Circuits and Systems for Video Technology, vol.17, pp. 168-186, Feb. 2007.

[82]. Jie Zhou, Senior, Dashan Gao, and David Zhang, "Moving Vehicle Detection for Automatic Traffic Monitoring," IEEE Trans. Vehicular Technology, vol. 56, Jan. 2007.

[83]. T. Downs, K. Gates, and A. Masters1, "Exact simplification of support vector solutions," Journal of Machine Learning Research, vol. 2, pp. 293-297, Dec. 2001.

[84]. Nguyen, D., TuBao Ho, "A bottom-up method for simplifying support vector solutions," IEEE Trans. Neural Networks, vol. 17, pp. 792-796, May 2006.

[85]. S. Sathiya Keerthi, Olivier Chapelle, Dennis DeCoste, "Building Support Vector Machines with Reduced Classifier Complexity," Journal of Machine Learning Research, vol. 7, pp. 1493-1515, Jul., 2006.

[86]. T.-F. Wu, C.-J. Lin, and R. C. Weng, "Probability estimates for multi-class classification by pairwise coupling," Journal of Machine Learning Research, vol.5, pp. 975-1005, Aug. 2004.

[87]. Qing Tao, Gao-Wei Wu, Fei-Yue Wang, Jue Wang, "Posterior probability support vector Machines for unbalanced data," IEEE Trans. Neural Networks, vol. 16, pp. 1561-1573, 2005.

[88]. GÖnen M.; TanuĞur A. G; Alpaydın E, "Multiclass Posterior Probability Support Vector Machines," IEEE Trans. Neural Networks, Accepted for future publication.

[89]. Chih-Chung Chang, Chih-Jen Lin. LIBSVM: a Library for Support Vector Machines. http: //www.csie.ntu.edu.tw/-cjlin/libsvm/. 2007.

[90]. Tihao Chiang and Ya-Qin Zhang, "A new rate control shceme using quadratic rate distortion model," IEEE Trans. Circuits and Systems for Video Technology, vol. 7, pp. 246-250, Feb. 1997.

[91]. P. Zhang, Y. Lu, Q. Huang, W. Gao, "Mode mapping method for h.264/AVC spatial downscaling transcoding," IEEE International Conference on Imaing Processing (ICIP 2004), vol. 4, pp. 2781-2784, Singapore, Oct. 2004.

[92]. June-Sok Lee, Goo-Rak Kwon, Jae-Won Kim, Nam-Hyeong Kim and Sung-Jea Ko, "An effective motion vector re-estimation method for low bit-rate video transcoding," Journal of Real-time Imaging, vol. 10, pp. 325-329, Oct. 2004.

[93]. Peng Zhang, Qing-Ming Huang and Wen Gao, "Key techniques of bit rate of reduction for h.264 streams," Springer Pacific Rim Conference on Multimedia (PCM 2004), LNCS 3332, pp. 985-992, Tokyo, Japan, Nov. 2004.

[94]. Haiwei Sun, Yap-peng Tan and YongQing Liang, "Fast motion vector and bitrate

re-estimation for arbitrary downsizing video transcoding," IEEE International Symposium on Circuits and Systems (ISCAS 2003), vol. 2, pp. 856-859, Bangkok, Thailand, May 2003.

[95]. Chi-Wang Ho, Oscar C. Au, S.-H. Gary Chan, Hoi-Ming Wong and Shun-Kei Yip, "Improved refinement search for h.263 to h.264/AVC transcoding based on minimum cost tendency search," IEEE International Symposium on Circuits and Systems (ISCAS 2006), pp. 5275-5278, Island of Kos, Greece, May 2006.

[96]. Guobin Shen, Bing Zeng, Ya-Qin Zhang and Ming L. Liou, "Transcoding with arbitrarily resizing capability," IEEE International Symposium on Circuits and Systems (ISCAS 2001), vol. 5, pp. 25-28, Sydney, Australia, 2001.

[97]. Xiaoan Lu, Alexis Michael Tourapis, Peng Yin and Jill Boyce, "Fast mode decision and motion estimation for h.264 with a focus on MPEG-2/H.264 transcoding," IEEE International Symposium on Circuits and Systems (ISCAS 2005), pp. 1246-1249, Kobe, Japan, May 2005.

[98]. Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "Low-complexity and high-quality frame-skipping transcoding for continuous presence multipoint video conferencing," IEEE Trans. Multimedia, vol. 6, pp. 31-46, Feb. 2004.

[99]. Kai-Tat fung and Wan-Chi Siu, "Conversion between DCT coefficients and IT coefficients in the compressed domain for h.263 to h.264 video transcoding," IEEE (ICIP 2005), vol. 3, pp. 57-60, Genova, Italy, Sept. 2005.

[100]. Bo Shen, "Submacroblock Motion Compensation for Fast Down-Scale Transcoding of Compressed Video," IEEE Trans. Circuits and Systems for Video Technology, vol. 15, pp. 1291-1302, Oct. 2005.

[101]. Shi-Fu Chang and David G Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," IEEE Trans. Selected Areas in Communications, vol. 13, pp. 1-11, Jan. 1995.

[102]. Chen Chen, Ping-Hao Wu and Homer Chen, "Transform-domain intra prediction for h.264," IEEE International Symposium on Circuits and Systems (ISCAS 2005), vol. 2, pp. 1497-1500, Kobe Japan, May 2005.

[103]. Dongdong Zhu, Qionghai Dai and Rong Ding, "Fast inter prediction mode decision for h.264," IEEE International Conference on Multimedia and Expo, vol. 2, pp. 1123,-1126, Taipei, Taiwan, Jun. 2004.

[104]. Zhi Zhou, Jun Xin and Ming-Ting Sun, "Fast motion estimation and inter-mode decision for h.264/MPEG-4 AVC encoding," Journal of Visual Communication and Image Representation, vol. 17, pp. 243-263, Jul. 2005.

[105]. Dong-Gyu Sim, Yongmin Kim, "Context-adaptive mode selection for intra-block coding in h.264/MPEG-4 part 10," Journal of Real-time Imaging, vol. 11, pp. 1-6, Feb. 2005.

[106]. Jens Bialkowski, Marcus Barkowsky and André Kaup, "On requantization in intra-frame video transcoding with different transform block size," IEEE 7th workshop multimedia signal processing, pp. 1-4, Oct. 2005.

[107]. Hani Sorial, William E.Lynch and André Vincent, "Selective requantization for transcoding of MPEG compressed video," IEEE International Conference on Multimedia and Expo (ICME 2000), vol. 1, pp. 217-220, New York City, USA, Aug. 2000.

[108]. Junji tajime, Yuzo Senda and yoshihiro Miyamoto, "Fast software MPEG-2 video transcoding with optimization of requantization error compensation," IEEE International

Conference on Imaging Processing (ICIP 2002), vol. 1 pp. 705-708, New York, USA, Sept. 2002.

[109]. Bo Shen, "Optimal requantization-based rate adaptation for h.264," IEEE International Conference on Multimedia and Expo (ICME 2006), pp. 317-320, Toronto, Canada, July 2006.

[110]. P. A. A. Assuncao and M. Ghanbari, "Transcoding of single-layer MPEG video into lower rates," IEE Proceedings: Vision, Image and Signal Processing, vol. 144, pp. 377-383, 1997.

[111]. Chunrong Zhang, Shibao Zheng, Chi Yuan, Feng Wang, "A Novel low-complexity and high-performance frame-skipping transcoding in DCT domain," IEEE Trans. Consumer Electronics, vol. 51, pp. 1306-131. Nov. 2005.

[112]. Wei Jen Lee, and Wen Jen Ho, "Adaptive frame-skipping for video transcoding," IEEE International Conference on Imaging Processing, vol. 1, pp: 165-168. Barcelona, Catalonia, Spain, Sept. 2003.

[113]. Yusuf, A.A.; Murshed, M.; Dooley, L.S, "An adaptive motion vector composition algorithm for frame skipping video transcoding," IEEE Electrotechnical Conference, 2004. MELECON 2004. Proceedings of the 12th IEEE Mediterranean, vol. 1, p: 235-238. May 2004.

[114]. Siyoung Yang; Donghyung Kim; Yeonggyun Jeon; Jechang Jeong, "An efficient motion re-estimation algorithm for frame-skipping video transcoding," IEEE International Conference on Imaging Processing (ICIP 2005), vol. 3, pp: 668-671, Genova, Italy, Sept. 2005.

[115]. Changsung Kim, and C.-C. Jay Kuo, "Feature-based intra/inter coding mode selection for h.264/AVC," IEEE Trans. on Circuits and Systems for Video Technology, vol. 16, pp. 1-13. Nov. 2006.

[116]. D. Wu, F. Pan, K. P. Lim, S. Wu, Z. G. Li, X. Lin, S. Rahardja, and C. C. Ko, "Fast intermode decision in h.264/AVC video coding," IEEE Trans. Circuits and Systems for Video Technology, vol. 15, pp. 953-958, Jul. 2005.

[117]. D. Wu, F. Pan, K. P. Lim, S. Wu, Z. G. Li, X. Lin, S. Rahardja, and C. C. Ko, "Fast intermode decision in h.264/AVC video coding," IEEE Trans. Circuits and Systems for Video Technology, vol. 15, pp. 953-958, Jul. 2005.

[118]. Haiyan Shu, Lap-Pui Chau, "Intra/Inter Macroblock Mode Decision for Error-Resilient Transcoding," IEEE Trans. Multimedia, vol. 10, pp. 97-104, Jan. 2008.

[119]. Wan-Chi Siu, Yui-Lam Chan, and Kai-Tat Fung, "On Transcoding a B-Frame to a P-Frame in the Compressed Domain," IEEE Trans. Multimedia, vol. 9, pp. 1093-1102, Oct. 2007.

[120]. Haiyan Shu and Lap-Pui Chau, "A Resizing Algorithm With Two-Stage Realization for DCT-Based Transcoding," IEEE Trans. Circuits and Systems for Video Technology, vol. 17, pp. 248-253. Feb. 2007.

[121]. Haruhisa Kato, Yasuhiro Takishima, and Yasuyuki Nakajima, "A Fast DV to MPEG-4 Transcoding Integrated With Resolution Conversion and Quantization," IEEE Trans. Circuits and Systems for Video Technology, vol. 17, pp. 111-119. Jan. 2007.

[122]. Carlos Salazar and Trac D. Tran, "A Complexity Scalable Universal DCT Domain Image Resizing Algorithm, IEEE Trans. Circuits and Systems for Video Technology, vol. 17, pp. 495-499, Apr. 2007.

[123]. Vasant Patil, Rajeev Kumar, and Jayanta Mukherjee, "A Fast Arbitrary Factor Video

Resizing Algorithm," IEEE Trans. Circuits and Systems for Video Technology, vol. 16, pp. 1164-1171, Sep. 2006.

[124]. Ping-Hao Wu, Chen Chen, and Homer H. Chen, "Rounding Mismatch Between Spatial-Domain and Transform-Domain Video Codecs," IEEE Trans. Circuits and Systems for Video Technology, vol. 16, pp. 1286-1293, Oct. 2006.

[125]. Bo Shen, "Perfect requantization for video transcoding," Springer Multimedia Tools and Application, vol. 35, pp. 163-173. Nov. 2007.

[126]. Jun Xin, Jianjun Li, Anthony Vetro, and Shun-ichi Sekiguchi, "Motion mapping and mode decision for MPEG-2 to H.264/AVC transcoding," Springer Multimedia Tools and Application, vol. 35, pp. 163-173. Nov. 2007.

[127]. Zhijun Lei, Nicolas D. Georganas, "Adaptive video transcoding and streaming over wireless channels," The Journal of Systems and Software, vol. 75, pp. 253-270, 2005.

[128]. Yap-Peng Tan, Yongqing Liang, Haiwei Sun , "On the methods and performances of rational downsizing video transcoding," Signal Processing: Image Communication, vol. 19, pp. 47-65, 2004.

[129]. Zhijun Lei, Nicolas D. Georganas, "A rate adaptation transcoding scheme for real-time video transmission over wireless channels," Signal Processing: Image Communication, vol. 18, pp. 641-658, 2003.

[130]. Ben Fei and Jinbai Liu, "Binary Tree of SVM: A New Fast Multiclass Training and Classification Algorithm," IEEE Trans. Neural Networks, vol. 17, pp. 696-704, May 2006.

[131]. Hyunsoo Kim, Peg Howland, Haesun Park, "Dimension Reduction in Text Classification with Support Vector Machines," Journal of Machine Learning Research, vol. 6, pp. 37-53, Jan. 2005.

[132]. "MPEG-7 Overview v.9," Int. Standards Org./Int. Electrotech.Comm. (ISO/IEC) JTC 1, ISO/IEC JTC1/SC29/WG11N5525, Mar. 2003.

[133]. "MPEG-21 Overview v.5," Int. Standards Org./Int. Electrotech.Comm. (ISO/IEC) JTC 1, ISO/IEC JTC1/SC29/WG11/N5231, Oct. 2002.

[134]. J. Song and B.-L. Yeo, "A fast algorithm for DCT-domain inverse motion compensation based on shared information in a macroblock," IEEE Trans. Circuits Syst. Video Technol., vol. 10, no. 5, pp. 767–775, Aug. 2000.

[135]. C.-W. Lin and Y.-R. Lee, "Fast algorithms for DCT-domain video transcoding," in Proc. IEEE Int. Conf. Image Processing, vol. 1, 2001, pp. 421–424.

致 谢

首先要感谢导师彭玉华教授对作者的悉心指导，在作者攻读博士学位期间，导师彭玉华教授在工作，科研，生活上都给予了大量的帮助和指导。导师渊博的学识，严谨的学风，平易近人的风格给作者留下了极为深刻的印象，也对作者的科研方法，处事生活态度等各方面产生了巨大影响。作者衷心祝愿彭玉华教授身体健康，工作顺利！

作者在这里还要感谢香港理工大学多媒体信号处理中心的萧允治教授，对作者在香港工作学习期间，萧教授在工作和科研上给予了细致入微的指导，使得作者受益匪浅。另外，作者还要感谢实验室的杨明强，孙文红，曲怀敬，韩民，杨阳，孙伟峰，刘云霞，万洪林等各位老师和同学的帮助。实验室融洽的气氛给作者影响至深，作者也感谢实验室全体老师和同学对作者的各种帮助。

作者还要感谢父母的支持，是他们给了我刻苦的精神和勤奋的性格，使得我可以顺利完成博士学业。作者还要特别感谢妻子郭敏，她的支持才使得我能考取并顺利完成博士学业。

最后，作者以此文作为生日礼物送给我可爱的儿子刘砚清，祝他即将到来的两周岁生日快乐！愿他健康成长，生活幸福！

刘兆广

攻读学位期间发表和投稿的论文

1. 第一作者：An adaptive GOP structure selection for haar-like MCTF encoding based on mutual information, Springer international journal of Multimedia Tools and Application. (修回版本已提交). (SCI，EI 检索).
2. 第一作者：Adaptive HAAR-Like MCTF Based Wavelet Video Coding Scheme，IEEE 15th international conference on Digital Signal Processing 2007, July, 2007.
3. 第一作者：一种基于类 haar 小波的 MCTF 视频编码方案, 电子学报，2008 年第二期. (EI 检索号：081511196418)
4. 第一作者： 一种基于 h.264 的视频转码方案, 电子学报. 2008 年第五期. (EI 全文检索)
5. 第一作者：一种基于支持向量机的 H.264 视频转码中帧内宏块类型快速选择方案, 电子与信息学报. (修回中). (EI 源刊)
6. 第一作者：Intra mode selection in downsizing video transcoding based on H.264, Wiley international journal of Imaging Systems and Technology. (审稿中). (SCI，EI 检索.)
7. 第一作者：Fast intra macroblock mode decision for H.264 video transcoding based on support vector machine，IEE electronics letters. (审稿中). (SCI，EI 检索.)
8. 第一作者：一种基于 H.264 的帧率转码中的宏块类型选择方案, 通信学报. (审稿中). (EI 源刊)
9. 第三作者：A Fast Algorithm for YCbCr to RGB Conversion, IEEE Transactions on Consumer Electronics, Nov. 2007. (SCI，EI 检索)
10. 第三作者：基于 OMAP 平台的 AVS 解码实现，电子设计应用，2006 年第四期.

