

## 摘要

随着网络技术、通信技术及多媒体技术的发展,网络多媒体应用,尤其是网络视频应用变得越来越广泛。基于 H.323 或 H.324 的网络视频电话[1][2]、视频会议以及基于无线网的视频通信得到了迅猛的发展。但是由于存在大量易损信道(Lossy Channel),使当前网络状况并不能很好地满足视频传输的需要。H.26L 作为一种新的视频编码标准,它的高压缩率、网络适应性以及对差错鲁棒性使它很适合于作为易损信道中的视频传输压缩标准。但是它在差错控制方面还存在一些不足需要改进。

本文在研究 H.26L 视频编码标准的基础上重点研究了 H.26L 的差错控制技术,包括差错恢复和差错隐藏技术。H.26L 的编码端采用了重同步、数据分区、帧内宏块刷新等差错恢复技术,限制了差错范围,减小了误码扩散的影响。解码端采用了差错隐藏技术,主要包括帧内编码中的像素距离加权平均算法和帧间编码中的根据预测运动向量进行运动补偿。本文根据仿真实验的结果,发现了 H.26L 中差错隐藏算法的不足,并提出一种新的改进方案,通过仿真实验,验证了改进后的差错隐藏方案具有更好的性能表现。

针对 H.26L 在差错控制方面的不足,本文研究了一种实用的信道编码——Reed-Solomon (RS) 码。通过编程实现了 RS 码的编译码,然后应用于视频传输系统的信道差错控制,对于纠正视频在易损信道中传输出现的随机错和突发错起到了重要作用,并编程实现了一种高效率的查表算法,为实际应用提供了保证。

本文借鉴 H.263 的视频传输模型,研究了基于 H.26L 的视频传输系统中帧内刷新率和信道编码效率的参数最优化问题以获得最佳视频质量。通过仿真实验验证了基于 H.26L 的模型在较低误码率情况下的准确性,并根据模型寻找了特定网络环境下的这两个参数的最优值。

**关键字:** H.26L, 差错隐藏, Reed-Solomon 码, 视频传输模型

## Abstract

With the development of network, communication and multimedia technology, the network multimedia, especially the network video applications, such as network videophone based on H.323/H.324, videoconference and video communication based on wireless network, have developed rapidly. But present network environment cannot meet the need of video transmission for there still exist some lossy channels. As a new video coding standard, H.26L becomes a suitable choice for video transmission on lossy channel because it has the advantages of high compression ratio, network adaptability and error robustness. But it still has some disadvantages in error control.

This paper mainly studies on the error control techniques in H.26L video coding standard, which contains error resilience and error concealment. The error resilience techniques include resynchronization, data partition and INTRA macroblock update, which can restrain error scope and reduce the influence of error propagation. And the error concealment techniques include weighted pixel averaging algorithm based on INTRA frame coding and motion compensation using predicted motion vectors based on INTER frame coding. The defects of error concealment algorithms in H.26L are exposed by the results of simulation. Then a new improved scheme is proposed, and it is verified by the simulation results.

To overcome the disadvantages of error control in H.26L, this paper discusses a useful channel code, Reed-Solomon (RS) Code. The RS Codec is implemented by programming and is used for channel error control in the video transmission system. It provides quality insurances for the video transmission over lossy channels because it plays an important part in the random and burst error correction. This paper also implements a table lookup algorithm to improve the codec efficiency.

Finally, referring to the H.263 video transmission system, this paper discusses the optimization of the INTRA rate and the channel code rate in the video transmission system based on H.26L to achieve the best overall performance of decoded video. The model proves out to be feasible at low error rate by the simulation experiments. At last, the optimized model parameters are found for given network environment.

**Keywords:** H.26L, Error Concealment, Reed-Solomon Code, Video Transmission System Model

# 独 创 性 声 明

本人声明所呈交的学位论文是我本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表和撰写过的研究成果，也不包含为获得国防科学技术大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

学位论文题目： 易损信道中 H. 26L 视频传输差错控制模型及技术研究

学位论文作者签名： 朱 为 日期： 2003 年 11 月 25 日

## 学位论文版权使用授权书

本人完全了解国防科学技术大学有关保留、使用学位论文的规定。本人授权国防科学技术大学可以保留并向国家有关部门或机构送交论文的复印件和电子文档，允许论文被查阅和借阅；可以将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、缩印或扫描等复制手段保存、汇编学位论文。

（保密学位论文在解密后适用本授权书。）

学位论文题目： 易损信道中 H. 26L 视频传输差错控制模型及技术研究

学位论文作者签名： 朱 为 日期： 2003 年 11 月 25 日

作者指导教师签名： 王 昭 日期： 2003 年 12 月 4 日

## 图目录

图 1-1 误码扩散示意图.....	1
图 1-2 出错导致同步丢失.....	1
图 1-3 MPEG-4 中对运动数据与 DCT 数据分区 .....	3
图 1-4 RVLC 双向解码 .....	3
图 1-5 空域差错隐藏.....	4
图 1-6 时域差错隐藏.....	4
图 2-1 ITU-T 和 MPEG 标准发展过程.....	7
图 2-2 H.26L 系统层结构 .....	8
图 2-3 H.26L 编码流程图 .....	8
图 2-4 H.26L 解码流程图 .....	9
图 2-5 4×4 亮度块帧内预测模式 .....	9
图 2-6 16×16 亮度块帧内预测模式 .....	10
图 2-7 树形运动补偿块分割模式.....	11
图 2-8 实际图像运动补偿块分割示意图.....	11
图 2-9 整数像素和 1/4 像素运动向量预测 .....	11
图 2-10 多参考帧示意图.....	12
图 2-11 4×4 整数变换公式 .....	12
图 2-12 变换系数扫描方式.....	13
图 2-13 差错控制流程图.....	14
图 2-14 差错状态图.....	16
图 2-15 像素距离加权平均算法示例.....	16
图 2-16 运动向量预测及边界差值计算.....	17
图 2-17 仿真丢包文件的使用.....	18
图 2-18 Foreman 序列差错隐藏主观图像恢复效果比较, 丢包率 5% .....	19
图 2-19 Coastguard 序列差错隐藏主观图像恢复效果比较, 丢包率 5% .....	19
图 2-20 Foreman 序列中 I 帧丢失 Slice 时差错隐藏效果 .....	20
图 2-21 Coastguard 序列中 I 帧丢失 Slice 时差错隐藏效果 .....	20
图 2-22 各种数据率下差错隐藏恢复图像平均 PSNR 比较.....	21
图 3-1 信道编码在数字通信系统中的位置.....	22
图 3-2 RS 编码流程工作图.....	26
图 4-1 视频传输系统构成图.....	28
图 4-2 不同 $\beta$ 值的 RD 曲线.....	32
图 4-3 $D_v$ 实验数据和模型拟合曲线图 .....	34
图 4-4 差错控制信道性能.....	35
图 4-5 $PSNR_d$ 随 $\beta$ 变化曲线.....	36
图 4-6 $PSNR_d$ 随 $r$ 变化曲线 .....	37
图 4-7 $\beta$ 和 $r$ 联合优化 .....	38
图 4-8 不同差错率下 $\beta$ 和 $r$ 的最优组合 .....	38

## 表目录

表 2-1 H.26L 和 MPEG-4 对比.....	14
表 2-2 差错隐藏仿真结果 PSNR 平均值对比, 帧率为 30 帧 / 秒.....	19
表 2-3 改进差错隐藏性能对比.....	21
表 3-1 查表法性能测试对比.....	27
表 4-1 视频传输系统参数列表.....	29

## 第一章 绪论

### § 1.1 背景及问题的提出

近年来, 由于网络技术、通信技术及多媒体技术的长足发展, 网络视频应用变得越来越广泛, 通过网络进行传输的视频电话、视频会议系统、可视化远程教育和远程医疗系统具有广泛的市场应用前景。而且无线网络的视频通信也随着无线通信技术的提高而逐渐实用化。但是这些网络视频应用要完全满足用户的使用要求还有很多困难需要克服:

#### 1. 网络的带宽限制

尽管网络传输速度不断地高速向前发展, 光纤链路也可达到很高的传输速率, 但人们仍然面临着大量的低速网络, 如当前广泛使用的因特网和移动网正是这样的网络。为了在带宽受限的网络上传输视频, 必须使用高效的视频压缩编码。

#### 2. 易损信道影响

由于信道的物理原因和外界环境的影响, 很多低速不可靠网络很容易出现差错, 包括各种随机错、突发错以及网络丢包。这种易出现差错的信道称之为易损信道, 例如目前的 CDMA、GPRS、无线局域网等都属于易损信道。目前一般的视频编码标准都是基于运动补偿预测和熵编码的, 由于运动预测补偿, 出错会导致一帧图像中的差错影响后续图像的质量 (如图 1-1 所示), 这称为误码扩散 (Error Propagation); 而且由于压缩都需要熵编码, 当出现差错时会导致同步丢失, 出错位置至下一个同步标志之间的数据都会被丢弃 (如图 1-2 所示), 使视频编码对差错的鲁棒性较差。