在读期间参加的项目及职责

1. 山大海信研究院：AVS视频解码技术及DSP移植的项目
2. 山东省科技攻关项目 2005 年（第三批）项目编号：2005GG3201117
3. 香港理工大学：Video Transcoding Techniquies and Strategies for Advanced Video Coding
4. 青岛第四十一电子研究所：非平稳信号时频分析技术研究

外文论文一

Intra mode selection in downsizing video transcoding based on H.264

Liu Zhaoguang, Peng Yuhua, Yang Yang

(School of Information Science and Engineering, Shandong University, Jinan 250100, China)

[Abstract]

In downsizing video transcoding based on H.264, a common topic is how to select macroblock type in downsized frame. An intra mode selection method is proposed in this paper, which supports downsize transcoding and re-quantization transcoding simultaneously. In the proposed method, a threshold about the total non-zero coefficients of responding four macroblocks in pre-coded frame is used to select intra macroblock type. To calculate this threshold which related to re-quantization parameter (called Q_r in this paper), we propose a Th_I-Q_r model including two methods to calculate the threshold, and called direct method and percentage I16MB method, respectively. In the direct method, an exponent model is proposed to describe the relationship between the threshold and Q_r; In the percentage I16MB method, the threshold is converted into the calculation of the percentage of macroblocks with I16MB mode in downsized frame (called *per*_16 in this paper), and the relationship between *per*_16 and Q_r is also modeled as an exponent function. The two exponent models are all converted into linear regression model, and Least Square Estimation is used to estimate the parameters in the models. Furthermore, if I4MB is selected, the intra prediction modes in pre-coded frame are utilized to select prediction mode for 4x4 blocks in downsized frame to save computational complexity. We compared the rate distortion performance and time cost of proposed method with the full search algorithm. The results of simulations demonstrate that the proposed method can attain a time of saving up to 30% and 80% in total encoding time and find mode time, respectively. At the same time, the compression performance of proposed method is close to the results of the full search algorithm.

[Keywords] video transcoding, H.264, macroblock selection, linear regression

1. Introduction

Video sequences are often used in different application environments, ranging from transmittion channel, storage media and display terminals. Transcoding is one of technologies to meet up these applications. Video sequences were encoded in high resolutions, and the target low resolutions to suit for special application can be converted directly in transcoding. In homogenous transcoding, there are mainly three transcoding types, including downsizing transcoding [1]~[8], frame rate transcoding [9]~[10], and re-quantization transcoding [12]~[13]. In this paper, we will talk about downsizing transcoding and re-quantization transcoding.

Downsizing transcoding includes integer downsize factor [1]~[4], and arbitrary downsize factor [5]~[7]. The downsizing transcoding with integer factor (usually equals to 2) is relatively simple. One of the topics of downsizing transcoding is motion vector compose. For the situation of downsize factor equals to two, which means there are four motion vectors in pre-coded frame, the problem is how to find the best motion vector in downsized frame. There are many methods were proposed to solve this problem, including random-choose-one [1], DCmax [15], majority [10], average [1], median [1], [10], etc; If the downsize factor is an arbitrary value, one motion vector in downsized frame is responding to several motion vectors in pre-coded frame, and the commonly used methods to compose these motion vectors are weighted median [6], [8], weighted average [4], [5], [8], etc. Another topic in downsizing transcoding is block mode selection [4], [3] in standards which allows different block modes, e.g. H.264. In fact, the motion vectors and block mode are associated with each other. Reference [3] utilized motion vectors and residual data in pre-coded frame and determined motion vector and block mode in one scheme.

In transcoding, if the target frame size and frame rate are not changed, re-quantization can also be used to achieve low bitrate [11]~[13]. The block mode used in higher quantization parameter is different to the one in lower quantization parameter. In reference [11], the bit length of pre-coded macroblock was used as the criterion to determine the macroblock mode in re-quantization process. Another topic in re-quantization transcoding is how to compensate re-quantization error, which will cause drift effect. References [12] introduced a mode-dependent matrix to compensate re-quantization error.

According to data operation domain, transcoding can also be classified into pixel domain transcoding and transform domain transcoding. In reference [16], mismatch between motion compensation in spatial domain and motion compensation in transform domain is analyzed and a lift constant is proposed to compensate the mismatch.

The Intra-prediction technique, which utilizes the spatial residual correlation between adjacent macroblocks/blocks, is recognized to be one of the main factors that contribute to the success of H.264. The difference of mode selection between pure encoder [17]~[19] and re-encoder in transcoding [20] is that there are many useful information can be utilized in transcoding, including residual data, macroblock type, etc. As far as authors' knowledge, there has not been any paper discussing about intra-frame mode selection in downsizing transcoding based on H.264. In this paper, we propose an intra macroblock mode selection method in downsizing and re-quantization transcoding in pixel domain based on H.264, and downsize factor is equal to 2 in the paper. In the proposed method, the input bitstream should be full decoded. After downsizing, decoded information is utilized to speed up re-encoding process in transcoding.

The rest of the paper is organized as follows. The intra macroblock mode of H.264 is reviewed in section 2. The proposed $Th\_I\text{-}Q_r$ model which is adopted to classify I4MB and I16MB is introduced in section 3. Method to select prediction direction in I4MB and I16MB is discussed in section 4. Experimental results are provided in section 5. Finally, conclusions and future works will be discussed in section 6.

2. Intra macroblock mode

In H.264, block size of luminance component can be predicted as 4x4 (I4MB), 8x8 (I8x8) or 16x16 (I16MB). I8MB is only used in FREXT (Fidelity Range Extensions), and it is not discussed in this report.

In I4MB macroblock mode, the whole macroblock is divided into sixteen 4x4 blocks, and every 4x4 block has one prediction mode. Figure 1 shows one 4x4 block (a, b, c, d, e,..., m, n, o, p) and its adjacent pixels (X, A, B, ..., K, L), and Figure 2 shows different prediction modes for 4x4 block.

X	A	B	C	D	E	F	G	H
I	a	b	c	d				
J	e	f	g	h				
K	i	j	k	l				
L	m	n	o	p				

Fig. 1 4×4 block and its adjacent pixels

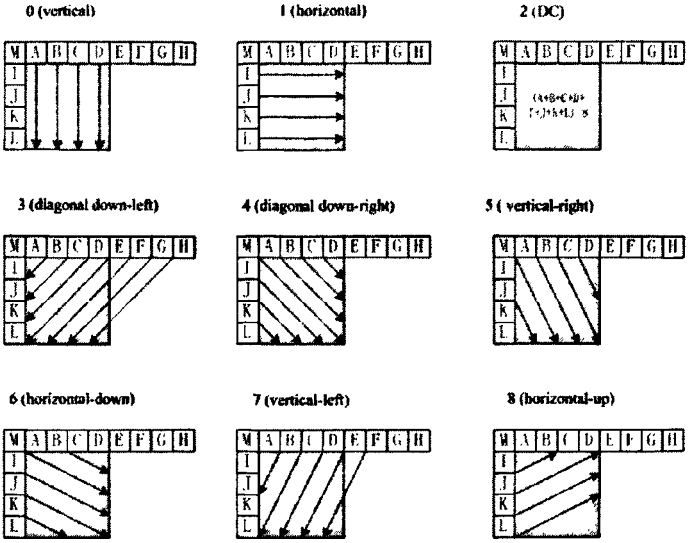

0 (vertical) 1 (horizontal) 2 (DC)
3 (diagonal down-left) 4 (diagonal down-right) 5 (vertical-right)
6 (horizontal-down) 7 (vertical-left) 8 (horizontal-up)

Fig. 2 4×4 Luma prediction modes

The whole macroblock will be predicted with one prediction mode in I16MB mode. There are four 16×16 prediction modes are used, which are mode 0(vertical), mode 1(horizontal), mode 2(DC) and PLANE.

The macroblock mode of chrominance components is similar as I16MB mode, which includes mode 0(vertical), mode 1(horizontal), mode 2(DC) and PLANE. The difference is the block size of chrominance is 8x8 in baseline profile of H.264. The detail discuss about intra prediction mode can be seen in reference [21], [22].

3. $Th\_I\text{-}Q_r$ model

In fact, there are two parts in the intra macroblock mode selection, macroblock type selection and prediction direction selection. Firstly, I4MB or I16MB is determined to be used, and prediction direction is selected in the following process. The proposed $Th\_I\text{-}Q_r$ model is used to classify I16MB and I4MB in the first part.

3.1 Introduce of the model

For each macroblock to select mode in downsized frame, there are responding four macroblocks in pre-coded frame (when downsize factor is 2). Let us consider the total number of non-zero coefficients of these responding four macroblocks in pre-coded frame, which will be called '*num_nz*' in this paper. Obviously, the value of *num_nz* is larger than zero and smaller than 1024 (32×32), and it is the indicator of residual data of pre-coded frame. The high value of *num_nz* is responding to great residual energy, and vice versa.

There are two quantization parameters in transcoding, QP in pre-coded frame (called Q_i) and QP in downsized frame (called Q_r). In the following experiments, $Q_i = 20$. and after downsizing, different Q_r (25, 30, 35, 40, 45) is used. The rate-distortion optimization method employed in H.264 (called full search algorithm in this paper) is used to find the best intra macroblock mode. Figure 3 shows the relationship between the average value of *num_nz* and macroblock type in downsized frame. The horizontal coordinate of Figure 3 is QP (Q_r) used in re-encoding process while the vertical coordinate shows the average value of *num_nz* in pre-coded frame. For every fixed Q_r, the blue bars represent the average value of *num_nz* for all macroblocks using I16MB mode in downsized frame, and the red bar represents I4MB mode. From Figure 3, we can see that the macroblock using I16MB in downsized frame corresponds to smaller *num_nz* value (smaller residual data), and I4MB macroblock has larger *num_nz* value.

The experimental results in this paper are the average of many commonly used video sequences, except for special declare.

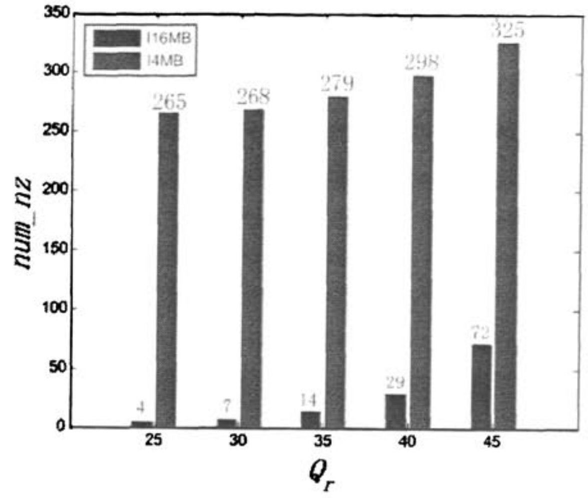


Fig. 3 The relationship between *num_nz* and Q_r for I16MB/I4MB

It is well known that the percentage of I16MB increases with Q_r. When Q_r is small, only macroblocks with very low *num_nz* are selected as I16MB. As the increasing of Q_r, more and more macroblocks with larger *num_nz* are selected as I16MB mode. Hence, the average value of *num_nz* for I16MB macroblock increases with Q_r. Meanwhile, as more and more macroblocks responding to small *num_nz* are selected as I16MB with the increase of Q_r, the average value of *num_nz* for left macroblocks which mode are I4MB will be higher too.

From this experiment results, we can conclude that when deciding the block mode of the current macroblock in downsized frame, the probability to use I16MB mode will be high if the value of responding *num_nz* is small. And if the value of responding *num_nz* is large, the probability to use I4MB is high. Let us explain this conclusion in another experiment, as shown in Figure 4. The horizontal coordinate is the value of *num_nz* (responding to every macroblock in downsized frame) in pre-coded frame, the vertical coordinate is the probability using I16MB for macroblock in downsized frame, and different Q_r (30, 40, 50) is represented by red, green and blue curves. When *num_nz* is small, the probability of using I16MB for current macroblocks is very high, and this probability will decrease with the increase of *num_nz*. On the other hand, with the increase of Q_r, the macroblock with the same value of *num_nz* has the higher probability to use I16MB. For example, for the macroblocks which the responding value of *num_nz* is 100, the probability to use I16MB mode is very low (be nearly to zero) when the Q_r is 30. This probability will increase to about 80 and 100 respectively when the Q_r is 40 and 50.

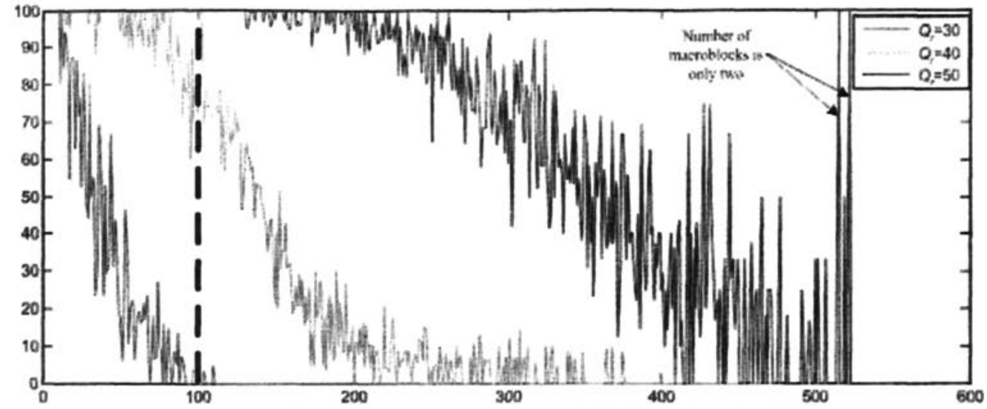


Fig. 4 Different Q_r and percentage I16MB type

From the previous analysis, we can select *num_nz* as the criterion to choose macroblock type. Further more, we select a threshold for *num_nz* to determine mode (called *Th_I* in this paper). The value of *num_nz* in pre-coded frame for current macroblock will be accounted and compared with *Th_I*. If *num_nz* is smaller than *Th_I*, the macroblock will be coded with I16MB type, otherwise use I4MB type.

In this paper, two methods to calculate this threshold, which called direct method and percentage I16MB method, are proposed and the discussion in detailed is made in section 3.2 and 3.3.

3.2 Direct Method

It is obviously that the value of *Th_I* should increase with Q_r. Our experimental results of some video sequences are shown in Figure 5 where Q_r are constant values [20, 25, 30, 35, 40, 45]. A fixed value of *Th_I* is selected for every constant Q_r, which can be seen in right part of figures. If the value of *num_nz* in pre-coded frame for current macroblock is lower than the fixed *Th_I*, the current macroblock will select I16MB directly; otherwise, I4MB will be used. The blue curve of responding left part is the rate-distortion curve of the selection of *Th_I* in right part. The red curve in left part is the rate-distortion curve using full search algorithm to find best macroblock type.