图 1-1 误码扩散示意图

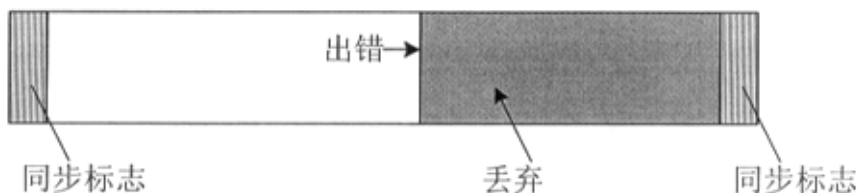


图 1-2 出错导致同步丢失

针对上述困难, 为了满足用户对网络视频应用的需求, 除了研究高效的视频编码, 对于易损信道上视频传输的差错控制技术研究也显得十分重要。差错控制技术包含的内容比较多, 应用范围涵盖了信源的差错恢复技术、信道的信道编码和信宿的差错隐藏。H.26L 作为一种新的视频编码标准, 它的高压缩率、网络适应性以及对差错鲁棒性使它很适合于

作为易损信道中的视频传输压缩标准。因此,本文对基于 H.26L 的端对端视频传输系统做了整体的研究,通过视频传输系统模型全面考虑信源、信道和信宿各个部分的差错控制技术,使最终解码后的视频质量达到最优。

## § 1.2 国内外研究现状

### 1.2.1 差错控制技术概述

对付视频传输差错的差错控制技术主要有以下两种:一种是传统的前向纠错 FEC (Forward Error Correction),自动反馈重传 ARQ (Automatic Repeat Request) 等差错控制和恢复技术,以保证通信时尽可能做到无损为目的,主要应用于通信系统的底层;另一种则是系统层的差错恢复 (Error Resilience) 和差错隐藏 (Error Concealment) 技术。差错恢复技术使差错可以被检测出并限制在某一范围内,为差错隐藏奠定基础,差错隐藏则根据通过差错恢复得到的数据用填补掩盖等办法恢复不能正常解码的数据,使图像的视觉效果接近于原来的效果,两部分结合才能更好地实现差错控制,保证视频传输的质量。

前向纠错 FEC 是发端将数据按一定的规律附加多余的码元,组成具有纠错能力的码。接收端收到码元后按预定的规则译码,以确定接收码组中是否有错误,若有错误则确定其位置并进行纠正。它不需要反馈信道,能用于单向通信,译码延迟固定,较适于实时传输系统,但是 FEC 要纠正较多的错误就要附加更多的码元。因此,前向纠错 FEC 适合于随机错和突发错的纠错,对于信道误码率较高和大量丢失数据包的情况并不是很适用。

自动反馈重发 ARQ 是在接收端收到码组后,按一定规则对发送信号进行有无错误的判别并把判别结果通过反馈信道送回到发端,发端根据应答信号把接收端认为有错的那组数据再次传输,直到接收端认可为止。因此,该方式需要反向信道,同时也存在传输时延,对于某些实时应用(如视频会议)这是无法容忍的。另外,为了有效地利用网络的带宽,现阶段在多点通信的应用中广泛采用了广播(broadcast)和多播(multicast)技术,这种情况下也无法通过重传来获得可靠性。但是,对于某些低延迟的网络环境,适当应用缓冲的情况下 ARQ 还是具有很好的差错控制效果的。

差错恢复技术的种类比较多,其中广泛应用于 H.263、MPEG-4 中的主要有以下几种[3][4]:

1. 重同步 (Resynchronization): 它是一种简单而实用的差错恢复技术。在传统编码方法中,由于压缩编码的固有性质,当解码器遇到比特错误时,后续数据不能继续解码,也就是失去与编码器的同步,造成视频质量大幅度下降。重同步方法是在编码之后的比特流的不同位置插入重同步标识,当解码器检测到错误时,它将查找这些标识,从而重新获得同步开始继续解码。虽然使用重同步会增加一些重同步标识比特,但是该技术可以限制差错范围,提高了压缩数据的使用效率,其增强视频码流差错鲁棒性的作用还是很显著的。

2. 数据分区 (Data Partition): 在 MPEG-4 中通过在运动矢量等运动数据部分和 DCT 数据部分之间加一个 MBM 标识将两部分分开,如图 1-3 所示。此图也表示了采用数据分组的视频包的比特流与组织形式,为了减少与传统方法的差异,在数据部分将保持同样的元素以及结构和识别方法,所有与运动相关的元素信息放入运动数据部分(即运动矢量等数据),与 DCT 数据相关放入 DCT 数据部分。这样当视频包中的数据出现错误时解码器就可以分辨出是运动数据错误还是 DCT 数据有误,从而可以分别进行处理。



作为易损信道中的视频传输压缩标准。因此,本文对基于 H.26L 的端对端视频传输系统做了整体的研究,通过视频传输系统模型全面考虑信源、信道和信宿各个部分的差错控制技术,使最终解码后的视频质量达到最优。

## § 1.2 国内外研究现状

### 1.2.1 差错控制技术概述

对付视频传输差错的差错控制技术主要有以下两种:一种是传统的前向纠错 FEC (Forward Error Correction),自动反馈重传 ARQ (Automatic Repeat Request) 等差错控制和恢复技术,以保证通信时尽可能做到无损为目的,主要应用于通信系统的底层;另一种则是系统层的差错恢复 (Error Resilience) 和差错隐藏 (Error Concealment) 技术。差错恢复技术使差错可以被检测出并限制在某一范围内,为差错隐藏奠定基础,差错隐藏则根据通过差错恢复得到的数据用填补掩盖等办法恢复不能正常解码的数据,使图像的视觉效果接近于原来的效果,两部分结合才能更好地实现差错控制,保证视频传输的质量。

前向纠错 FEC 是发端将数据按一定的规律附加多余的码元,组成具有纠错能力的码。接收端收到码元后按预定的规则译码,以确定接收码组中是否有错误,若有错误则确定其位置并进行纠正。它不需要反馈信道,能用于单向通信,译码延迟固定,较适于实时传输系统,但是 FEC 要纠正较多的错误就要附加更多的码元。因此,前向纠错 FEC 适合于随机错和突发错的纠错,对于信道误码率较高和大量丢失数据包的情况并不是很适用。

自动反馈重发 ARQ 是在接收端收到码组后,按一定规则对发送信号进行有无错误的判别并把判别结果通过反馈信道送回到发端,发端根据应答信号把接收端认为有错的那组数据再次传输,直到接收端认可为止。因此,该方式需要反向信道,同时也存在传输时延,对于某些实时应用(如视频会议)这是无法容忍的。另外,为了有效地利用网络的带宽,现阶段在多点通信的应用中广泛采用了广播(broadcast)和多播(multicast)技术,这种情况下也无法通过重传来获得可靠性。但是,对于某些低延迟的网络环境,适当应用缓冲的情况下 ARQ 还是具有很好的差错控制效果的。

差错恢复技术的种类比较多,其中广泛应用于 H.263、MPEG-4 中的主要有以下几种[3][4]:

1. 重同步 (Resynchronization): 它是一种简单而实用的差错恢复技术。在传统编码方法中,由于压缩编码的固有性质,当解码器遇到比特错误时,后续数据不能继续解码,也就是失去与编码器的同步,造成视频质量大幅度下降。重同步方法是在编码之后的比特流的不同位置插入重同步标识,当解码器检测到错误时,它将查找这些标识,从而重新获得同步开始继续解码。虽然使用重同步会增加一些重同步标识比特,但是该技术可以限制差错范围,提高了压缩数据的使用效率,其增强视频码流差错鲁棒性的作用还是很显著的。

2. 数据分区 (Data Partition): 在 MPEG-4 中通过在运动矢量等运动数据部分和 DCT 数据部分之间加一个 MBM 标识将两部分分开,如图 1-3 所示。此图也表示了采用数据分组的视频包的比特流与组织形式,为了减少与传统方法的差异,在数据部分将保持同样的元素以及结构和识别方法,所有与运动相关的元素信息放入运动数据部分(即运动矢量等数据),与 DCT 数据相关放入 DCT 数据部分。这样当视频包中的数据出现错误时解码器就可以分辨出是运动数据错误还是 DCT 数据有误,从而可以分别进行处理。



作为易损信道中的视频传输压缩标准。因此,本文对基于 H.26L 的端对端视频传输系统做了整体的研究,通过视频传输系统模型全面考虑信源、信道和信宿各个部分的差错控制技术,使最终解码后的视频质量达到最优。

## § 1.2 国内外研究现状

### 1.2.1 差错控制技术概述

对付视频传输差错的差错控制技术主要有以下两种:一种是传统的前向纠错 FEC (Forward Error Correction),自动反馈重传 ARQ (Automatic Repeat Request) 等差错控制和恢复技术,以保证通信时尽可能做到无损为目的,主要应用于通信系统的底层;另一种则是系统层的差错恢复 (Error Resilience) 和差错隐藏 (Error Concealment) 技术。差错恢复技术使差错可以被检测出并限制在某一范围内,为差错隐藏奠定基础,差错隐藏则根据通过差错恢复得到的数据用填补掩盖等办法恢复不能正常解码的数据,使图像的视觉效果接近于原来的效果,两部分结合才能更好地实现差错控制,保证视频传输的质量。

前向纠错 FEC 是发端将数据按一定的规律附加多余的码元,组成具有纠错能力的码。接收端收到码元后按预定的规则译码,以确定接收码组中是否有错误,若有错误则确定其位置并进行纠正。它不需要反馈信道,能用于单向通信,译码延迟固定,较适于实时传输系统,但是 FEC 要纠正较多的错误就要附加更多的码元。因此,前向纠错 FEC 适合于随机错和突发错的纠错,对于信道误码率较高和大量丢失数据包的情况并不是很适用。

自动反馈重发 ARQ 是在接收端收到码组后,按一定规则对发送信号进行有无错误的判别并把判别结果通过反馈信道送回到发端,发端根据应答信号把接收端认为有错的那组数据再次传输,直到接收端认可为止。因此,该方式需要反向信道,同时也存在传输时延,对于某些实时应用(如视频会议)这是无法容忍的。另外,为了有效地利用网络的带宽,现阶段在多点通信的应用中广泛采用了广播(broadcast)和多播(multicast)技术,这种情况下也无法通过重传来获得可靠性。但是,对于某些低延迟的网络环境,适当应用缓冲的情况下 ARQ 还是具有很好的差错控制效果的。

差错恢复技术的种类比较多,其中广泛应用于 H.263、MPEG-4 中的主要有以下几种[3][4]:

1. 重同步 (Resynchronization): 它是一种简单而实用的差错恢复技术。在传统编码方法中,由于压缩编码的固有性质,当解码器遇到比特错误时,后续数据不能继续解码,也就是失去与编码器的同步,造成视频质量大幅度下降。重同步方法是在编码之后的比特流的不同位置插入重同步标识,当解码器检测到错误时,它将查找这些标识,从而重新获得同步开始继续解码。虽然使用重同步会增加一些重同步标识比特,但是该技术可以限制差错范围,提高了压缩数据的使用效率,其增强视频码流差错鲁棒性的作用还是很显著的。

2. 数据分区 (Data Partition): 在 MPEG-4 中通过在运动矢量等运动数据部分和 DCT 数据部分之间加一个 MBM 标识将两部分分开,如图 1-3 所示。此图也表示了采用数据分组的视频包的比特流与组织形式,为了减少与传统方法的差异,在数据部分将保持同样的元素以及结构和识别方法,所有与运动相关的元素信息放入运动数据部分(即运动矢量等数据),与 DCT 数据相关放入 DCT 数据部分。这样当视频包中的数据出现错误时解码器就可以分辨出是运动数据错误还是 DCT 数据有误,从而可以分别进行处理。

重同步标识	MB.no	QP	运动矢量数据	MBM	DCT数据
-------	-------	----	--------	-----	-------

图 1-3 MPEG-4 中对运动数据与 DCT 数据分区

3. 帧内宏块刷新 (INTRA macroblock update): 帧内宏块刷新是指在编码过程中, 对于预测编码帧 (P 帧和 B 帧) 按照一定的规律或是在一定范围内随机采用帧内编码方式编码某一个宏块, 使 P 帧和 B 帧中每过一段时间也会出现一个帧内编码的宏块。这样在出现误码扩散时, 帧内编码宏块可以在一定程度上减少误码扩散, 降低误码扩散带来的视频质量下降。

4. 可逆变长编码 (RVLC): 在易损信道中传输压缩视频通常是变字长编码 (VLC) 或算术编码。在解码端, 如果解码器检测到编码数据中的误码, 它将失去同步信号, 因此, 不得不将直到下一个重同步点的所有数据全部舍弃, 而 RVLC 避免了这个问题, 使得解码器通过在误码处的数据反转, 以更好地确定误码位置, 如图 1-4 所示。RVLC 是有前缀特性的特别的 VLC, 可以从前向或反向进行解码。这种码字的优点在于, 当解码器在前向解码时遇到误码, 它可以跳到下一个重同步点进行反向解码, 直到遇到误码, 基于两个误码的位置解码器可以比普通重同步方法多恢复一些数据, 提高了数据使用率。

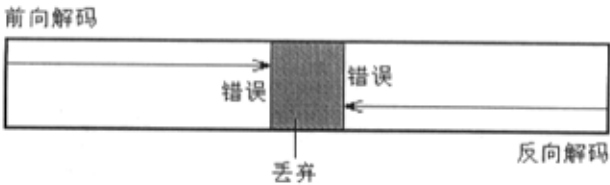


图 1-4 RVLC 双向解码

5. 差错跟踪: 差错跟踪技术是在不改变标准的情况下, 编码器借助解码器通过反馈信道传来的图像受损信息, 根据事先保存下来的运动向量, 依照差错扩散规律, 在编码器方评估解码器受损情况, 并跟踪到当前待编码帧, 确定出哪些宏块需要采用帧内编码模式以此来消除错误。该方法不需要重传损失的数据, 因而只要求较少信道带宽。在允许反馈信道的情况下, 该方法是一种效率比较高、效果比较好的差错恢复技术。

差错隐藏技术的目的是在解码端降低错误的影响, 由于图像序列本身就有很强的空间和时间上的相关性, 这为用正确接收到的视频数据去预测丢失的数据提供了可能。而且由于该过程是在解码器端完成的, 故不会引起网络的拥塞和延迟。差错隐藏技术适合于实时应用中大量数据丢失的情况。差错隐藏方法一般分为两类, 即空域的差错隐藏 (如图 1-5 所示) 和时域的差错隐藏 (如图 1-6 所示)。

1. 空域差错隐藏利用当前帧上正确接收的信息, 用丢失宏块的相邻宏块的数据恢复遭到破坏的数据。这些方法包括最大平滑恢复和双向内插。这些方法假设相邻的图像块中存在强相关性, 因此, 如果受损块和相邻块处于亮度均一的区域, 则这些方法会产生较好的掩盖效果, 但如果它们处于图像边缘部分或复杂的细节部分, 则会导致图像质量的钝化。

2. 时域差错隐藏算法则利用前一帧的图像数据来估计错误数据, 即用邻域宏块的运动

矢量来估计受损块的运动矢量，然后用其运动补偿块来代替受损宏块。一种最简单的方法是将受损宏块的运动矢量变为 0，即用上一帧中相应位置的宏块来替换当前宏块的数据。这种方法对于图像的静止或运动较小的部分会收到理想的图像质量（如图 1-6 上面两幅图所示），但是对于快速运动区域，以及当情景切换时，该方法则会导致明显的方块效应，造成图像的不连续（如图 1-6 下面两幅图所示）。重建运动矢量的其它方法包括运动矢量的线性内插和中值。



图 1-5 空域差错隐藏



图 1-6 时域差错隐藏

### 1.2.2 H.26L 的研究现状

H.26L 标准到 2002 年才正式定案，目前基于该标准的研究主要是关于单独的编码器或解码器性能的研究。文献[5]中提出了一种基于 H.26L 的精细的可伸缩视频编码技术，意在使之更适合于不同码率的网络接入状况。文献[6]中研究了在丢包网络环境中 H.26L 编码器的率失真优化。文献[7]提出了一种基于图像内容的差错隐藏技术。

由于 H.26L 编码的特点，它还具有一些不足需要在研究加以改进。H.26L 和以往的视频编码标准类似都使用了运动预测补偿，这使误码扩散问题也存在于 H.26L 编码标准中。采用帧内宏块刷新（INTRA Macroblock Update）是一个比较有效的阻止误码扩散的方法，帧内刷新率（INTRA Rate）越高误码扩散的影响越小。但是随着帧内刷新率的提高，编码数据码率就会相应提高，降低了编码效率。因此帧内刷新率的选择是一个比较重要的问题，

文献[8]中提出一种优化帧内刷新率的选择方法。

H.26L 编码标准中熵编码部分采用 CABAC 或 UVLC, 即使编码数据只有一位错误, 都可能导致该位错误所在像条 (Slice) 或帧数据块的后续数据全部不能解码, 只有丢弃, 因此不能兼顾编码数据的高效性和鲁棒性。对于信道中出现的一位或多位的数据错误, 在保持视频数据实时性的前提下, 一个较好的办法就是使用信道编码技术, 让编码后的视频数据具有自我纠错的能力, 使出错数据在接收端可以恢复成正确的数据。然而使用信道编码实际上是带来了编码数据的冗余, 在增强了编码数据的鲁棒性的同时, 增加了数据率, 降低了编码效率, 在网络带宽受限的情况下更增加了带宽的负担。一方面增加校验位长度能纠正更多错误, 但同时占用更多带宽, 在带宽紧张的情况下可能会影响数据传输; 另一方面校验位太少只能纠正较少错误, 丢弃数据太多, 解码图像质量下降很大。为了解决这个问题, 可以有选择地对视频数据进行信道编码保护[9][10], 例如只保护运动预测得到的运动向量等预测信息, 但是这样毕竟还是丢失了残差信息, 解码图像质量还不是很好。

根据以上情况, 本文借鉴 H.263 的视频传输模型, 应用于基于 H.26L 的端对端视频传输系统, 分析寻找帧内刷新率和信道编码监督段长度的最优值, 在固定带宽的情况下在接收端获得最佳的视频质量。

### § 1.3 本篇论文解决的问题与贡献

本篇论文主要做了以下工作:

1. 研究了 H.26L 视频编码标准及其采用的差错控制技术。H.26L 的差错控制技术分为差错恢复和差错隐藏技术。编码端采用了一些差错恢复技术, 如重同步、数据分区、帧内宏块刷新, 限制了差错范围, 减小了误码扩散的影响。解码端采用了差错隐藏技术, 利用人眼的容错性, 对丢失的数据区域进行填补掩盖, 尽最大可能保证解码的视频质量。本文着重研究了差错隐藏技术, 并且针对其不足提出了一种改进方案, 通过仿真实验验证了该方案的有效性。