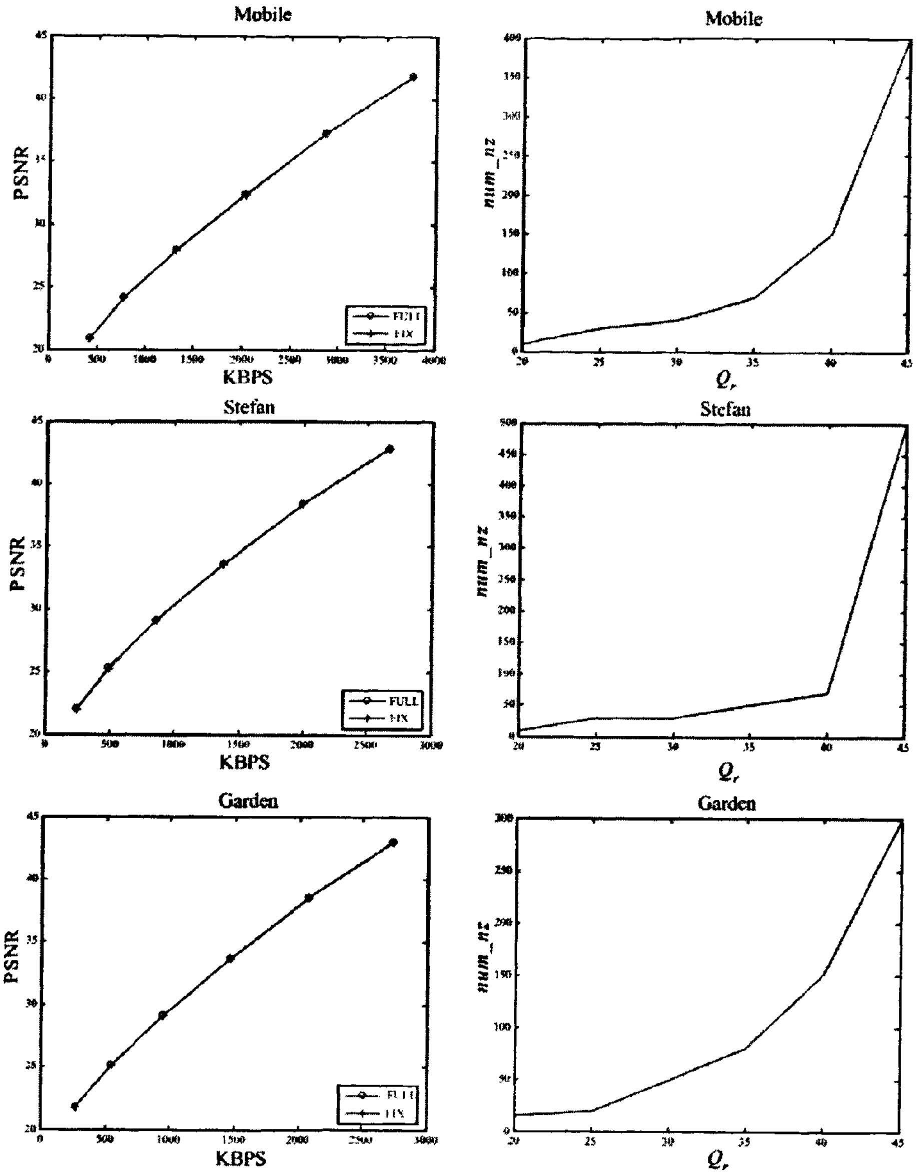


Fig. 5 Relationship between Q_r and Th_I

Under the condition that the compression performance is almost same as full search algorithm, and the relationship between Th_I and Q_r is nearly an exponent function. We use two parameters to represent this function, as shown in equation(1):

$$Th\_I = ae^{bQ_r} \tag{1}$$

From equation(1), we can obtain the following equation easily:

$$\ln Th\_I = \ln a + bQ_r \tag{2}$$

Let us replace Th_I and a using y and c respectively, just as following equations:

$$y = \ln Th\_I \tag{3}$$

$$\ln a = c \tag{4}$$

Hence, equation (2) can be written as linear regression model:

$$y = c + bQ_r \tag{5}$$

As is well known, Least Square Estimation can be used to estimate parameters in linear regression model:

$$b = \frac{n\sum_{i=1}^{n} Q_{ri}y_i - \sum_{i=1}^{n} Q_{ri}\sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} Q_{ri}^2 - (\sum_{i=1}^{n} Q_{ri})^2} \tag{6}$$

$$c = \frac{\sum_{i=1}^{n} y_i - b\sum_{i=1}^{n} Q_{ri}}{n} \tag{7}$$

Let us replace c and y using equation (3) and (4):

$$b = \frac{n\sum_{i=1}^{n} Q_{ri}\ln Th\_I_i - \sum_{i=1}^{n} Q_{ri}\sum_{i=1}^{n} \ln Th\_I_i}{n\sum_{i=1}^{n} Q_{ri}^2 - (\sum_{i=1}^{n} Q_{ri})^2} \tag{8}$$

$$a = \exp\left(\frac{\sum_{i=1}^{n} \ln Th\_I_i - b\sum_{i=1}^{n} Q_{ri}}{n}\right) \tag{9}$$

Where:

n is the number of frames.

Q_{ri} is the Q_r in ith frame.

Th_I_i is the threshold used in the ith frame.

Let us consider an example, Q_r = [20, 25, 30, 35, 40, 45], Th_I = [11, 23, 42, 77, 133, 345]. The estimated parameters are a = 0.7932 and b= 0.1320 using equation (8) and (9). The result is shown in the following figure, the red curve is the real Th_I-Q_r curve, and the green curve is the estimated result.

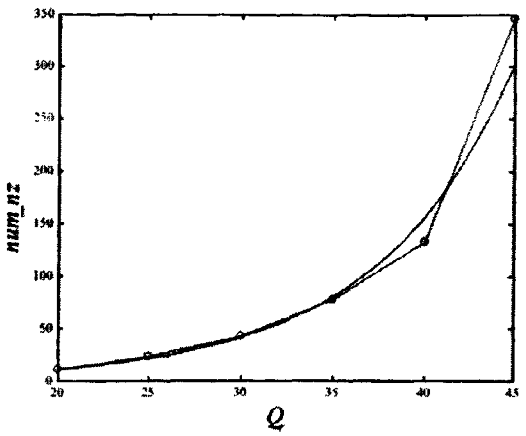


Fig.6 Estimation of parameters *a* and *b*.

Before the re-encoding process, transcoding does not know the best value of parameters for current video sequence. Hence, initial Th_I-Q_r set with expensive experiments are required to calculate initial value of *a* and *b* using equation (8) and (9), and the initial value of Th_I is calculated using equation (1). After the encoding of whole frame, *a* and *b* can be updated according to encoded results, and Th_I is also updated certainly. The following pseudo codes are the process of calculating and updating of parameters *a*, *b*, and Th_I:

```
a:  Input initial $Th\_I$- $Q_r$ set.
b:  calculate $a$ and $b$ using current $Th\_I$- $Q_r$ set using equation (8) and (9).
c:  calculate $Th\_I$ by $a$ and $b$ using equation (1).
d:  calculate mode refinement thresholds, $Th\_low$ and $Th\_high$.
        $Th\_low = 0.9 \times Th\_I$;
        $Th\_high = 1.1 \times Th\_I$;
e:  for(all macroblocks in downsized frame)
    {
      Account $num\_nz$ for current macroblock.
      if($num\_nz < Th\_low$)
        encode_I16MB();
      else if ($num\_nz > Th\_high$)
        encode_I4MB();
      else
      {
        Mode refinement between I4MB and I16MB;
        if(best mode is I4MB)
          num_4x4++;
        else
          num_16x16++
      }
    }
    if(num_16x16 > num_4x4)
      add $Th\_high$ and $Q_r$ into current $Th\_I$-$Q_r$ set;
    else
      add $Th\_low$ and $Q_r$ into current $Th\_I$-$Q_r$ set;

f:  go to b to re-encode next frame.
```

In this process, two mode refinement thresholds (*Th_low* and *Th_high*) are calculated. If the value of ***num_nz*** for current macroblock is lower than *Th_low*, I16MB mode is selected; If *num_nz* is larger than *Th_high*, I4MB is selected; Otherwise, I16MB and I4MB are all tested to select the best one, and the result is recorded ('num_4x4' or 'num_16x16' added by one). After the encoding of whole frame, if the num_16x16 is larger, which means current threshold is too low, the higher threshold (*Th_high*) will be added into *Th_I-Q_r* set; otherwise, the lower threshold (*Th_low*) is added into set. And the threshold for next frame will be updated before re-encoding next frame.

3.3 Percentage I16MB method

In this part, we will introduce another method to calculate threshold *Th_I*. Let *x* represents the percentage of macroblocks with I16MB in downsized frame. *Th_I* is the threshold we want to find to classify I4MB and I16MB. Let *M* represents the total number of macroblocks in downsized frame, and it is known for current transcoding. Hence, $x\times M$ represents the number of macroblocks with I16MB in downsized frame. From the previous section, we know that one macroblock in downsized frame is responding to one value of ***num_nz***. Hence, for every value of ***num_nz*** (0~1024), it is certainly responding to some macroblocks in downsized frame, just as shown in Figure 7. From zero, we accumulate the number of macroblock for every value of ***num_nz*** (just as the shadow area in Figure 7), and let ***n*** represents the accumulated value. If ***n*** is greater than $x\times M$, the responding value of *num_nz* will be the threshold *Th_I*.

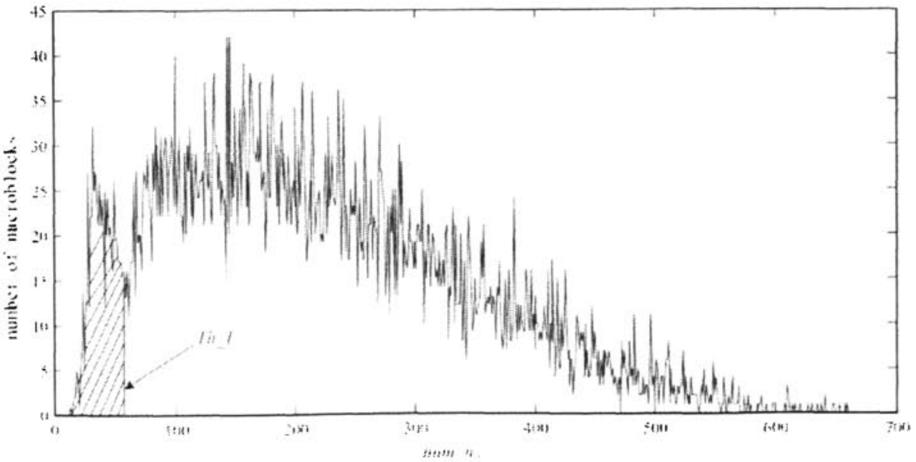


Fig.7 *num_nz* and responding number of macroblock in downsized frame

The relationship between *num_nz* and responding number of macroblock in downsized frame is known in transcoding which means the curve shown in Figure 7 is known. M is also known to transcoding. Hence, the unknown value of this method is only x. We know that the value of x (the percentage of macroblocks with I16MB) increases with the QP, as shown in Figure 8 (red curve). To fit the curve shown in Figure 8, we can use the following exponent function:

$$per\_16 = ae^{bQ_r} \tag{10}$$

Where:

per_16 is the percentage of macroblock with I16MB.

Hence, the estimation of parameters a and b are similar to the method one. The blue curve in Figure 8 is the fitted curve using the equation (10).

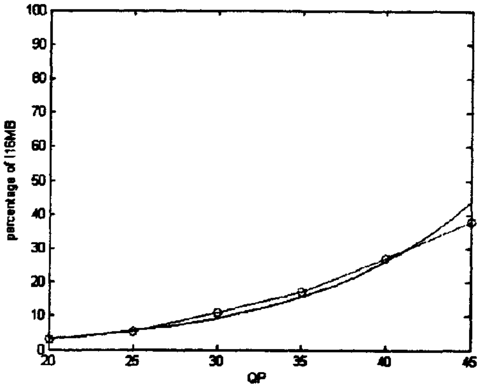


Fig.8 The percentage I16MB and QP

Before the re-encoding process, initial per_16-Q_r set with extensive experiments are required to calculate the initial value of a and b using equation (8) and (9), and the initial value of per_16 is calculated using equation (10). Furthermore, the responding Th_I is calculated using per_16.

The update of parameters in equation (10) is similar as section 3.2, and pseudo codes are no longer given here. The actual threshold used in the paper are $Th\_low = 0.9 \times Th\_I$, and $Th\_high = 1.1 \times Th\_I$. If the value of num_nz is lower than Th_low, I16MB will be selected; If the value of num_nz is greater than Th_high, I4MB will be selected; otherwise, mode will be refined between I16MB and I4MB, and the best one will be selected. When it comes to the end of encoding of current frame, the final result of percentage of macroblcoks with I16MB will be added to the current per_16-Q_r set. Hence, the value of Th_I is updated when the next frame begins to re-encode.

4. Prediction direction selection

After the selection of macroblock type, the following problem is how to select the prediction direction for I16MB and I4MB mode. If the selected macroblock type is

I16MB, there are four candidate intra prediction modes can be used. And if the selected macroblock type is I4MB, there are nine candidate modes can be used for every 4x4 block.

In this paper, if I16MB is selected, all four candidate modes will be tested with SAD (Sum of Absolute Difference) to select the one with the minimum SAD.

If the macroblock type is I4MB, the prediction mode should be selected for sixteen 4x4 blocks from nine candidate modes. As is well known, every 4x4 block in downsized frame is responding to four 4x4 blocks in pre-coded frame, and they are in I16MB or I4MB macroblock, as shown in Figure 9. In this example, the left sub-figure shows the responding four macroblocks in pre-coded frame, and the right sub-figure is the current macroblock in downsized frame. Only the left-up macroblock in pre-coded frame is I16MB in this example.

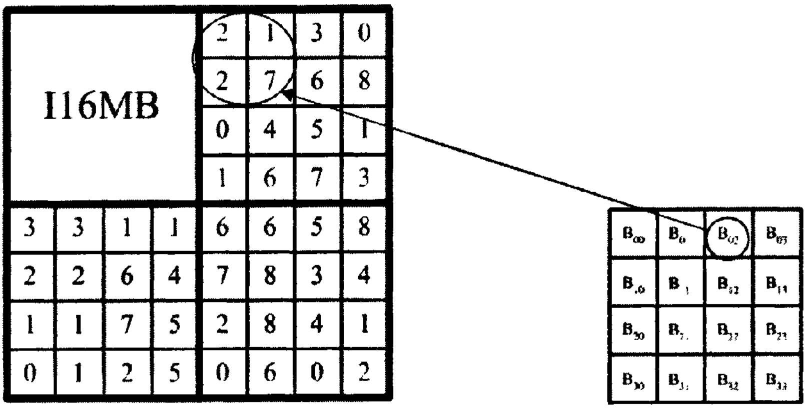


Fig. 9 Intra prediction modes

(1) Situation one: the responding four 4x4 blocks are in I16MB macroblock, e.g., B_{00}, B_{01}, B_{10}, and B_{11} in Figure 9. Figure 10 shows the probability of every prediction mode used in this situation. The horizontal coordinate of Figure 10 are nine candidate intra prediction modes, and vertical coordinate is the percentage of every prediction mode selected using full search algorithm. From the experimental result, we can see that the sum of the percentage of mode 0, 1, and 2 (about 87.5%) is much higher than other six modes. Hence, only mode 0, 1 and 2 will be tested with SAD in situation one and the mode with minimum SAD will be selected.