2. 本文研究了一种实用的信道编码技术——RS 码。通过编程实现了 RS 码的编译码, 然后应用于视频传输系统的信道差错控制, 对于纠正部分随机错和突发错起到了重要作用, 为视频传输提供了一定的质量保证。而且在编程过程中实现了一种查表算法, 使以往困扰软件信道编码实现的效率问题得到了解决, 为实际应用提供了保证。

3. 本文借鉴 H.263 的视频传输模型研究了基于 H.26L 的端对端视频传输系统的差错控制特性。通过模型分析了帧内刷新率  $\beta$  和信道编码效率  $r$  的最优化问题, 这两个参数的变化对视频解码质量都有直接的影响, 是影响模型性能的最主要参数。通过仿真实验验证了基于 H.26L 的模型在较低误码率情况下的准确性, 并根据模型寻找了特定网络环境下的这两个参数的最优值, 从而使解码端视频质量在固定码率的情况下达到最好。该模型的研究为易损信道上视频传输的差错控制提供了分析平台, 为今后在实用系统中的差错控制研究打下了基础。

### § 1.4 本文结构

本文的内容是这样安排的:

第一章是绪论, 介绍了网络视频应用的研究背景和面临的一些困难, 提出了本文研究

的目标。

第二章将简要介绍 H.26L 标准及其差错控制技术, 重点介绍差错隐藏技术, 并且针对 H.26L 差错隐藏技术的不足提出了一种改进方案, 通过仿真实验验证差错隐藏的效果和改进后的性能提升。

第三章将介绍信道编码理论及一种实用的信道编码——RS 码, 以及它的实现和算法改进。

第四章介绍基于 H.26L 的端对端视频传输系统差错控制模型, 并通过仿真实验验证差错控制模型的有效性, 找到了模型在特定环境下的参数最优值。

第五章是对本文的总结。

## 第二章 H.26L 视频编码标准及差错控制技术研究

### § 2.1 H.26L 视频编码标准研究

#### 2.1.1 H.26L 编码标准概述

目前制定视频编码标准的国际组织主要有两个：ITU-T 和 ISO/IEC。ITU-T 制定的视频编码标准一般称为建议，表示为 H.26x（例如 H.261、H.262、H.263 和 H.26L），而 ISO/IEC 制定的标准表示为 MPEG-x（例如 MPEG-1，MPEG-2，和 MPEG-4）。ITU-T 的标准主要用于实时视频通信，如视频电视会议、可视电话等。而 MPEG 标准主要用于广播电视、DVD 和视频流媒体。图 2-1 总结了 ITU-T 和 MPEG 标准发展过程。大多数情况下，这两个标准组织独立制定不同的标准，但 H.262 和 MPEG-2 是个例外，它是由两个组织联合制定的。目前，由 ITU-T 发起，两个组织再一次联手，共同制定了 H.26L 标准。该标准最初是由 ITU-T 的 Video Coding Experts Group (VCEG) 在 1997 年制定，到 2001 年底，ISO/IEC MPEG 和 ITU-T VCEG 联合组成了新的组织 Joint Video Team (JVT)，并接管了 ITU-T 的 H.26L 研究，到 2002 年底 H.26L 标准正式定案[11]。H.26L 的制定旨在提供更高的压缩效率，更灵活的网络适应性，以及增强的对于差错的鲁棒性。

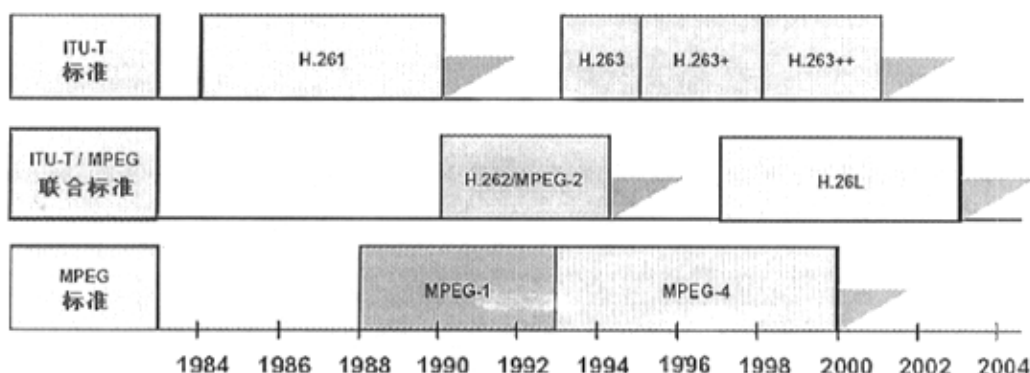


图 2-1 ITU-T 和 MPEG 标准发展过程

H.26L 是一种高效的压缩方法，它集中了以往标准的优点，并吸收了标准制定中积累的经验。与 H.263 v2 (H.263+) 或 MPEG-4 简单类 (Simple Profile) 相比，H.26L 在使用与上述编码方法类似的最佳编码器时，在大多数码率下最多可节省 50% 的码率。H.26L 在所有码率下都能持续提供较高的视频质量。H.26L 能工作在低延时模式以适应实时通信的应用（如视频会议），同时又能很好地工作在没有延时限制的应用，如视频存储和以服务器为基础的视频流式应用。H.26L 提供包传输网中处理包丢失所需的工具。H.26L 在系统层面上提出了一个新的概念，在视频编码层 (Video Coding Layer, VCL) 和网络适配层 (Network Adaptation Layer, NAL) 之间进行概念性分割，前者是视频内容的核心压缩内容之表述，后者是通过特定类型网络进行传送的表述。这样的结构便于信息的封装和对信息进行更好的优先级控制，如图 2-2 所示。



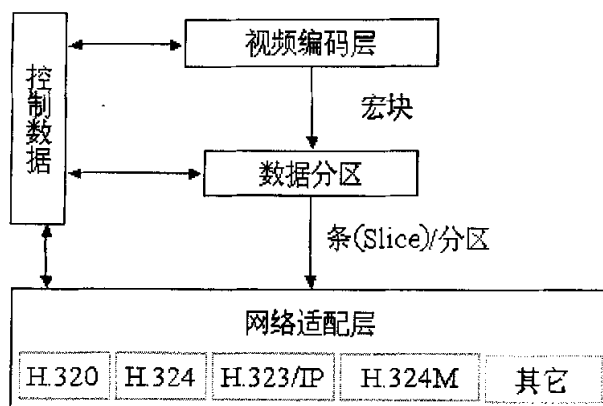


图 2-2 H.26L 系统层结构

## 2.1.2 H.26L 编码标准特点

### 2.1.2.1 H.26L 编解码流程

图 2-3 和图 2-4 所示分别为 H.26L 的编码和解码流程图。和以往的 H.263、MPEG-4 编码标准相比，H.26L 的编码流程图与之类似，只是在部分模块的特性上不同。唯一多的一个部分就是块滤波器。