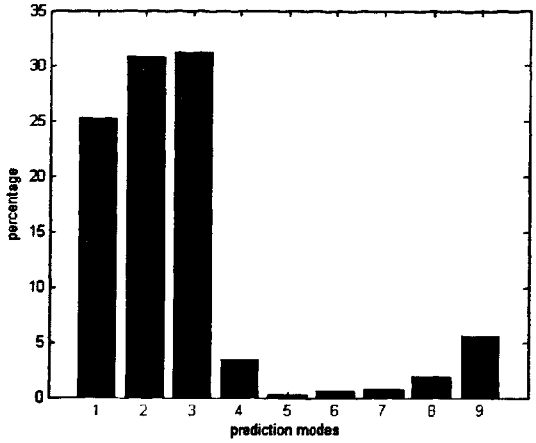


Fig. 10 Probability of every intra prediction mode in situation one

(2) Situation two: the responding four 4x4 blocks are in I4MB macroblock, e.g., B_{02}, B_{03}, B_{12}, and B_{13} in Figure 9. One 4x4 block in downsized frame is responding to four 4x4 blocks in pre-coded frame, which all have their prediction modes. For example, the responding four prediction modes of 4x4 block B_{02} is {2, 1, 2, 7}. Let us consider the probability of B_{02} to use one of these four modes. Figure 11 shows this probability between modes in four 4x4 blocks and current 4x4 blocks. The experimental results are obtained by full search algorithm. The left bar of every sub-figure means that the optimal prediction mode of B_{02} is one of the responding four prediction modes (Akiyo: 74.01%; Foreman: 65.07%). The right bar of sub-figure means that the optimal or sub-optimal prediction mode of B_{02} is one of the responding four prediction modes (Akiyo: 84.963%; Foreman: 81.15%). That means the prediction mode of B_{02} has a high probability to select one of the four responding prediction modes in pre-coded frame. Hence, we can only consider responding four prediction modes as candidate modes to save computing time. In this paper, only these four modes are tested with SAD and the mode with minimum SAD is selected as final intra prediction mode.

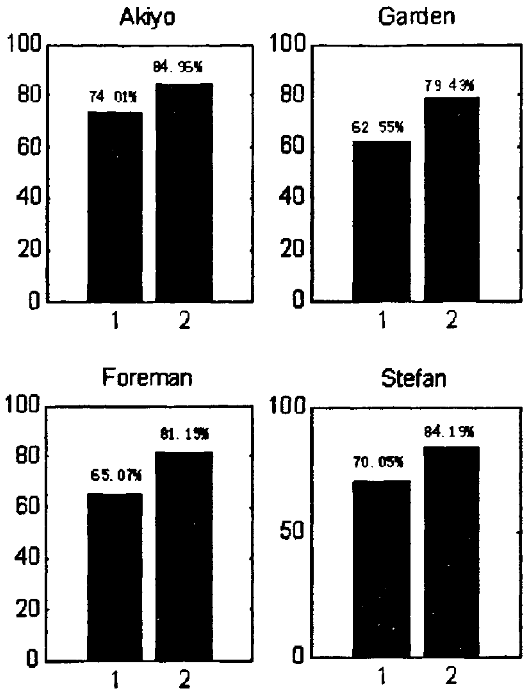


Fig. 11 Relationship between pre-coded modes and current mode

(3) A phenomenon in situation two. Figure 12 shows some results in situation two. The horizontal coordinate of figures is Q_r in re-encoding process, which values are {20, 25, 30, 35, 40, 45}. The vertical coordinate are percentages for all nine candidate prediction modes selected by full search algorithm, which is indicated by nine bars in figures for every Q_r. When Q_r increases, the prediction modes used in I4MB blocks will concentrate on mode 0, 1, and 2 (e.g., Q_r=20: 60.04%, Q_r=30: 65.16%, Q_r=40: 74.57%), just as shown in Figure 12. It means that, in situation two, the probability to use mode 0~2 is large when Q_r is high. In this paper, a hard threshold is set for Q_r. If Q_r is larger than this hard threshold, mode 0, 1, 2 will be added as candidate modes besides responding four prediction modes. The intra prediction mode will be selected from these candidate modes with the minimum SAD.

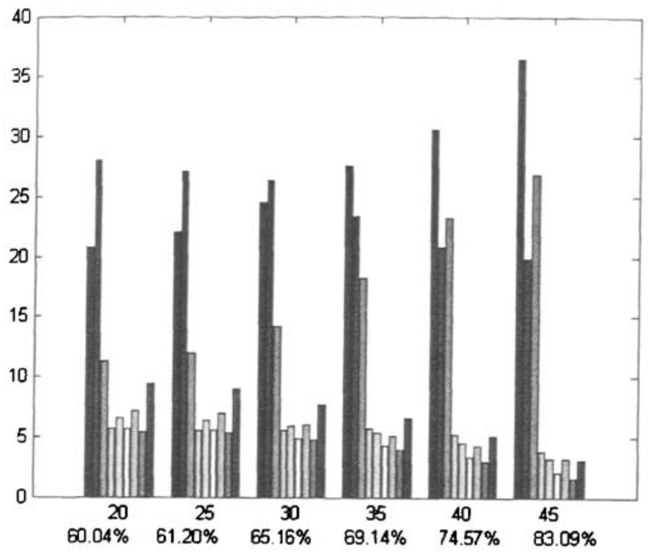


Fig. 12 Relationship between Q_r and percentage of mode 0, 1, and 2

5. Experimental results

In the following experiments, the frame size of input pre-coded frame is CIF (352×288), frame rate is 30fps, and the input QP is 20. All frames in test sequence are I-frames encoding with JM12.1. Symbol "Proposed_1" and "Proposed_2" correspond to the proposed direct method and percentage I16MB method to calculate threshold respectively, and "Full search" means the full search algorithm to find intra mode for every macroblock.

In the experimental results shown in Figure 13, the initial Th_I-Q_r set used in experiment is Q_r=[20, 25, 30, 35, 40, 45], Th_I = [11, 23, 42, 77, 133, 345]. The initial pe_16-Q_r set used in experiment is Q_r=[20, 25, 30, 35, 40, 45], per_16 = [2.97, 5.38, 10.91, 17.12, 26.93, 37.89]. The hard threshold discussed in section 4 is 30.

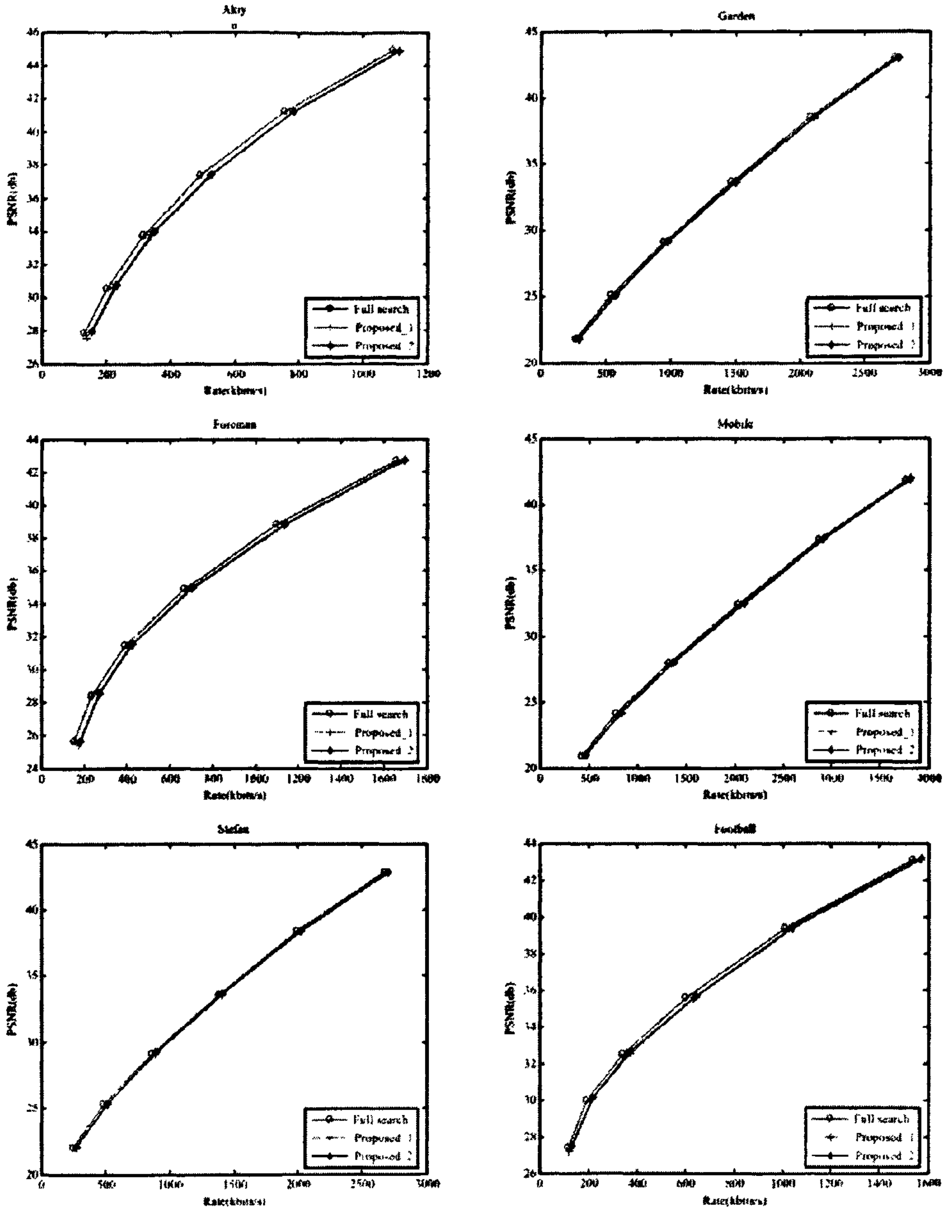


Fig. 13 Rate-distortion of several sequences

In the experimental results shown in Figure 14, the initial *Th_I*-Q_r set used in experiment is Q_r=[20, 25, 30, 35, 40, 45], *Th_I* = [8, 20, 35, 70, 150, 350]. The initial *pe*_16-Q_r set used in experiment is Q_r=[20, 25, 30, 35, 40, 45], *per_16* = [2.5, 5.0, 12.0, 19.0, 29.5, 42.8]. The hard threshold discussed in section 4 is 30.

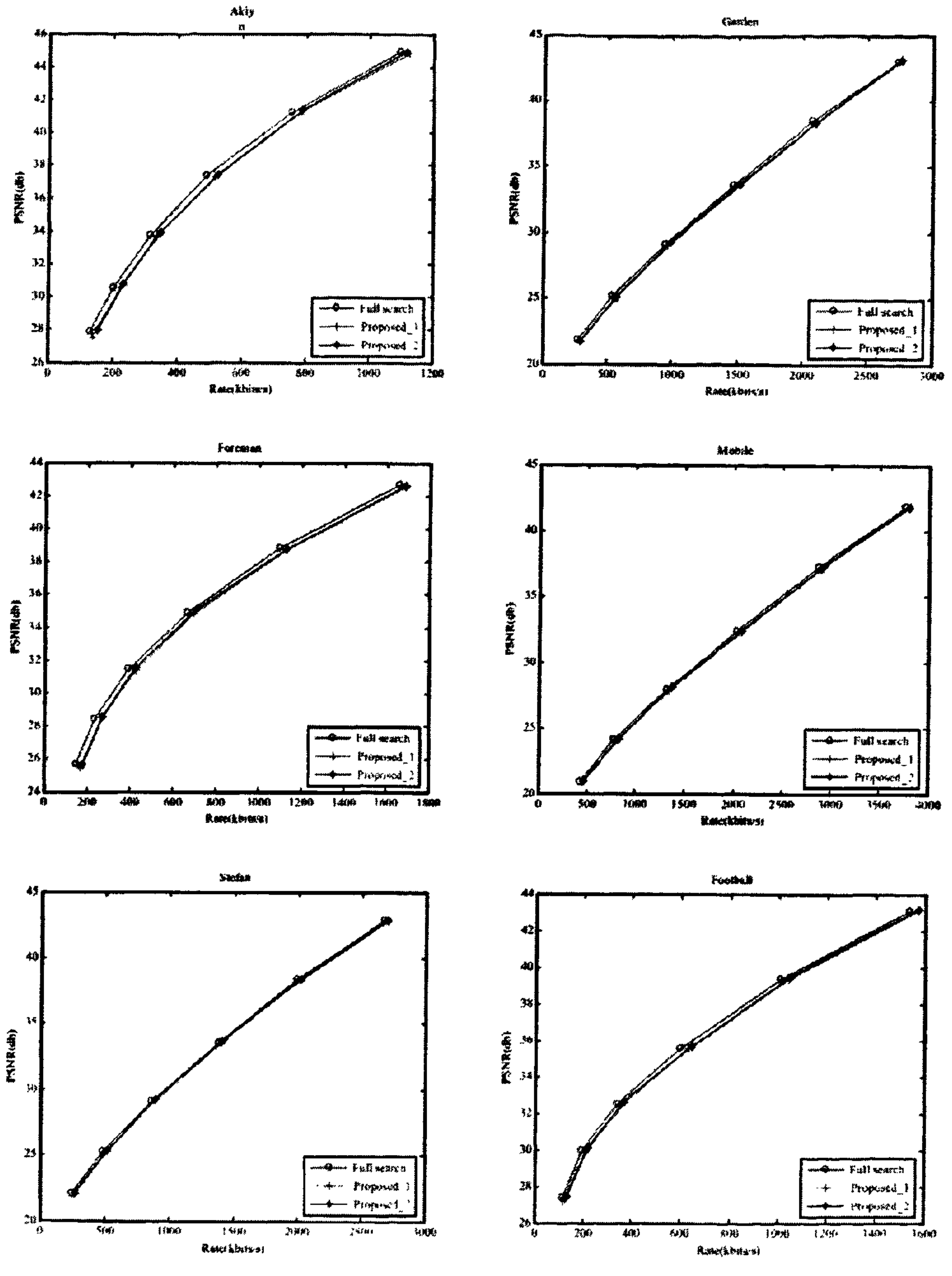


Fig. 14 Rate-distortion of several sequences

According to Figure 14, the proposed algorithm yields similar rate distortion performance compared with the full search intra-mode selection algorithm, and compression performance of the proposed two methods to calculate threshold are very close to each other. Because the update process (can be seen in section 0 and 0) of parameters in $Th\_I\text{-}Q_r$ model is introduced in the re-encoding process, the proposed two methods to calculate threshold are not sensitive to initial $Th\_I\text{-}Q_r$ set and $pe\_16\text{-}Q_r$ set according to the results of Figure 13 and Figure 14.

0 shows the computational complexity compared with the full search algorithm. There are three time-cost parts in transcoding, including decoding process, downsize-sampling process, and re-encoding process. Because the research topic of this paper is intra mode selection in re-encoding process, the time-cost of other two parts are not listed here. The ‘re-encoding time’ is the total re-encoding time, and ‘find mode time’ is time cost in intra macroblock mode selection. According to 0, the total encoding time saved by our method is 20%~30% and the time cost in mode selection can be saved by our method is 75%~80% comparing to full search algorithm. The computational complexities of two methods to calculate threshold are very close to each other.

Table 1 Comparison of time cost, between full search algorithm and proposed method

Sequence	re-encoding time (sec)			find mode time (sec)			frame number
	Proposed_1	Proposed_2	FULL	Proposed_1	Proposed_2	FULL	
Akiyo	3.518	3.533	5.624	0.478	0.480	2.445	300
Garden	3.957	4.060	5.877	0.519	0.540	2.189	250
Foreman	4.108	4.029	6.007	0.571	0.568	2.670	300
Mobile	5.905	5.811	7.630	0.650	0.680	2.846	300
Stefan	4.610	4.655	6.983	0.578	0.533	2.496	300
Football	1.165	1.174	1.797	0.185	0.197	0.847	90

To obtain better compression performance, a hard threshold is introduced in section 0. The experimental result of sequence ‘Akiyo’ is shown in Figure 15. The red rate distortion curve represents that no hard threshold is used, and blue curve corresponds that hard threshold is equal to 30. The method to calculate Th_I is the direct method.

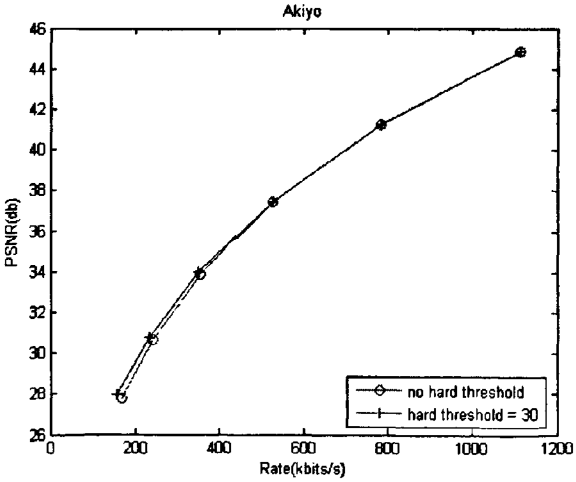


Fig. 15 The use of hard threshold

According to this result, the compression loss will be higher with the increase of Q_r, and the maximum value of loss is about 0.5 db. The situation without hard threshold is certainly can save computational complexity, which total encoding time is about 3.480 seconds (3.518 when hard threshold is 30) and find mode time is about 0.402 (0.478 when hard threshold is 30) seconds.

6. Conclusions and future work

An intra macroblock mode selection method in downsizing transcoding based on H.264 is proposed in this paper. The total number of non-zero coefficients of responding four macroblocks in pre-coded frame is used as the criterion to classify I4MB and I16MB in the proposed method. The *Th_I*-Q_r model is proposed in this paper to suit different video sequences and re-quantization parameters. In the proposed *Th_I*-Q_r model, two methods are proposed to calculate *Th_I*, called direct method and percentage I16MB method, respectively. In the calculation of *Th_I*, the proposed exponent models are converted into linear regression model, and Least Square Estimation is used to estimate the parameters in the linear regression model. The results of simulations demonstrate that the proposed method can attain a time of saving up to 30% and 80% in total encoding time and find mode time, respectively, compared with full search algorithm. And the rate distortion performance of proposed method is close to the full search algorithm.

In the proposed method, some initial data from extensive experiments are required. How to modify the proposed model to omit the input of initial data is a possible extension of current work. Many fast intra mode selection algorithms are proposed in pure encoder based on H.264 [17]~[19]. How to utilize the exiting algorithms in transcoding is also a fruitful direction for further research.

7. REFERENCES

[1]. Niklas Bjork, Charilaos Christopoulo, "Transcoding architectures for video coding," IEEE Trans. Consumer Electronics, vol: 44, pp:88–98, Feb. 1998.

[2]. Chi-Hung Li, Chung-Neng Wang, Tihao Chiang, "A fast downsizing video transcoding based on H.264/AVC standard," PCM, pp: 215-223, 2004.

[3]. Kai-Tat Fung and Wan-Chi Siu, "Diversity and importance measures for video downscaling," IEEE ICASSP, vol. 2, pp. 1061-1064, Mar. 2005.

[4]. Bo Shen, Ishwar K. Sethi and Bhaskaran Vasudev, "Adaptive motion-vector resampling for compressed video downscaling," IEEE Trans. Circuits and system for video technology, vol. 9, pp. 929-936, Sept. 1999.

[5]. YongQing Liang, Lap-Pui Chau and Yap-Peng Tan, "Arbitrary downsizing video transcoding using fast motion vector reestimation," IEEE letters signal processing, vol. 9, pp. 352-355. Nov. 2002.

[6]. Yap-Peng Tan and Haiwei Sun, "Fast motion re-estimation for arbitrary downsizing video transcoding using h.264/AVC standard," IEEE Trans. Consumer electronics, vol. 50, pp. 887-894, Aug. 2004.

[7]. Rajeev Kumar and Vasant Patil, "An efficient motion vector composition scheme for arbitrary frame

down-sampling video transcoding," IEEE Trans. Circuit and system for video technology, vol. 16, pp. 1148-1152, Sept. 2006.

[8]. Jun Xin, Ming-Ting Sun, byung-Sun Choi and Kang-Wook Chun, "An HDTV-to-SDTV spatial transcoding," IEEE Trans. Circuits and system for video technology, vol. 12, pp. 998-1008, Nov. 2002.

[9]. Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "New Architecture for Dynamic Frame-Skipping Transcoding," IEEE Trans. Image processing, vol. 11, pp. 886-900, Aug. 2002.

[10]. Tamer Shanableh and Mohammed Ghanbari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," IEEE Trans. Multimedia, vol. 2, pp. 101-110. Jun. 2000.

[11]. Damien Lefol, Dave Bull, "Mode refinement algorithm for H.264 inter frame requantization," IEEE ICIP, pp:845-848, 2006.

[12]. Jan De Cock, Stijn Notebasert, Peter Lambert, Davy De Schrijver and Rik Van de Walle, "Requantization transcoding in pixel and frequency domain for intra 16x16 in h.264/AVC," ACIVS 2006, LNCS 4179, pp. 533-544, Belgium, Sept. 2006.

[13]. Oliver Werner, "Requantization for transcoding of MPEG-2 intraframes," IEEE Trans. Image processing, vol. 8, pp. 179-191, Feb. 1999.

[14]. Jae-Ho Hur, Hyouk-Kyun Kwon, Yung-Lyul Lee, "H.264/AVC baseline profile to MPEG-4 visual simple profile transcoding to reduce the spatial resolution," Wiley International Journal of Imaging System and Technology, vol. 16, pp. 24-33, Jul 2006.

[15]. Ishfraq Ahmad, Xiaohui Wei, Yu Sun and Ya-Qin Zhang, "Video Transcoding: an overview of various techniques and research issues," IEEE Trans. Multimedia, vol. 7, pp.793-804. Oct. 2005.

[16]. Seung-Kyun Oh and Hyun Wook Park, "Analysis of IDCT and motion-compensation mismatches between spatial-domain and transform-domain motion-compensated coders," IEEE Trans. Circuit and system for video technology, vol. 15, pp. 835-843, Jul. 2005.

[17]. Andy C. Yu, Ngan King Ngi and Graham R. Martinn, "Efficient intra- and inter-mode selection algorithms for H.264/AVC," Science direction visual communication and image representation, vol. 17, pp. 322-344, Aug. 2005.

[18]. Changsung Kim, Hsuan-Huei Shih, C.-C. Jay Kuo, "Fast H.264 Intra-prediction mode selection using joint spatial and transform domain features," Science direct visual communication and image representation, vol:17, pp: 291-310, 2006.

[19]. Dong-Gyu Sim, Yongmin Kim, "Context-adaptive mode selection for intra-block coding in H.264/MPEG-4 Part 10," Science direct real time imaging, vol:11, pp: 1-6, 2005.

[20]. Hari Kalva and Branko Petljanski, "Exploiting the directional features in MPEG-2 for H.264 intra transcoding," IEEE Trans. Consumer electronics, vol. 52, pp. 706-711, May 2006.

[21]. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, JVT-K051, version 3 of H.264/AVC, 12th meeting: Redmond, WA, USA, 17-23 July, 2004.

[22]. Iain E. G. Richardson, "H.264 and MPEG-4 video compression – video coding for next-generation multimedia," John Wiley & Sons Ltd. The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England.

外文论文二

An adaptive GOP structure selection for haar-like MCTF encoding based on mutual information

Liu Zhaoguang, Peng Yuhua, Yang Yang

(School of Information Science and Engineering, Shandong University, Jinan 250100, China)

[Abstract]

In conventional motion compensated temporal filtering based wavelet coding scheme, where the group of picture structure and low-pass frame position are fixed, motion activities in video sequences are not considered. In this paper, we propose an adaptive group of picture structure selection scheme, in which the group of picture size and low-pass frame position are selected based on mutual information. Furthermore, the temporal decomposition process is determined adaptively according to the selected group of picture structure. A large amount of experimental work is carried out to compare the compression performance of proposed method with the conventional motion compensated temporal filtering encoding scheme and adaptive group of picture structure in standard scalable video coding model. The proposed low-pass frame selection can improve the compression quality by about 0.3-0.5db comparing to the conventional scheme in video sequences with high motion activities. In the scenes with un-even variation of motion activities, e.g. frequent shot cuts, the proposed adaptive group of picture size can achieve a better compression capability than conventional scheme. When comparing to adaptive group of picture in standard scalable video coding model, the proposed group of picture structure scheme can lead to about 0.2~0.8 db improvements in sequences with high motion activities or shot cut, especially abrupt shot cut.

[Keywords] Key Frame, Mutual Information, Motion Compensated Temporal Filtering, Group of Picture

1. Introduction

Video sequences are often used in different application environments, ranging from information transmission, storage media and display terminals. In order to satisfy different applications, different encoded bit streams are required. In Scalable Video Coding (SVC), the bit streams are coded in terms of different possible resolutions, and decoders can receive and decode different resolution bit streams for their specific applications.

The popular hybrid motion-compensated prediction and block transform scheme will cause the “drift” problem when the decoder receives the in-complete bit stream because of its recursive structure. The wavelet encoding scheme based on Motion Compensated Temporal Filtering (MCTF), which entirely abandons recursive structure, can provide high flexibility in bitstream scalability for different spatial, temporal and quality resolutions.

In fact, the use of MCTF was available more than 20 years ago [1]. Ohm [2] solved the unconnected/connected problem caused by motion estimation, while the temporal decomposition was explained by Quadrature Mirror Filters (QMF's). An approach to obtain low-pass and high-pass frames, and a rate-control scheme in the Group of Pictures (GOP) level were proposed in reference [3]. The introduction of lifting structure [4]-[6] in MCTF can resolve the un-complete reconstruction problem when the fractional pixel precision motion estimation is adopted. To encode the high-pass frames and low-pass frames more efficiently, Motion-Compensated Embedded Zeroblocks Coder (MC-EZBC) was proposed and developed in references [7], [8]. To save more bits in low bit rate coding, the scalable motion information encoding methods, e.g., Hierarchical Variable Size Block Matching (HVSBM) [3], were proposed. The MCTF in the wavelet transform domain was also developed in references [9]-[11], which allow better spatial scalability capability. Since the MCTF is one of the core technologies of the wavelet based scalable video coder [12], some hardware architecture design methodologies for MCTF, including computational complexity, external memory bandwidth, and external memory size, etc., were discussed in reference [13].

In the MCTF based wavelet encoding scheme, the GOP structure is fixed by temporal filter type and temporal layer. But in the real video sequence, the fixed GOP structure may cause that some successive frames with same motion type are divided into different GOPs, and frames with different motion types are included in one GOP. When the bitstream is decoded at the low temporal layer, there will be much variation of motion activities in the reconstructed video sequence. To achieve flexible encoding scheme, Unconstrained Motion Compensated Temporal Filtering (UMCTF) was proposed [14], in which some control parameters including GOP size, temporal decomposition level, reference number, high-pass frame number, and low-pass frame number etc. can be setup by user according to different video sequences. However, the paper did not address the ways for the determination of these parameters.

Because of variation of motion activities in video sequences, a variable GOP structure is necessary, and different methods have been proposed in coding MPEG-x or H.26x bit streams. The histogram of frame difference (HOD) was calculated in references [15]-[16] to measure the correlation between frames, and another commonly used method is difference of histogram (DOH) [15]. Because the DOH is not sensitive to local motion information, the block-based histogram method was introduced in references [16], [17]. Mean of absolute difference (MAD), sum of absolute difference (SAD), and variance of SAD of all macroblocks in the current frame and reference frame were calculated to determine the intra/inter encoding modes and GOP structure in reference [18]. The mean and variance of blocks were used in reference [19] for scene adaptive video encoding. In reference [15], the percentage of intra prediction macroblocks of the current frame was used as a criterion to determine the GOP size. In the

MCTF based wavelet video encoding scheme, adaptive GOP structures were used in references [7] and [20]. A threshold on the percentage of unconnected pixels at the current temporal level was used to determine whether to process motion compensation filtering in the next temporal level; hence an appropriate number of temporal decomposition levels were achieved. But this method cannot modify the GOP size according to the variation of motion activity in video sequence. An adaptive GOP size selection scheme was also proposed in standard SVC model [71]-[73]. A full GOP size is pre-determined in this scheme, e.g. 16, and pre-encodings are required with all sub-GOP sizes, i.e. 16, 8, 4, and 2, which is leading to expensive computational complexity.

In this paper, we proposed a method to select the GOP structure including GOP size and low-pass frame based on Mutual Information. The temporal decomposition process is then consequently modified according to the selected GOP structure.

Mutual Information (MI) is a measure of the information transported from one frame to another, which can detect the differences among successive frames. Hence, it can be used to detect shot boundaries and key-frame extraction [24]-[26]. In reference [24], MI was used to detect the shot cut, fade-in, and fade-out in video sequences, and retrieve key frames from the selected frame clusters. In reference [25], an automatic determination of the threshold based on MI was proposed for shot boundaries detection. The camera motion and zooming may decrease MI values and cause false detections of non-existing shot boundaries. In order to overcome these limitations, reference [26] maximized the MI with respect to a specified transformation model of the inverse of the non-desirable camera motion.

The remainder of the paper is organized as follows. In Section Ⅱ, the background of MCTF and MI will be discussed briefly. A haar-like MCTF encoding scheme, including the adaptive selection of GOP size, low-pass frame and temporal decomposition process is proposed in Section Ⅲ. Experimental results are presented in Section Ⅳ, and conclusions are drawn in Section Ⅴ.

2. Background Technology

2.1 Haar MCTF

The MCTF employs the open-loop structure. This approach can avoid the temporal recursive structure which causes the 'drift' effect in the decoding process. The original frames are filtered along the temporal direction with the motion trajectory as shown in Fig. 1. The low-pass frames and high-pass frames are transformed by 2-D spatial wavelet and coded by embedded coding, e.g. MC-EZBC [7]-[8].