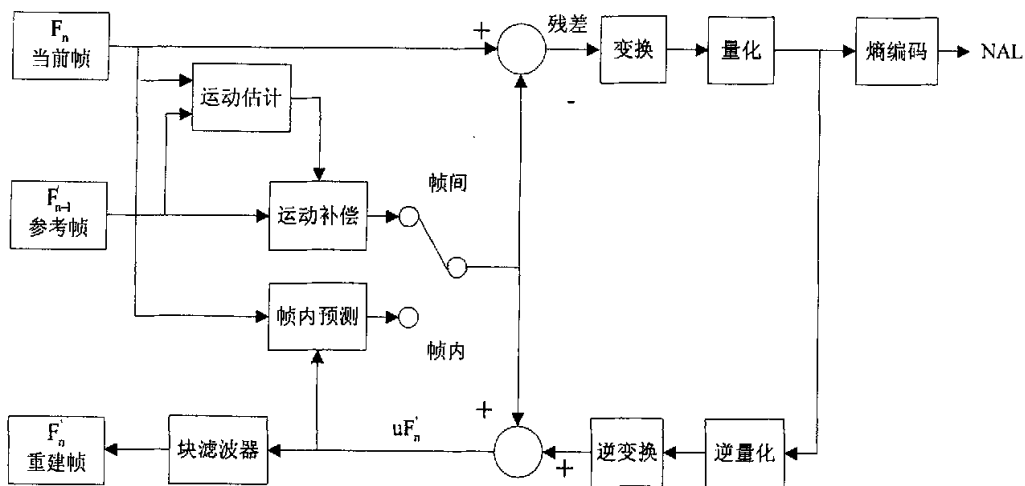


图 2-3 H.26L 编码流程图

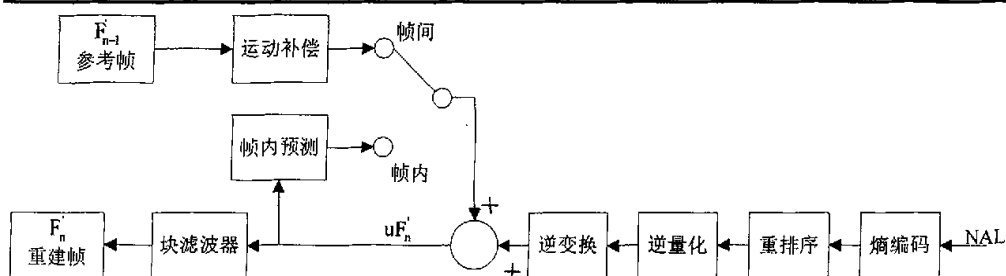


图 2-4 H.26L 解码流程图

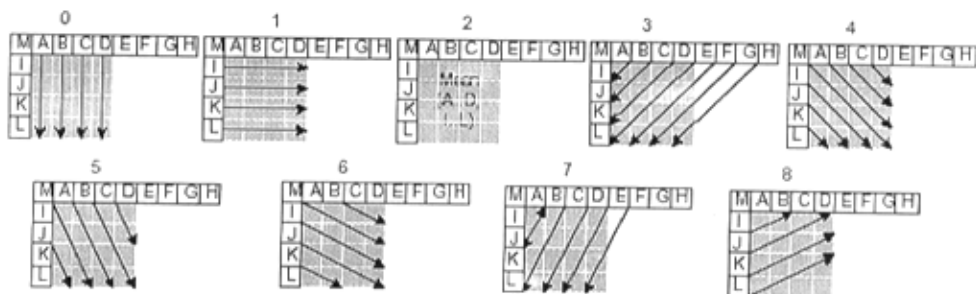
### 2.1.2.2 H.26L 编码支持格式

H.26L 支持的图像格式常用的有 CIF 和 QCIF, CIF 的图像为  $352 \times 288$  像素, 以  $16 \times 16$  像素为一个宏块分成  $22 \times 18$  共 396 个宏块, QCIF 图像是 CIF 图像的  $1/4$ , 即  $176 \times 144$  像素, 分成  $11 \times 9$  共 99 个宏块。对于其它大小的图像格式, 只有宽和高都是 16 的整数倍才符合 H.26L 的标准, 也就是必须能按照  $16 \times 16$  分割宏块。而且 H.26L 在编码时将一幅图像分成整数个像条 (Slice), 每个 Slice 分别编码, 有利于差错恢复。H.26L 与以往编码标准类似都有 I、P、B 帧的概念。I 帧即帧内编码帧, 解码不需要参考帧, 编码后数据量大, 帧数少, 每隔一段时间编码一帧, 终止误码扩散的最直接方法。P 帧和 B 帧只编码通过帧间预测得到的运动向量和残差, 解码都需要参考帧, 编码后数据量小, 大部分编码帧都是 P 帧和 B 帧。

### 2.1.2.3 H.26L 预测编码特点

#### 1. 帧内预测

H.26L 标准为了提高压缩效率在帧内编码时也提供多种预测编码模式。对于亮度块支持两种形式的帧内预测编码, 即  $4 \times 4$  和  $16 \times 16$  模式,  $4 \times 4$  模式适用于图像复杂的细节部分,  $16 \times 16$  模式适用于图像中变化不大的平坦部分。对于色度块 H.26L 只采用简单的  $4 \times 4$  预测模式。在亮度块  $4 \times 4$  预测模式中, 如图 2-5 所示, 将一个待预测  $4 \times 4$  块的上方和左边共 13 个采样像素作为预测参考进行预测, 预测方式共有 9 种。有的时候这 13 个参考采样像素不一定都存在于当前编码的 Slice 中, 此时只用存在于本 Slice 中的参考像素进行预测。在亮度块  $16 \times 16$  预测模式中, 提供四种预测模式: 垂直预测、水平预测、直流预测以及平面预测, 如图 2-6 所示。

图 2-5  $4 \times 4$  亮度块帧内预测模式

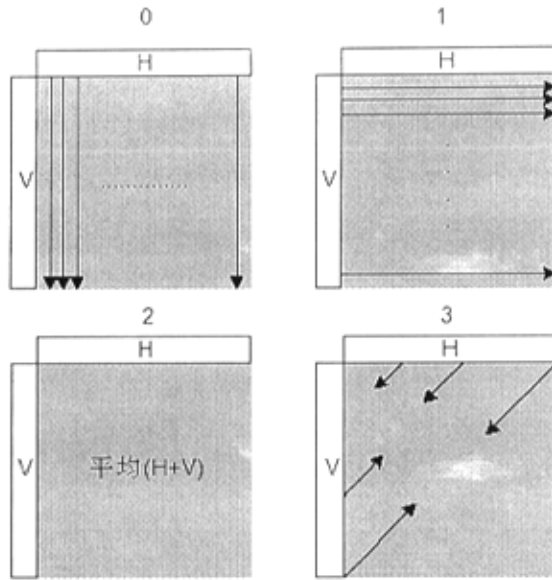


图 2-6 16×16 亮度块帧内预测模式

## 2. 帧间预测补偿

H.26L 在帧间编码中的运动补偿也很有特点，它支持运动补偿的块大小有很多种，从最大的 16×16 到最小的 4×4 亮度块之间有很多选择，如图 2-7 所示，上面一排是针对 16×16 宏块的四种分割，下面一排是对宏块分割后的 8×8 子块的子分割，最小分成 4×4 的块。这种分割方式使得一个宏块中可以用多种分割块进行组合，基于这种分割进行的运动补偿称为树形运动补偿，随着图像复杂细化的程度不断提高可以更加细化地分割进行运动补偿。每一个分割的块都有自己的运动向量，每个运动向量都被编码和传输，另外宏块的分割模式也必须被编码包含在压缩数据流中。当采用较大的分割块时（如 16×16，16×8，8×16）意味着块分割模式可以占用较少的比特，但是图像细节区域在运动补偿后的残差还具有较大的能量，意味着残差编码就需要较多的比特。反之，当采用较小的分割块时（如 8×4，4×4）运动补偿后的残差能量较小，需要编码的比特数少，但是表示块分割模式就会占用较多的比特数。因此分割模式的选取对编码性能的影响较大。一般而言，图像中大块的相似区域适合于大分割块运动补偿，而细节复杂区域适合于用小分割块。图 2-8 所示为一幅实际图像的块分割模式，可见越是细节区域分割越细，图像平坦区域就用大块分割。

H.26L 的运动补偿与以往的视频编码标准有一个区别就是它支持更高精度的运动向量，具体来说就是支持亮度块 1/4 像素精度的运动向量。由于 1/4 像素运动向量是小数，该运动向量所在位置没有实际的像素数据，所以需要通过相邻图像采样数据的插值获得。图 2-9 是整数像素和 1/4 像素运动向量预测示意图。由于编码每个分割块的运动向量也需要比特数，尤其当小块较多时占用比特数也较多。由于相邻块的运动向量具有较强的相关性，因此每个运动向量都用先解码的相邻块的运动向量作为参考向量进行预测，预测结果向量和当前运动向量的差再编码传输。

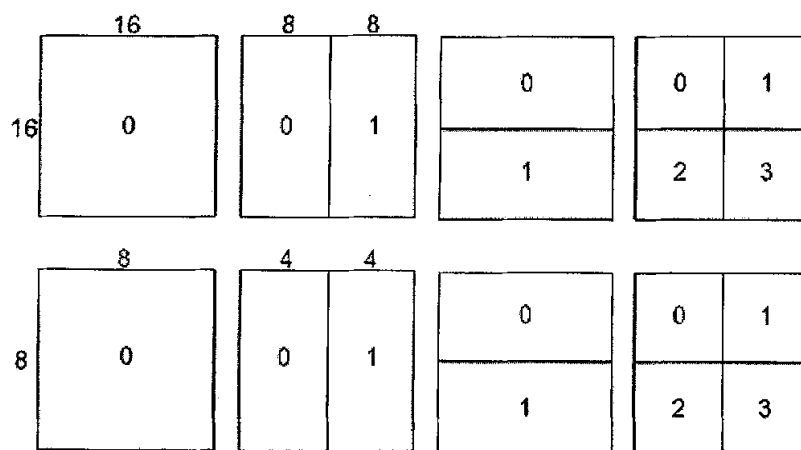


图 2-7 树形运动补偿块分割模式



图 2-8 实际图像运动补偿块分割示意图

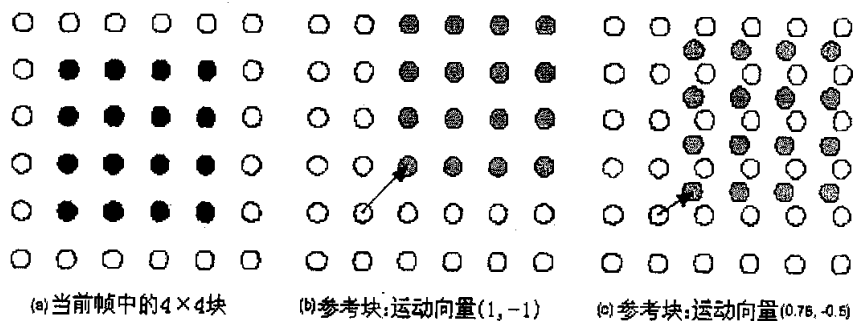


图 2-9 整数像素和 1/4 像素运动向量预测

H.26L 的帧间预测编码支持多参考帧，最多可以到 5 个参考帧，由此可以更有利于帧间预测，减少帧间预测所得残差编码的比特数，提高编码效率。如图 2-10 所示，第 3 帧以参考帧 1 作为参考帧更加合适，第 4 帧以参考帧 2 作为参考帧更加合适，预测所得残差更小。但是采用多参考帧会带来更大的处理延时并且需要更多的缓存空间。

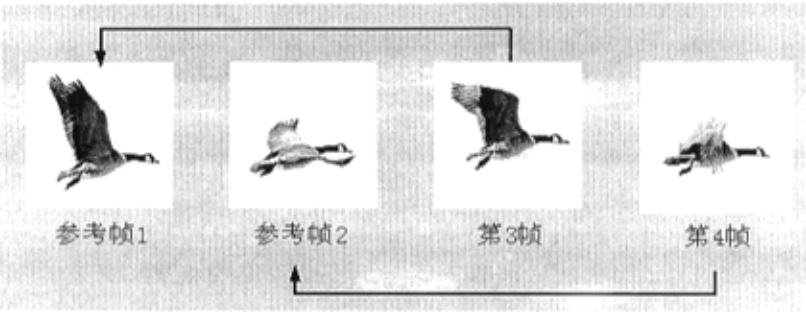


图 2-10 多参考帧示意图

H.26L 在预测环中加入了去块效应滤波器，对相邻块的边界像素进行滤波，从而降低块的边界效应，边界处连接更自然，图像更细腻，主观质量更好。

2.1.2.4 H.26L 变换编码特点

H.26L 和以往视频编码的不同之处还表现在变换编码部分。以前的编码标准如 MPEG-1、MPEG-2、MPEG-4、H.263 都使用  $8 \times 8$ DCT 变换作为基础变换编码。在 H.26L 中采用  $4 \times 4$  块的整数变换作为基本变换编码。 $4 \times 4$  整数变换编码与以往的  $8 \times 8$ DCT 变换编码相比具有以下优点：

(1) 减小方块效应：方块效应主要是由于变换编码并量化后，造成变换系数中的高频交流系数的损失而导致图像细节的丢失。H.26L 采用了更小的  $4 \times 4$  块进行变换，使得方块效应较  $8 \times 8$  更小。

(2) 算法简单明了，易于实现：H.26L 中  $4 \times 4$  块的变换与反变换均由图 2-11 所示的一次简单方程式构成，易于理解，便于实现。

$4 \times 4$ 变换	$4 \times 4$ 反变换
$A = 13a + 13b + 13c + 13d$	$a' = 13A + 17B + 13C + 7D$
$B = 17a + 7b - 7c - 17d$	$b' = 13A + 7B - 13C - 17D$
$C = 13a - 13b - 13c + 13d$	$c' = 13A - 7B - 13C + 17D$
$D = 7a - 17b + 17c - 7d$	$d' = 13A - 17B + 13C - 7D$

图 2-11  $4 \times 4$  整数变换公式

(3) 运算结果精度高且不会溢出：H.26L 中  $4 \times 4$  的变换与反变换公式中，系数均为整数，不像 H.263 中的  $8 \times 8$ DCT 变换公式中存在小数系数而在微机上造成偏差；并且  $4 \times 4$  整数变换编码与量化过程紧密结合，在数据不溢出的前提下，得到最大精度的计算结果。

(4) 运算速度快：由于 H.26L 中整数变换公式中的系数均为整数，可以用定点算法

实现，避免了浮点运算，提高了运算速度。

(5) 占用内存小：变换时，由于先行变换再列变换，用  $4 \times 4$  整数变换，内存一次同时只需保存  $4 \times 4 = 16$  个过渡值，不用像  $8 \times 8$  点 DCT 一次同时至少需要保存  $8 \times 8 = 64$  点计算值。

H.26L 对变换系数的扫描有两种方式，如图 2-12 所示，一种是以以往编码标准常用的 zigzag 扫描，另一种是双扫描，只在用较小量化步长对数据块进行帧内编码时使用，能得到更好的编码效率。

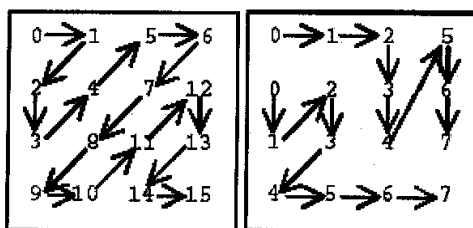


图 2-12 变换系数扫描方式

#### 2.1.2.5 H.26L 熵编码特点

视频编码的最后一步是熵编码，H.26L 采用两种熵编码方式，一种是基于通用变长编码 (UVLC)，另一种是基于上下文的自适应二进制算术编码 (CABAC)。

基于 VLC 编码是使用比较广泛的熵编码，它对量化后的变换系数、运动向量以及其它编码信息采用不同的统计码表进行编码。而 H.26L 使用的 UVLC 只使用一个码表，其优点是简单；缺点是单一的码表是从概率统计分布模型得出的，没有考虑编码符号间的相关性，在中高码率时效果不是很好。

算术编码使编码和解码两边都能使用所有句法元素（变换系数、运动矢量）的概率模型。为了提高算术编码的效率，H.26L 采用的 CABAC 通过基于上下文内容建模的过程，使基本概率模型能适应随视频帧而改变的统计特性。基于上下文内容建模提供了编码符号的条件概率估计，利用合适的内容模型，存在于符号间的相关性可以通过选择目前要编码符号邻近的已编码符号的相应概率模型来去除，不同的句法元素通常保持不同的模型。如果给定的符号为非二进制值，它将按照一个二进制判决的顺序进行转换，称为二进制化 (Bins)。实际的二进制化是根据给定的二进制树完成的，这里使用的是 UVLC 二进制树。每一次二进制判决后，算术编码器使用新的概率估计进行编码，此概率估计是在前一上下文内容建模阶段更新的。在每一次二进制化编码以后，向上调整用于刚刚编码的二进制符号的概率估计，这样，模型就保持了对实际统计特性的跟踪。H.26L 中利用了新方法来自编码变换系数，并利用查表法来估计概率，提高了编码效率和速率。

#### 2.1.2.6 H.26L 特点比较

综上所述，H.26L 编码标准和以往的视频编码标准相比很多地方都有改进，进一步提高了编码效率，下面将 H.26L 和 MPEG-4 进行一下对比，如表 2-1 所示。



表 2-1 H.26L 和 MPEG-4 对比

	H.26L	MPEG-4
帧内预测	9 种 4×4 块预测模式 四种 16×16 宏块预测模式 (预测更精确, 编码效率更高)	四种 16×16 宏块预测模式 (预测较简单, 编码效率低)
帧间预测	运动补偿块的大小范围从 16×16 到 4×4 有多种选择 1/4 像素精度运动预测 (预测更精确, 编码效率更高)	16×16 和 8×8 块运动预测 1/2 像素精度运动预测 (预测较简单, 编码效率低)
参考帧	支持最多 5 个参考帧	只支持单参考帧
块滤波器	有 (改善图像质量, 提高压缩率)	无
变换编码	4×4 无乘法整数 DCT 变换 (算法简单, 速度快, 无计算偏差)	8×8 含乘法小数 DCT 变换 (计算复杂, 有小数偏差)
熵编码	UVLC, CABAC (高压缩率)	VLC

§ 2.2 H.26L 差错控制技术

H.26L 综合运用差错恢复和差错隐藏技术实现差错控制。通过重同步、数据分区、帧内宏块刷新等差错恢复技术控制差错范围, 限制误码扩散, 为差错隐藏奠定基础, 差错隐藏则根据通过差错恢复得到的数据用填补掩盖等办法恢复不能正常解码的数据, 尽最大努力保证视频主观效果。整个差错控制流程如图 2-13 所示。

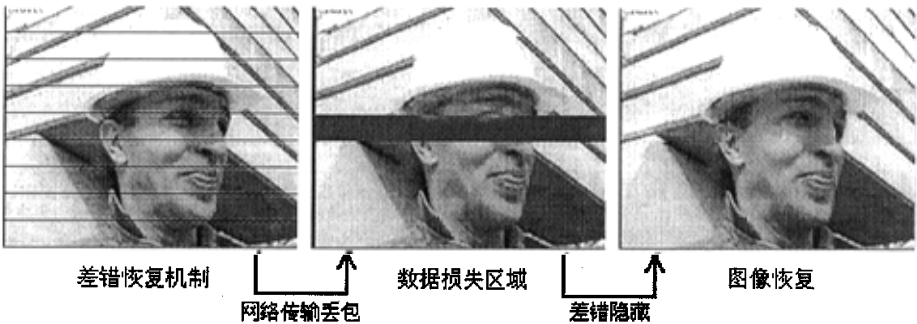


图 2-13 差错控制流程图

2.2.1 H.26L 的差错恢复技术

H.26L 中具有支持差错恢复的相关技术, 主要包括重同步、数据分区、帧内宏块刷新三种:

1. H.26L 中的重同步分为时间重同步和空间重同步。时间重同步是指每隔一定时间编码一帧 I 帧, 从而结束对上一个 I 帧的预测, 重新以当前 I 帧作为参考帧来预测后续图像。空间重同步是指对每一帧图像可以分成多个 Slice 进行编码, 如图 2-13 第一幅图就是对 QCIF 格式的图像进行的 Slice 分割。一般来说每一个 Slice 由一帧图像中一行宏块组成, 每一个 Slice 是一个编码单元, 误码扩散只会限制在一个 Slice 中, 不会跨越 Slice。
2. 数据分区将不同类型的编码数据分成不同的区存储, 每一个 Slice 分成三个区, 一个区存放所有编码宏块的头信息, 一个区存放帧内编码块的编码模式和变换系数, 还有一