As shown in Fig. 1, in the haar MCTF, pairs of frames are filtered using a two-channel haar filter-bank. For the connected pixels, the low-pass frames (L) and high-pass (H) frames are given in the lifting structure as shown below:

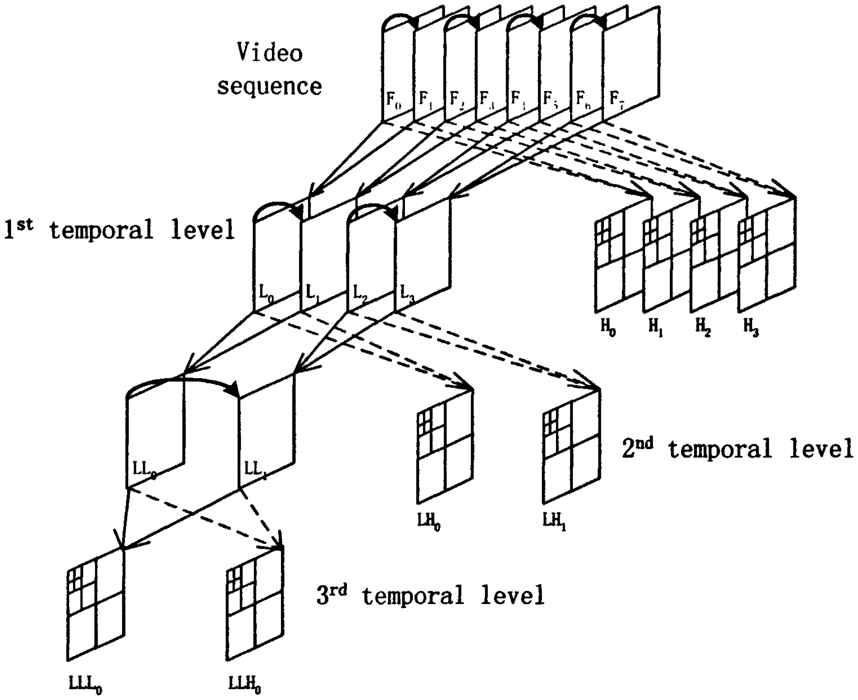


Fig. 1 Motion Compensated Temporal Filtering (MCTF)

$$H[m,n]=\frac{1}{\sqrt{2}}I_{2t+1}[m,n]-\frac{1}{\sqrt{2}}\tilde{I}_{2t}[m-d_m,n-d_n] \tag{1}$$

$$L[m-\bar{d}_m,n-\bar{d}_n]=\tilde{H}[m-\bar{d}_m+d_m,n-\bar{d}_n+d_n]+\sqrt{2}I_{2t}[m-\bar{d}_m,n-\bar{d}_n] \tag{2}$$

where $I_{2t+1}[m,n]$ represents the pixel value of position $[m,n]$ in frame I_{2t+1}, (d_m,d_n) is the motion vector of pixel $[m,n]$ in frame I_{2t+1}, $\tilde{I}_{2t+1}[m-d_m,n-d_n]$ is the interpolated value of pixel in $I_{2t}[m,n]$ when the fractional pixel motion compensation is adopted. $(\bar{d}_m,\bar{d}_n)$ is the inverse motion vector in frame I_{2t}, and the details about calculating it can be found in references [2]-[3], $\tilde{H}[m-\bar{d}_m+d_m,n-\bar{d}_n+d_n]$ is the interpolated value of pixels achieved by equation (1) in the high-pass frame. The process to calculate high-pass frame is also called 'prediction' and the process to calculate low-pass frame is called 'update'. Detail discussion can be found in [3], [7], and [20].

From Fig. 1, we can find the relationship between the GOP size and temporal decomposition levels:

$$n=2^i \quad (i=1,2,\ldots) \tag{3}$$

Where n is the GOP size, and i is temporal decomposition levels. In this fixed scheme, different types of motion activities are processed in the same way. There are two drawbacks in this arrangement. First, when the receiver decodes a bitstream in the lower temporal layer (frames in higher temporal layer should be discarded), it may find the much variations of motion activities in the decoded video because some high motion activity frames are discarded; While frames with low motion activity are reserved too long. Secondly, when the motion activity is high, the use of short GOP size can achieve better compression capability, while for frames with low motion activity, it is good to use long GOP size. So it is desirable allow an adaptive GOP size according to motion activity.

2.2 Mutual Information (MI)

MI refers to a measure of the information transporting between frames. It can be used to detect shot boundaries and key-frame extraction [20]. A large difference between frames (corresponding to the high motion activity) leads to a low MI value, while a small change between frames responds to a high MI value.

Let X be a discrete random variable with a set of possible outcomes $A_X = \{a_1,a_2,...,a_N\}$ with possibilities $\{p_1,p_2,...,p_N\}$, $p_X(x=a_i)=p_i$, $p_i \geqslant 0$, and $\sum_{x \in A_X} p_X(x) = 1$. According to the information theory, the entropy of X is:

$$H(X) = -\sum_{x \in A_X} p_X(x) \log p_X(x) \tag{4}$$

The joint entropy of variable X, Y is:

$$H(X,Y) = -\sum_{x,y \in A_X,A_Y} p_{XY}(x,y) \log(p_{XY}(x,y)) \tag{5}$$

The MI between random variable X and Y is given by:

$$I(X,Y) = -\sum_{x,y \in A_X,A_Y} p_{XY}(x,y) \log \frac{p_{XY}(x,y)}{p_X(x)p_Y(y)} \tag{6}$$

The relation between MI and joint entropy is given by:

$$I(X,Y) = H(X) + H(Y) - H(X,Y) \tag{7}$$

In a YUV formatted video sequence, MI of the luminance and chrominance components can be calculated respectively. Let us consider a gray level video sequence with intensity value ranging from 0 to $N-1$. For the luminance component, $P^Y_{t,t+1}(i,j)$ ($0 \le i,j \le N-1$) is the probability that a pixel with gray level i in frame F_t has a gray level j in frame F_{t+1}. So we can obtain the MI value of the luminance component and the total MI as shown below:

$$MI^Y_{t,t+1} = -\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} P^Y_{t,t+1}(i,j) \log \frac{P^Y_{t,t+1}(i,j)}{P^Y_t(i)P^Y_{t+1}(j)} \tag{8}$$

$$MI_{t,t+1} = MI^Y_{t,t+1} + MI^U_{t,t+1} + MI^V_{t,t+1} \tag{9}$$

The definition of chrominance components with probabilities $P^U_{t,t+1}(i,j)$ and $P^V_{t,t+1}(i,j)$,

$(0 \leq i, j \leq N-1)$, for calculating the MI values of $MI^{U}_{t,t+1}, MI^{V}_{t,t+1}$, in equation (9), are the same as that for luminance component.

It is well known that, the main energy of video sequence concentrates on the luminance component, so in our approach, the motion estimation is performed on luminance component, and motion vectors obtained are also used in chrominance components prediction.

In the following experimental work, Lagrange multiplier [3] was used to make a trade-off between the cost of motion vectors and the prediction errors. The goal of motion estimation is to achieve the minimum *COST* given below:

$$COST = \lambda R_{mv} + D_{pred} \tag{10}$$

Variable block size motion estimation was adopted in our experiment, and R_{mv} is the total bits required coding motion vectors, D_{pred} is the prediction error, and λ is lagrange multiplier which is selected as 24. Because of the relationship between the MI value and motion activity, the higher MI value corresponds to the lower *COST* between frames. To show the results more clearly, we use the reciprocal of *COST*: *COST* = 1/*COST*. The other commonly used methods for selecting GOP size DOH and HOD are calculated also in our experiment. All values are normalized for simplification and results are shown in Fig. 2.

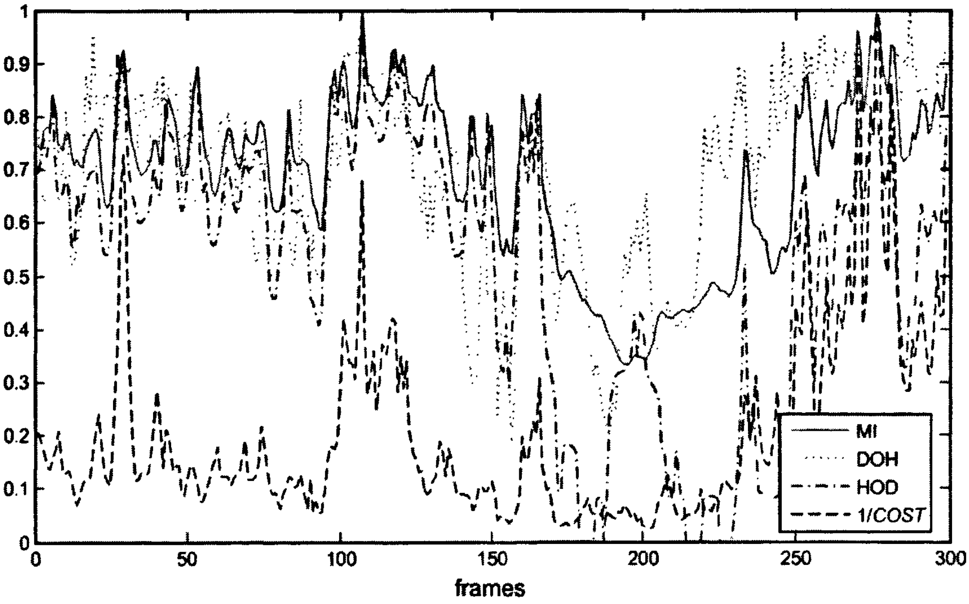
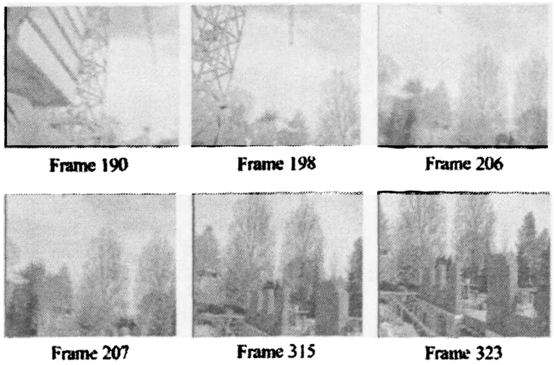


Fig. 2 Relationship between MI value, DOH, and HOD with motion estimation *COST*

We can see that when the motion activity is high, the MI value is low, and the motion estimation *COST* is high (1/*COST* is low). For example, in the position around frame number 200, the MI value is low and the inverse of *COST* is also low. Comparing to the original video sequence, we can find that the video in this segment is a shot transfer because of camera motion, see, Fig. 3, which leads to a high motion estimation *COST*. Hence, MI can be adopted as a criterion to measure motion activities between frames.



Frame 190 Frame 198 Frame 206

Frame 207 Frame 315 Frame 323

Fig. 3 Camera motion in foreman sequence

According to the experimental results shown in Fig. 2, DOH and HOD can't catch motion estimation curve as well as MI. Especially when the frame number is 200, there are peaks for DOH and HOD curves.

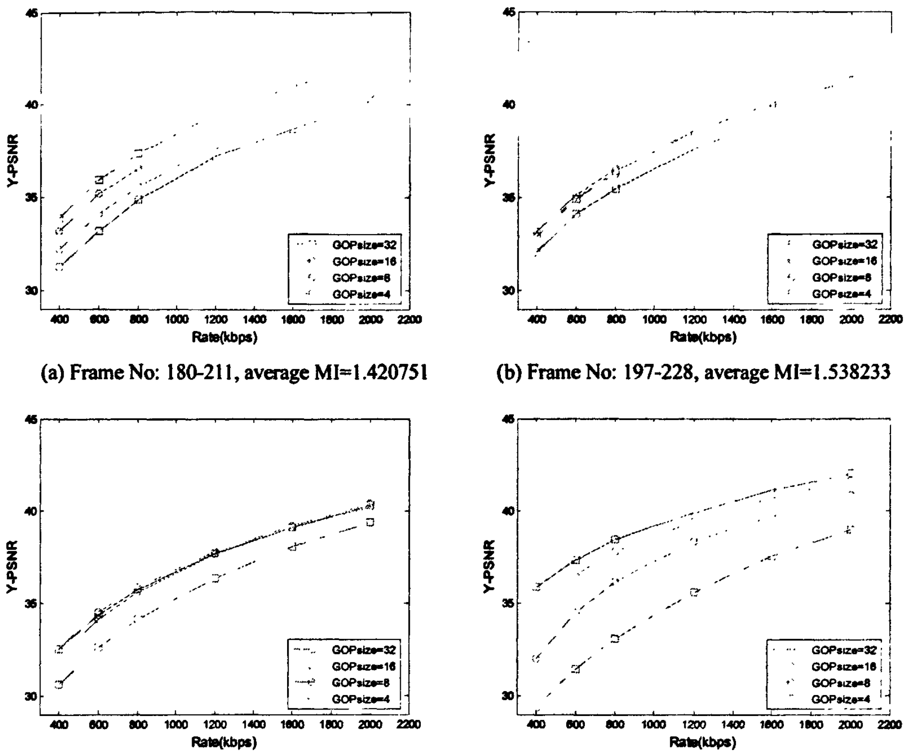
3. Haar-like MCTF Encoding Scheme

In our proposed haar-like MCTF encoding scheme, the GOP structure is selected adaptively based on MI which including GOP size selection and low-pass frame selection. Furthermore, the temporal decomposition process is determined according to the selected GOP structure.

3.1 GOP size selection

As analyzed in Section Ⅱ, high motion estimation *COST* leads to low MI value, and vice versa. So, we use the average MI value as one criterion to select GOP size. When the average MI value between frames is high, we can use a long GOP size, while a short GOP is used when the average MI value is low. More information is included in high motion activity frames. If the GOP size is short in this frame type, when the receiver decodes the low temporal layer, it can get more frames than a fixed GOP size. On the other hand, if the differences among frames are small, fewer frames are to be used.

From the following experimental results, we can find that the GOP size is related to compression capability in difference motion activities. Let us use sequence "Foreman" as an example, which is in CIF format. According to Fig. 4, when the average MI value between frames is low (frame No: 180-211), a short GOP size can achieve a higher compression efficiency, and when the average MI value is high (frame No:250-281), a long GOP size gives some better results.



(a) Frame No: 180-211, average MI=1.420751

(b) Frame No: 197-228, average MI=1.538233

(c) Frame No: 212-243, average MI=1.775050

(d) Frame No: 250-281, average MI=3.069295

Fig. 4 Compression performance with different video segments, MI values, and GOP size

When the MI values of successive frames vary greatly, it means that the motion activity in successive frames is heterogeneous, and these frames should be partitioned into different GOPs. In our approach, besides average MI, the standard deviation is adopted as another criterion to limit the variation of MI values in frames by a threshold *var_T*.

The utilization of average MI is implemented by three MI thresholds, low_MI, median_MI, and high_MI, and the relationship between the GOP size and these thresholds is given in Table 1.

Table 1 The relationship between average MI value and GOP size

average MI	GOP size
average_MI < low_MI	4
low_MI <= average_MI < median_MI	8
median_MI <= average_MI < high_MI	16
high_MI <= average_MI	32

The pseudo codes for adaptively selecting the GOP size is as follows:

```
a: initialization:
        n=0;
        set standard deviation threshold var_T;
        read the first frame data $F_0$;
```

```
b:  determine GOP size
        n++;
        read one frame data $F_n$;
        calculate MI value $MI_{n-1,n}$ of luminance component between $F_{n-1}$ and $F_n$;
        calculate average MI value of MI set: $\{MI_{0,1}, MI_{1,2}, \ldots, MI_{n-1,n}\}$, which given by:
        $average\_MI = \sum_{i=0}^{n-1} MI_{i,i+1} / n$
        and :
        if( (average_MI < low_MI) && (n >= 4) ) encode_GOP();
        else if( (low_MI <= average_MI < median _MI) && (n>=8) ) encode_GOP();
        else if((median_MI <= average_MI <high_MI) && (n>=16) encode_GOP();
        else if(average_MI >= high_MI) &&(n>=32)) encode_GOP();
        else
            calculate standard deviation $\sigma_n$ of $\{MI_{0,1}, MI_{1,2}, \ldots, MI_{n-1,n}\}$.
            if( $\sigma_n$ >= $var\_T$) encode_GOP();
            else     goto b:
```

where "encode_GOP()" is the process of encoding one GOP, MI value of luminance component "$MI_{n-1,n}$" can be calculated by equation (8).

In the proposed method, the average MI value and standard deviation are used simultaneously to determine the GOP size. The selected GOP size does not only adaptively change according to the motion activity, while also the motion activity within one GOP should be homogenous.