表 2-1 H.26L 和 MPEG-4 对比

	H.26L	MPEG-4
帧内预测	9 种 4×4 块预测模式 四种 16×16 宏块预测模式 (预测更精确, 编码效率更高)	四种 16×16 宏块预测模式 (预测较简单, 编码效率低)
帧间预测	运动补偿块的大小范围从 16×16 到 4×4 有多种选择 1/4 像素精度运动预测 (预测更精确, 编码效率更高)	16×16 和 8×8 块运动预测 1/2 像素精度运动预测 (预测较简单, 编码效率低)
参考帧	支持最多 5 个参考帧	只支持单参考帧
块滤波器	有 (改善图像质量, 提高压缩率)	无
变换编码	4×4 无乘法整数 DCT 变换 (算法简单, 速度快, 无计算偏差)	8×8 含乘法小数 DCT 变换 (计算复杂, 有小数偏差)
熵编码	UVLC, CABAC (高压缩率)	VLC

§ 2.2 H.26L 差错控制技术

H.26L 综合运用差错恢复和差错隐藏技术实现差错控制。通过重同步、数据分区、帧内宏块刷新等差错恢复技术控制差错范围, 限制误码扩散, 为差错隐藏奠定基础, 差错隐藏则根据通过差错恢复得到的数据用填补掩盖等办法恢复不能正常解码的数据, 尽最大努力保证视频主观效果。整个差错控制流程如图 2-13 所示。

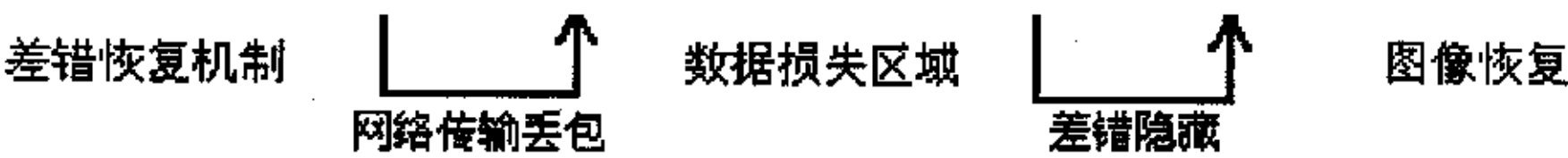


图 2-13 差错控制流程图

2.2.1 H.26L 的差错恢复技术

H.26L 中具有支持差错恢复的相关技术, 主要包括重同步、数据分区、帧内宏块刷新三种:

1. H.26L 中的重同步分为时间重同步和空间重同步。时间重同步是指每隔一定时间编码一帧 I 帧, 从而结束对上一个 I 帧的预测, 重新以当前 I 帧作为参考帧来预测后续图像。空间重同步是指对每一帧图像可以分成多个 Slice 进行编码, 如图 2-13 第一幅图就是对 QCIF 格式的图像进行的 Slice 分割。一般来说每一个 Slice 由一帧图像中一行宏块组成, 每一个 Slice 是一个编码单元, 误码扩散只会限制在一个 Slice 中, 不会跨越 Slice。
2. 数据分区将不同类型的编码数据分成不同的区存储, 每一个 Slice 分成三个区, 一个区存放所有编码宏块的头信息, 一个区存放帧内编码块的编码模式和变换系数, 还有一

个区存放帧间编码块的编码模式和变换系数。通过数据分区可以将每一个区的数据放在一个数据包中传输,按照数据重要性划分数据的优先级,运用非平衡保护措施重点保护优先级高的数据包。而且解码端可以根据出错数据的类型采用不同的差错处理方法,以达到最好的差错控制效果。

3. 帧内宏块刷新是指在编码过程中,对于 P 帧和 B 帧按照一定的频率每隔一定数量的宏块就采用帧内编码方式编码一个宏块,使 P 帧和 B 帧中每过一段时间也会出现一个帧内编码的宏块。这样在出现误码扩散时,帧内编码宏块可以在一定程度上减少误码扩散,降低误码扩散带来的视频质量下降。

以上三种差错恢复技术都有一个共同的缺点就是会增加额外的数据率,降低编码效率。I 帧的数量越多,显然编码数据比特率越大;划分 Slice 进行编码,每个 Slice 的头信息要花费更多的比特率;数据分区将不同的数据类型分区分数据包存放,多出来的数据包有更多的数据包头信息,还要花费更多区分数据类型的比特;帧内宏块刷新和增加 I 帧本质上相同,只是比特率增加程度和图像改善程度都不及增加 I 帧。一般来说, I 帧数量可以设定每编码多少帧编码一个 I 帧,一般不做变化; Slice 划分方式和数据分区方式作为编码参数在编码前输入;帧内宏块刷新根据编码数据率限制以及质量要求而动态改变。

差错恢复技术的引入虽然带来了一些冗余,但是对视频传输的差错控制也是很有好处的。它使差错总是限定在一定范围之内,误码扩散的影响随之受到了限制,帧内宏块刷新也使误码扩散的影响尽快消除,改善了解码端的视频质量。

## 2.2.2 H.26L 的差错隐藏技术

差错恢复技术能限制和确定数据出错的范围,但是还必须借助差错隐藏技术对丢失的信息进行估计进而恢复。由于人眼具有容错性,不是十分明显的错误人一般不会有明显感觉,而且由于图像序列本身就有很强的空间和时间上的相关性,这为用正确接收到的视频数据去预测丢失的数据提供了可能。通过对丢失数据进行填补,使视频质量恢复到人眼可以接受的地步。这对于较大数据量丢失是一种比较好的控制方式。

### 2.2.2.1 差错状态图

一般来说,将图像数据划分成 Slice,每一个 Slice 存于一个 NAL 单元中,在网络传输中丢包导致 NAL 单元中的 Slice 数据丢失。所有正确接收的 Slice 先解码,然后对丢失的 Slice 按照特定算法进行差错隐藏。差错隐藏是基于宏块进行处理,为了区分正确的宏块和丢失的宏块,引入了差错状态图(Error States Map),如图 2-14 所示。对于一帧接收到的数据进行解码时,能够正确解码的宏块数据在差错状态图中标记为 OK,反之标记为 Corrupted,解码后如果差错状态图中有标记为 Corrupted 的宏块,则进行差错隐藏,成功进行差错隐藏处理后将这些宏块标记为 Concealed。进行差错隐藏时,如果某一个 Corrupted 宏块没有正确解码的相邻宏块,则将经过差错隐藏处理后标记为 Concealed 的相邻宏块也当作可以信赖的宏块,参与差错隐藏。这种情况下,标记为 Concealed 的宏块在差错隐藏中的误差就会影响到相邻宏块的差错隐藏,造成误码扩散。因此,Corrupted 宏块进行差错隐藏的顺序就很重要。首先从图像边缘的宏块开始处理,然后向内部一行一行地处理,这种处理顺序避免了先对处理相对困难的内部数据进行差错隐藏时造成的更大可能的误码扩散。

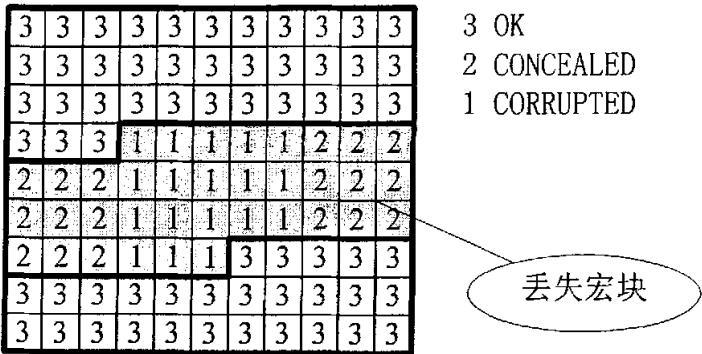


图 2-14 差错状态图

2.2.2.2 帧内差错隐藏

对于 I 帧的差错隐藏由于相邻 I 帧之间差异较大，所以采用空域差错隐藏。H.26L 标准中采用的空域差错隐藏算法是基于像素距离的加权平均，其权值反比于源像素和目的像素之间的距离[12]。如果相邻宏块中有 2 个以上标记为 OK，则只用标记为 OK 的相邻宏块进行加权平均计算，否则相邻宏块中标记为 Concealed 宏块也用于加权平均计算[13]。如图 2-15 所示为以亮度宏块为例的像素距离加权平均算法示例。C 的亮度值  $Y_c$  按照公式 2-1 计算，其中  $\{W_T, W_B, W_L, W_R\}$  分别为 T、B、L、R 四个像素的距离权值，由于权值反比于源像素和目的像素之间的距离，所以  $W_T$  实际上是 C 和 B 之间的距离，而  $W_B$  实际上是 C 和 T 之间的距离，对  $W_L$  和  $W_R$  同理。 $\{Y_T, Y_B, Y_L, Y_R\}$  分别为 T、B、L、R 四个像素的亮度值。公式 2-2 是  $\{W_T, W_B, W_L, W_R\}$  的计算公式。