3.2 Low-pass frame selection

In the conventional MCTF based wavelet encoding scheme, the position of low-pass frame in a GOP is determined by the temporal filter type. As shown Fig. 1, the position of low-pass frame LLL_0 is F_0 in GOP. If only one temporal level is decoded in receiver, the decoded frames are at fixed positions. But in many situations, they are not the optimal representative frame of GOP. As far as our knowledge goes, there is not any paper has discussed the low-pass frame selection in MCTF encoding scheme. Key frame extraction is to find one or several representative frames from one video shot for video indexing or retrieval [24]-[25]. Hence, the method to extract key frame from a video shot can also be introduced to select a low-pass frame in GOP. In our scheme, method in reference [24] to extract key frame from frame cluster is introduced to determine low-pass frame position. In this method, the most representative frame is the one which maximizes inter-frame MI in the cluster:

$$F_{key} = \max_{j}\left(\frac{1}{N}\sum_{\substack{i=0\\ j\neq i}}^{N-1} MI_{j,i}\right) \tag{11}$$

Where N is number of frames in the cluster, and in the selection of low-pass frame, we can consider it as the GOP size.

$MI_{j,i}$ is the MI value of luminance component between F_j and F_i.

F_{key} is the selected representative frame from GOP.

In this approach, the selected low-pass frame can preserve the maximum correlation between frames within the GOP, which can reduce the prediction error in the GOP and improve the compression capability.

Adaptive Temporal decomposition process

After determining adaptively the GOP size, the frame number in the GOP may not satisfy the equation (3). A problem arises from the adaptive GOP size selection is how to determine the temporal decomposition process. At the same time, after the selection of a low-pass frame, the selected low-pass frame position may not be position F_0, so the temporal filtering scheme in Fig. 1 cannot be used directly.

In MCTF encoding scheme, temporal filtering is proceed in frame pair, and with the increase of temporal decomposition level, the distances in reserved frame pairs become longer. For example, just as shown in Fig. 1, the distances in 1st, 2nd, and 3rd temporal level are 1, 2, and 4 respectively. If the distance in frame pair is too long, it will lead to expensive prediction error and further great encoded bit cost. Hence, the frames at longest position away from low-pass frame should be filtered in low temporal level as far as possible to ensure that the distances in reserved frame pairs are short.

We propose a haar-like MCTF encoding scheme based on previous analysis, which can adaptively determine temporal decomposition process according to the selection of GOP size and low-pass frames position. In our proposal, the longest frames before and after low-pass frame are all filtered in the current temporal decomposition level, and the reserved frames which be filtered in the next level will maintain a short distance with low-pass frame.

The temporal filter type is haar in this paper, but the motion compensation direction is no longer in accord with conventional MCTF encoding scheme because of offset of low-pass frame position. In the proposed approach, the backward motion estimation is used in frames before low-pass frame and the forward motion estimation is used in frames after low-pass frames. This method tries its best to induce the distance between current frame and reference frame which can reduce the motion estimation *COST* as well, i.e. it can improve compression efficiency.

An example is used to illustrate this proposed approach. Let us consider GOP size is equal to 14, and the low-pass frame is F_8. The temporal decomposition process is as shown in Fig. 5.

When the GOP size satisfies the equation (3) and the position of low-pass frame is F_0, the process of proposed scheme is the same as the conventional haar filter.

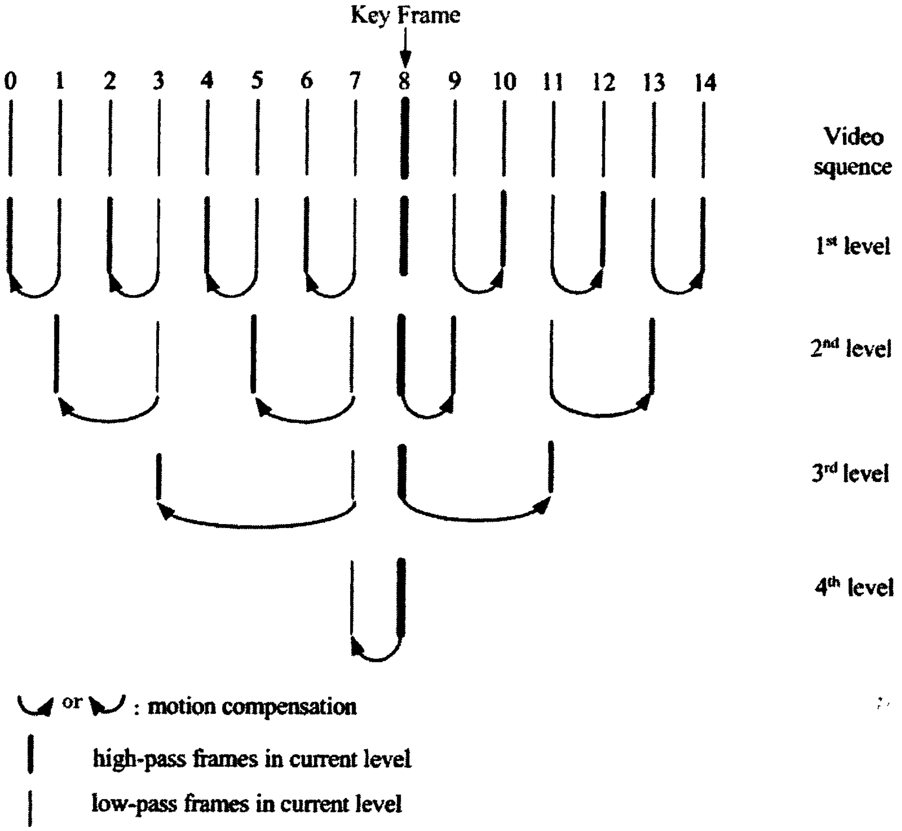


Fig. 5 Temporal decomposition process

4. Experimental Result and Discussion

Extensive experiments are performed to evaluate efficiency of the proposed encoding scheme. The experiments were run on Intel Pentium-IV 2.66GHz processor with 512M memory. In our experiments, video sequences with different motion characteristics are selected at 30fps (see Table 2). We use the Hierarchical Variable Size Block Matching (HVSBM) [3] in motion estimation and the block size selected varied from 64x64 to 4x4. The low-pass and high-pass frames obtained from MCTF are transformed and encoded by MC-EZBC [7] and [20]. Part of software module downloaded from [27] is utilized in our experiments.

The comparison with existing standard H.264 and AGS are also given in our experiments. The software to encode video sequence in H.264 format is JM12.1, and conditions in JM encoder is shown in Table 3.

Table 2 Video sequences and their motion characteristics

Video sequence	Frame size	Frames number	Motion characteristic
Mobile	CIF	300	Low
Foreman	CIF	300	Medium, shot cut

Stefan	CIF	300	High
Football	SIF	125	High
Tennis	SIF	112	Abrupt shot cut

Table 3 Conditions in JM encoder

Condistions	Values
Frame rate	30
ProfileIDC	Baseline profile
Motion estimation	Fast full search
Reference frame number	1
Rate control	Enable
Search range	16

The experimental results are shown in Table 5-Table 9. 'GOP8' and 'GOP16' represent the GOP sizes are 8 and 16 in the conventional MCTF encoding scheme respectively. 'ADGOP1' and 'ADGOP2' are the proposed adaptive GOP size selection method with different the parameters set are given in Table 4.

Table 4 Parameters set in adaptive GOP size selection

Name	low_MI	median_MI	high_MI	*var_T*
ADGOP1	1.6	2.1	3.2	0.16
ADGOP2	1.4	1.9	3.0	0.14

The symbol 'method+KF' means the method with adaptive low-pass frames selection.

According to experimental results, the proposed adaptive GOP size selection does not improve the compression performance of video sequences with the homogenous motion activities, e.g. mobile, stefan, football, and the low-pass frame selection is not suit to low motion activity sequence (mobile).

For a video sequence with the abrupt shot cuts (such as the tennis) or camera motion, the adaptive GOP selection can improve their compression performance. Especially for a shot being cut of abruptly, it can improve the compression efficiency greatly, about 0.8-1.0 db in the tennis video sequence. The low-pass frames selection scheme is suitable for the sequence with medium to high motion (foreman, stefan, football), and can improve the compression quality by about 0.3-0.5 db.

In the scene with low motion activity or abrupt shot cut, the overall compression performance of proposed method is very close to H.264, and the proposal still lags H.264 in other situations. Computational complexity of the proposal and H.264 is also compared in 0. According to the result, the computational complexity of the proposal is also too expensive and this is a future research topic.

In the comparison with AGS, about 0.2-0.3 db improvement can be achieved by the proposed GOP structure selection. Especially in the scene of abrupt shot cut, the proposal can improve about 0.9 db comparing to AGS. If the GOP size selection is considered only, there is about 0.1-0.2 db compression performance loss behind AGS except for abrupt shot cut scene.

The full GOP size in AGS is 16, and therefore the sub-GOP sizes are 16, 8, 4, and 2. The four pre-encoding should be carried out with full size GOP interval to selection final GOP size, and this is leading to expensive computational complexity just as shown in 0.

Table 5 Rate-distortion comparison of mobile

kbps	methods – PSNR(db)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
400	24.26	26.13	26.13	26.87	25.92	25.94	26.13	26.12
600	26.99	28.98	28.98	28.49	28.68	28.68	28.98	28.95
800	29.00	30.69	30.69	29.67	30.51	30.42	30.69	30.64
1200	31.67	32.99	32.99	31.62	32.81	32.75	32.99	32.92
1600	33.52	34.54	34.54	33.14	34.40	34.37	34.54	34.41
2000	34.95	35.93	35.93	34.38	35.79	35.69	35.93	35.78

Table 6 Rate-distortion comparison of foreman

kbps	methods – PSNR(db)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
400	32.93	33.29	33.61	35.64	33.52	34.01	33.50	34.00
600	34.94	35.18	35.51	37.21	35.35	35.72	35.36	35.80
800	36.29	36.41	36.78	38.36	36.48	37.13	36.59	37.03
1200	38.28	38.32	38.69	39.95	38.45	38.92	38.47	38.84
1600	39.68	39.67	40.05	41.13	39.79	40.31	39.83	40.17
2000	40.87	40.84	41.20	42.13	40.99	41.17	40.97	41.28

Table 7 Rate-distortion comparison of stefan

kbps	methods – PSNR(db)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
600	28.45	28.63	28.76	31.50	28.51	28.96	28.55	29.14
800	30.30	30.40	30.56	32.75	30.33	30.76	30.33	30.89
1200	32.82	32.74	33.01	34.69	32.79	33.20	32.83	33.29
1600	34.58	34.46	34.76	36.22	34.64	35.13	34.61	35.07
2000	36.03	35.86	36.20	37.50	36.15	36.42	36.08	36.47

Table 8 Rate-distortion comparison of football

kbps	methods – PSNR(db)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
600	25.12	24.66	25.34	27.39	25.18	25.62	25.10	25.58
800	26.43	25.99	26.58	28.77	26.42	26.92	26.38	26.86
1200	28.31	27.90	28.43	30.78	28.29	28.81	28.27	28.74
1600	29.93	29.56	30.00	32.34	29.89	30.37	29.88	30.31
2200	31.89	31.50	31.96	34.37	31.87	32.46	31.82	32.29
2600	33.10	32.73	33.18	35.47	32.98	33.61	33.02	33.50

3000	34.21	33.92	34.26	36.52	34.09	34.62	34.16	34.58

Table 9 Rate-distortion comparison of tennis

kbps	methods – PSNR(db)							
	GOP8	GOP16	AGS	H.264	ADGOP1	ADGOP1 +KF	ADGOP2	ADGOP2 +KF
400	29.84	30.38	30.54	31.52	31.31	31.39	31.31	31.40
600	31.61	32.10	32.29	33.18	33.18	33.25	33.11	33.15
800	32.96	33.51	33.67	34.51	34.51	34.61	34.48	34.57
1200	35.06	35.42	35.61	36.37	36.42	36.59	36.44	36.56
1600	36.71	37.05	37.23	37.74	37.92	38.10	37.98	38.01
2000	38.03	38.22	38.39	38.90	38.97	39.12	39.06	39.13

Table 10 Comparison of computational complexity between H.264 and proposed method

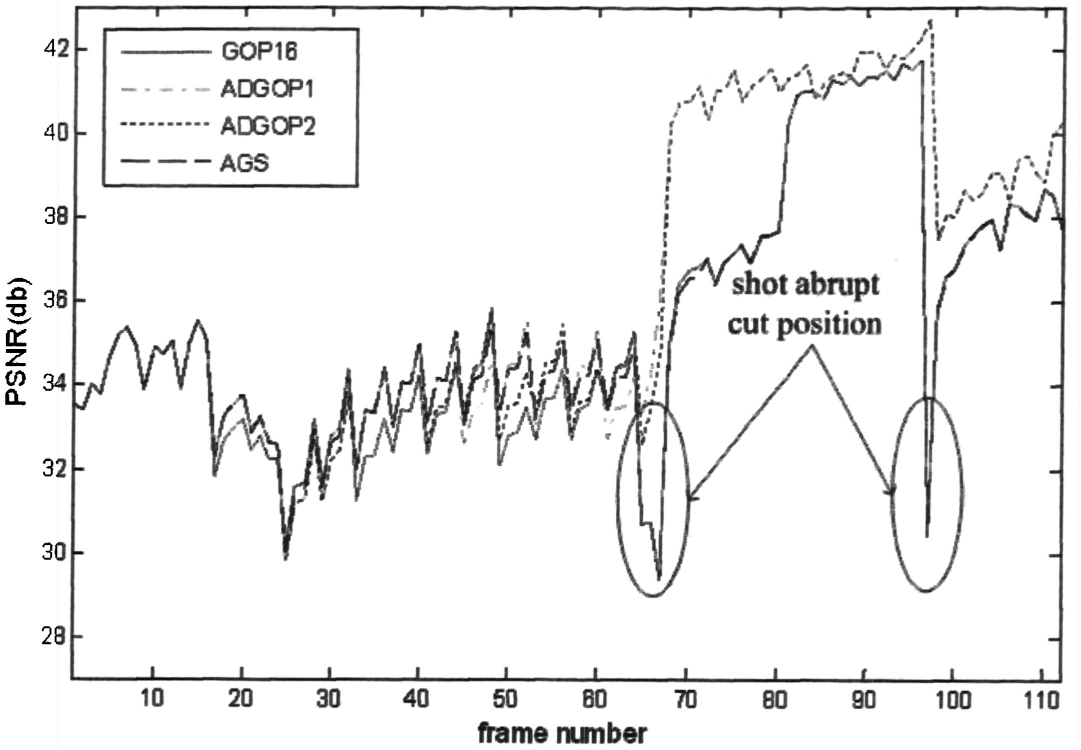
	Mobile	Foreman	Stefan	Football	Tennis
H.264 (sec)	335	315	328	125	101
ADGOP1 (sec)	6672	6486	7028	2873	2624

Table 11 GOP structure selection computational complexities in AGS and proposed method

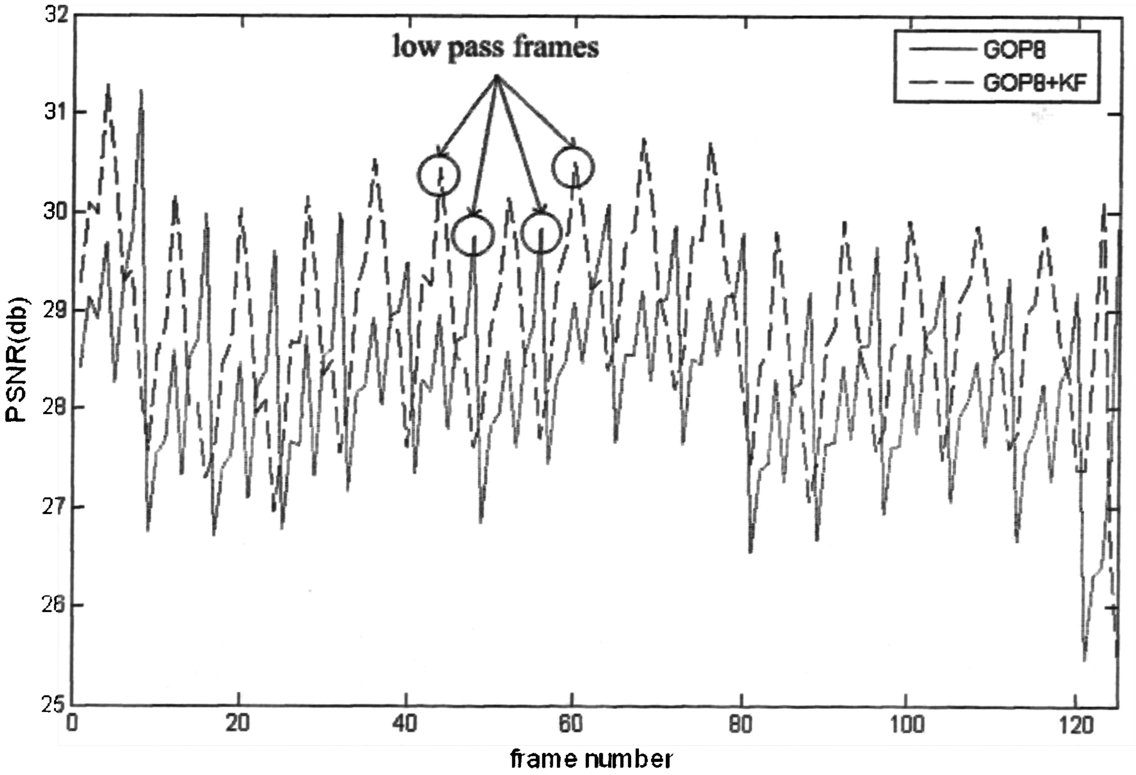
	Mobile	Foreman	Stefan	Football	Tennis
AGS (sec)	17610	17700	17570	6093	5259
ADGOP1 (sec)	77	72	65	17	15

In ‘tennis’, there are abrupt scene changes at frame 66 and frame 96, which lead to a great decrease in compression performance, as shown in Fig. 6-(a), and this situation is improved greatly in our adaptive GOP structure selection schemes. However, AGS can’t improve the compression performance greatly especially in abrupt shot cut position because of its weak scene change detection capability.

Positions of the low-pass frames have changed in the ‘GOP8+KF’ scheme as compared to ‘GOP8’ scheme, as shown in Fig. 6-(b), but the average values of PSNR of the ‘GOP8+KF’ is higher than that of the ‘GOP8’ scheme by about 0.5 db.



(a) Rate-distortion comparison in tennis kbps=1200



(b) Rate-distortion comparison in football kbps=1200

Fig. 6 Rate-distortion in video sequence

Temporal scalable decoding analysis

In our approach, the GOP structure selection is adaptively modified according to the motion activity among successive frames. When the motion activity is high, a short GOP size is used, while a long GOP size associates with the low motion segments. As discussed in Section II, the main part of information is included in frames with high motion activity. When a receiver decodes a bitstream in low temporal level, it can get more frames in high motion activity video segments and fewer frames in low motion activity segments. Hence, the decoded video sequence can descript the original video sequence more correctly than that of a fixed GOP size MCTF scheme.

For example, the GOP size selection of the video sequence 'foreman' is shown in Table 12, using the parameters set 'ADGOP1' given in Table 4.

Table 12 GOP selection in 'foreman'

Frame number	GOP size	*var_T*	average_MI
000-015	16	0.0419	2.9560
016-028	13	0.1481	2.9209
029-044	16	0.0445	2.9480
045-060	16	0.0894	2.9180
061-076	16	0.0151	2.9514
077-092	16	0.0509	2.6391
093-096	4	0.1220	2.7308
097-128	32	0.0457	3.4541
129-139	11	0.1498	3.0021
140-155	16	0.1419	2.5840
156-158	3	0.0342	2.3588
159-168	10	0.1390	3.0532
169-176	8	0.0145	1.9014
177-184	8	0.0044	1.5685
185-188	4	0.0008	1.4385
189-192	4	0.0018	1.3059
193-196	4	0.0008	1.2738
197-200	4	0.0000	1.3065
201-204	4	0.0068	1.4911
205-212	8	0.0005	1.5364
213-220	8	0.0013	1.5716
221-228	8	0.0008	1.6622
229-242	14	0.0645	1.9820
243-249	7	0.0714	2.0123
250-265	16	0.0705	2.8209
266-284	19	0.1418	3.2199
285-299	15	0.0821	2.8443

When the receiver only decodes one temporal level, the decoded low-pass frames of 'ADGOP1' and 'GOP16', in 'foreman', are shown in Fig. 7.

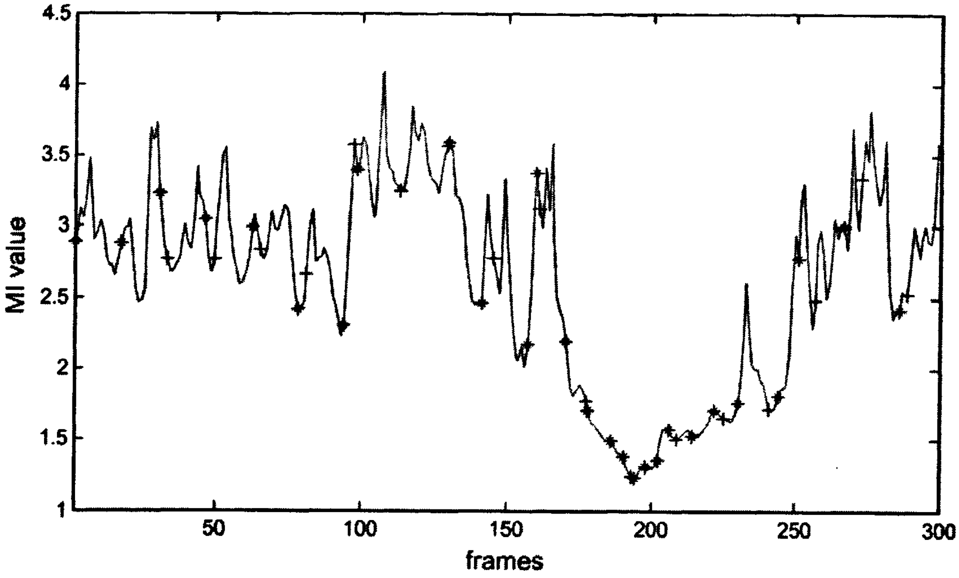


Fig. 7 Comparison of low-pass frames between fixed GOP size and adaptive GOP size

In Fig. 7, the curve represents the MI values among frames, '+' indicated low-pass frames decoded in the lowest temporal level in a fixed GOP size of 16 for encoding, and '*' indicated low-pass frames decoded from the adaptive GOP size selection using parameter set one 'ADGOP1'. Obviously, the number of low-pass frames decoded in 'ADGOP1' scheme is more than the fixed GOP size scheme when motion activity is high, around frame number 200.

5. Conclusions and future work

A haar-like MCTF encoding scheme is propose in the paper. In the proposed method, the GOP size can be selected adaptively according to MI values between successive frames. After determining the GOP, the low-pass frame in a GOP is also adaptively selected using the MI values. Finally, the temporal decomposition process in MCTF is determined according to the selected GOP structure. Experimental results show that when the proposed method is applied to videos including shot cut, or high motion activity, better compression performance can be obtained comparing to the conventional MCTF encoding scheme. In the scene with low motion activity or abrupt shot cut, the overall compression performance of proposed method is very close to H.264, and the proposal still lags H.264 in other situations. With the comparison with H.264, the computational complexity of the proposal is also too expensive and this is a future research topic. About 0.2-0.3 db improvement can be achieved by the proposed GOP structure selection comparing to AGS in standard SVC model.

The parameter sets of calculating the GOP size given by the proposed method are achieved by extensive experiments. The theoretic relationship between MI values and GOP size selection are possible extensions of the work. The method of low-pass frame selection using the MI values among frames in GOP is computationally expensive, so, fast algorithm or other methods to extract low-pass frames are fruitful directions for further research.

References

[1]. Eric Dubios, and Shaker Sabri, "Noise Reduction in Image Sequences Using Motion-Compensated Temporal Filtering," IEEE Trans. Communications, vol. COM-32, No. 7, pp. 826-831, Jul. 1984.

[2]. Jens-Rainer.Ohm, "Three-dimensional subband coding with motion compensation," IEEE Trans. Image Processing, vol. 3, pp. 559-571, Sept. 1994.

[3]. S.-J. Choi, and J. W. Woods, "Motion-compensated 3-D subband coding of video," IEEE Trans. Image Processing, vol. 8, pp. 155-167, Feb. 1999.

[4]. L. Luo, J. Li, S. Li, Z. Zhuang, and Y.-Q. Zhang, "Motion compensated lifting wavelet and its application in video coding," IEEE Int. Conf. Multimedia and Expo ICME, pp. 365-368, Aug. 2001.

[5]. B. Pesquet-Popescu, and V. Bottreau, "Three-dimensional lifting schemes for motion-compensated video compression," IEEE Int. Conf. Acoustics, Speech, and Signal Processing, pp. 1793-1796, May 2001.

[6]. A. Secker, and D. Taubman, "Motion-compensated highly-scalable video compression using an adaptive 3D wavelet transform based on lifting," IEEE Int. Conf. Image Processing, vol. 2, pp. 1029-1032, Oct. 2001.

[7]. Peisong Chen, and John W. Woods, "Bidirectional MC-EZBC With Lifting Implementation," IEEE Trans. Circuits and System for Video Technology, vol. 14, pp. 982-993, Oct. 2004.

[8]. Shih-Ta Hsiang, and John W. Woods, "Embedded video coding using invertible motion compensated 3-D subband/wavelet filter bank," Signal Processing: Image Communication, vol. 16, pp. 705-724, 2001.

[9]. Yonghui Wang, Suxia Cui, and James E. Fowler, "3-D Video coding with redundant-wavelet multihypothesis," IEEE Trans. Circuits and System for Video Technology, vol. 16, pp. 166-177, Feb. 2006.

[10]. Yiannis Andreopoulos, Adrian Munteanu, Joeri Barbarien, Mihaela Van der Schaar, Jan Cornelis, and Peter Schelkens, "In-band motion compensated temporal filtering," Signal Processing: Image Communication, vol. 19, pp. 653-673, Aug. 2004

[11]. Xin Li, "Scalable video compression via overcomplete motion compensated wavelet coding," Signal Processing: Image Communication, vol. 19, pp. 637-651, Aug. 2004.

[12]. Riccardo Leonardi, and Jens-Rainer Ohm, "Wavelet Video Coding - an Overview," MPEG Workgroup Video Subgroup, ISO/IEC JTC1/SC29/WG11 W7824, Bangkok, Thailand, Jan. 2006.

[13]. Ching-Yeh Chen, Chao-Tsung Huang, Yi-Hau Chen, Shoa-Yi Chien, and Liang-Gee Chen, "System analysis of VLSI architecture for 5/3 and 1/3 motion-compensated temporal filtering," IEEE Trans. Image Processing, vol. 54, pp. 4004-4014, Oct. 2006.

[14]. D.S. Turaga, M. van der Schaar, Y. Andreopoulos, A. Munteanu, and P. Schelkens, "Unconstrained motion compensated temporal filtering (UMCTF) for efficient and flexible

interframe wavelet video coding," Signal Processing: Image Communication, vol. 20, pp. 1-19, 2005.

[15]. Hwangjun Song, Jongwon Kim, and C.-C. Jay Kuo, "Real-time encoding frame rate control for H.263+ video over the internet," Signal Processing: Image Communication, vol. 15, pp. 127-148, 1999.

[16]. Jungwoo Lee, and Bradley W. Dickinson, "Temporally adaptive motion interpolation exploiting temporal masking in visual perception," IEEE Trans. Image Processing. vol. 3, pp. 513-526, Sept. 1994.

[17]. Lee J., Shin I, and Park H, "Adaptive intra-frame assignment and bit-rate estimation for variable GOP length in H.264," IEEE Trans. Circuits and Systems for Video Technology, vol. 16, pp. 1271-1279, Oct. 2006.

[18]. Yu-Lin Wang, Jing-Xin Wang, yen-Wen Lai, and Alvin W. Y Su, "Dynamic GOP structure determination for real-time MEPG-4 advanced simple profile video encoder," IEEE Int. Conf. Multimedia and Expo., pp. 293-296, Jul. 2005.

[19]. L. Wang, "Rate control for MPEG video coding," Signal processing: Image communication, vol. 15, pp. 493-511, 2000.

[20]. Peisong Chen, "Fully scalable subband/wavelet coding," Doctoral Thesis, Rensselaer Polytechnic Institute Troy, New York, May, 2003.

[21]. G.H. Park, M.W. Park, S Jeong, J. Cha, K. Kim, and J. Hong, "Adaptive GOP structure for SVC," ISO/IEC/JTC1/SC29/WG11/MPEG/ M11563, Hong Kong, Jan. 2005.

[22]. G.H. Park, M.W. Park, S. Jeong, K. Kim, and J. Hong, "Improve SVC coding efficiency by adaptive GOP structure (SVC CE2)," Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) JVT-O018, Korea, Apr. 2005.

[23]. M.W. Park, G.H. Park, S. Jeong, Doug-Young Suh, and K. Kim, "Adaptive GOP Structure for Joint Scalable Video Coding," IEICE Trans. Communication, vol. E90–B, No.2, Feb. 2007.

[24]. Zuzana Cerneková, Ioannis Pitas, and Christophoros Nikou, "Information theory-based shot cut/fade detection and video summarization," IEEE Trans. Circuits and System for Video Technology, vol. 16, pp. 82-91, Jan. 2006.

[25]. Wengang Cheng, Yaniing Liu, and De Xu, "Shot boundary detection based on the knowledge of information theory," IEEE Int. Conf. Neural Networks and Signal Processing, vol. 2, pp. 1237-1241, Dec. 2003.

[26]. T Butz, and JP Thiran, "Shot boundary detection with mutual information," IEEE Int. Conf. Image Processing, vol. 3, pp.421-424, Oct. 2001.

[27]. Peisong Chen, Software package of MC-EZBC wavelet coder is publicly available at ftp://ftp.cipr.rpi.edu/personal/chen.

[28]. Christophe Tillier, Béatrice Pesquet-Popescu, and Mihaela van der Schaar, "3-Band motion-compensated temporal structures for scalable video coding," IEEE Trans. Image Processing, vol. 15, pp. 2545-2557, Sep. 2006.

[29]. Deepak S. Turaga, Mihaela van der Schaar, and Beatrice Pesquet-Popescu, "Complexity scalable motion compensated wavelet video encoding," IEEE Trans. Circuits and System for

Video Technology, vol. 15, pp. 982-993, Aug. 2005.

[30]. Jens-Rainer Ohm, "Advances in Scalable Video Coding," Proceedings of the IEEE, vol. 93, Issue 1, pp. 42-56, Jan. 2005.

[31]. Hendrik Eeckhaut, Harald Devos, Benjamin Schrauwen, Mark Christiaens, and Dirk Stroobandt, "A hardware-friendly wavelet entropy codec for scalable video," IEEE Design, Automation and Test in Europe Conference and Exhibition (DATE'05), vol. 3, pp. 14-19, 2005.