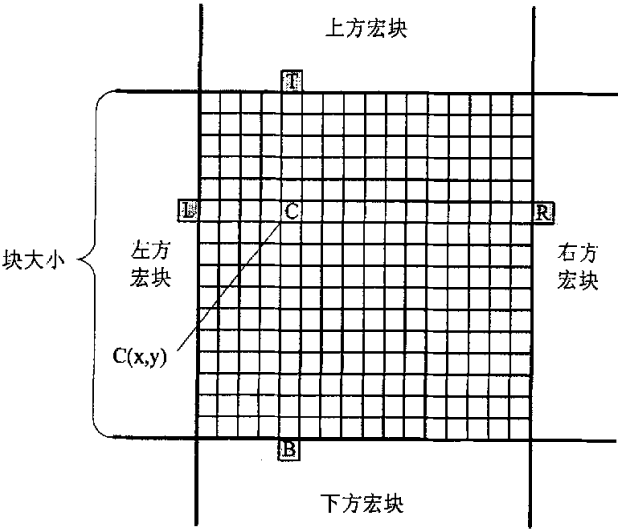


图 2-15 像素距离加权平均算法示例

$$Y_c = \frac{W_T \times Y_T + W_B \times Y_B + W_L \times Y_L + W_R \times Y_R}{W_T + W_B + W_L + W_R} \tag{2-1}$$

$$W_T = BSize - y, W_B = y + 1, W_L = BSize - x, W_R = x + 1 \quad (2-2)$$

### 2.2.2.3 帧间差错隐藏

对于预测编码帧进行差错隐藏的一个有效的措施就是根据空域或时域相邻宏块的运动向量运用预测算法“猜测”丢失宏块的运动向量,然后利用“猜测”出的运动向量进行参考帧的运动补偿,恢复丢失的宏块。首先对标记为 OK 的宏块的运动向量进行预测计算,如果运动向量平均值小于一个阈值(比如  $1/4$  像素),则认为图像运动幅度很小,丢失的宏块通过复制参考帧中相应位置的宏块数据实现差错隐藏。如果运动向量平均值大于阈值,就对丢失宏块的运动向量进行预测,从相邻块的候选运动向量中选取最合适的一个运动向量作为预测结果。采用这种方法的根据是通过大量统计观察得出的规律,即空间相邻的块的运动具有很高的相关性。而且通过以往的实验可以得到另一个规律,就是丢失宏块的运动向量大多数情况下与某一个相邻宏块的运动向量更加匹配,而用所有相邻宏块的运动向量的平均值作为丢失宏块的运动向量效果却不好。为了确定用哪一个相邻块的运动向量作为预测运动向量,以恢复图像最大平滑度为标准进行计算。对每一个候选运动向量,用此运动向量进行运动补偿,计算恢复宏块区域边界上亮度值的差,使差值最小的运动向量就作为最终的预测运动向量[12],如图 2-16 所示,同时零向量总是作为后选运动向量之一参与计算。边界差值按照公式 2-3 进行计算。 $N$  是边界上像素总数,  $\hat{Y}(mv^{dir})_j^N$  是以  $mv^{dir}$  进行运动补偿得到的恢复宏块的内边界第  $j$  个亮度值,  $Y_j^{OUT}$  是相对应的外边界第  $j$  个亮度值,  $d_{sm}$  是计算得到的边界差绝对值的平均值。使  $d_{sm}$  最小的运动向量就为最后的预测结果,用此运动向量进行运动补偿恢复丢失宏块。如果相邻宏块有正确接收的,则在计算  $d_{sm}$  时只对正确接收宏块的边界进行匹配计算。

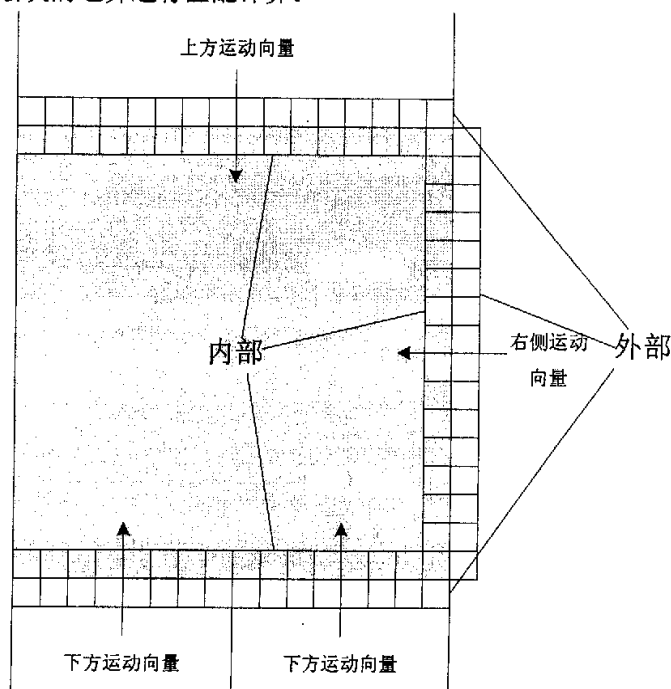


图 2-16 运动向量预测及边界差值计算

$$\min_{\arg \left\{ d_{sm} = \frac{1}{N} \sum_{j=1}^N \left| \hat{Y}(mv^{dir})_j^{IN} - Y_j^{OUT} \right| \right\}} \quad (2-3)$$

### § 2.3 差错隐藏技术仿真实验及改进

#### 2.3.1 仿真实验

仿真实验使用的测试软件是 H.26L 的参考软件 JM61[14], 该软件是 JVT 组织提供的一个开放源码软件, 实现了 H.26L 基本编解码过程。

网络环境是模仿 H.323/Internet 环境下的数据包丢失情况[15], 测试的仿真网络丢包率分别为 3%, 5%, 10%和 20%。丢失数据包的仿真利用了一个仿真丢包文件, 该文件实际上是一个字符文件, 每一个字符对应一个网络数据包。它是通过大量实际网络传输得到的字符集, 如果网络中实际的数据包可以正确接收, 则文件中存储一个非 0 字符, 如果实际数据包丢失则存储一个 0 字符, 由此得到一个记录了大量连续字符的文件, 并以此来模拟实际网络丢包状况。仿真丢包文件在实验中使用如图 2-17 所示, 将视频编码文件与仿真丢包文件相对应, 0 字符对应的编码数据包被丢弃, 非 0 字符对应的数据包保留, 由此构成一个新的视频编码文件, 即仿真丢包后收到的视频数据, 再进行解码和差错隐藏。



图 2-17 仿真丢包文件的使用

通过计算测试视频序列的平均 PSNR 值来比较客观效果, PSNR 的计算是在解码数据帧和源数据帧之间进行, 计算公式如下所示

$$20 \log_{10} \frac{255N}{\sqrt{\sum [f(i,j) - F(i,j)]^2}} \quad (2-4)$$

$N$  是宏块长宽的像素数, 例如亮度块是 16, 色度块是 8;  $f(i,j)$  是源宏块像素数据,  $F(i,j)$  是解码重建后的宏块像素数据。

实验时选取两个 QCIF 格式测试视频序列, 一个是 Foreman 序列, 另一个是 Coastguard 序列。对编码数据采用 RTP 数据包方式进行组织, 对每一帧图像进行了 Slice 分割, 每一帧分成 9 个 Slice, 每个 Slice 的编码数据存入一个 RTP 数据包。通过网络仿真模拟丢失 RTP 数据包, 对丢失的 Slice 数据进行差错隐藏恢复, 与之相对比的是不进行差错隐藏, 只是单纯的复制参考帧中的相应位置的宏块。

实验所得到的仿真结果数据对比如表 2-2 所示。除了从客观评价标准上进行分析对比之外, 从主观感受上也能发现使用差错隐藏比不使用差错隐藏在大多数情况下差错恢复效果要好, 具体如图 2-18 和 2-19 所示, 上面一排是没有使用差错隐藏解码的图像, 下面一排是相应的使用差错隐藏解码的图像, 可见上面的图像在丢失数据区边缘有明显的错位现象, 这是因为直接复制参考帧中数据没有考虑到运动, 图像运动越剧烈, 这种错位越明显, 如 Foreman 序列中上面一排第三幅图。



表 2-2 差错隐藏仿真结果 PSNR 平均值对比, 帧率为 30 帧 / 秒

Foreman 256kbps(30f/s)	Packet Loss Rate				Coastguard 128kbps(30f/s)	Packet Loss Rate			
	3%	5%	10%	20%		3%	5%	10%	20%
直接复制	32.01	30.09	26.81	21.58	直接复制	29.31	28.32	25.76	22.25
差错隐藏	32.39	30.77	27.81	23.04	差错隐藏	29.68	28.32	25.83	22.35

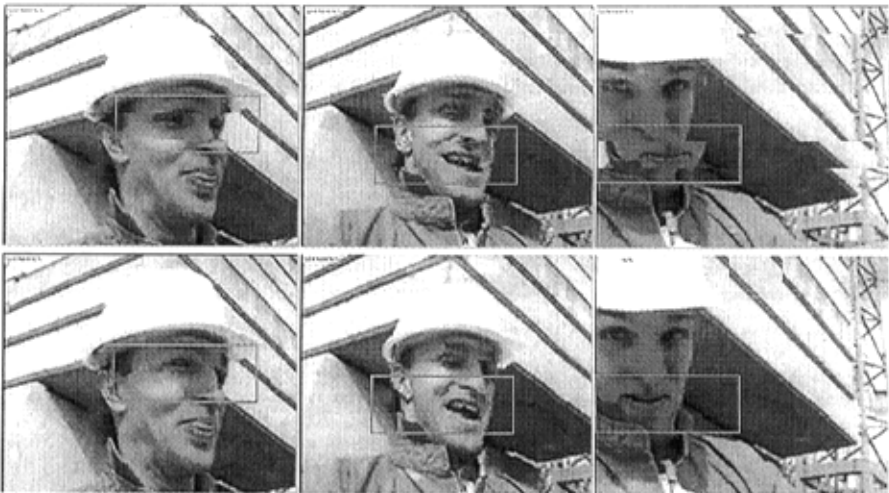


图 2-18 Foreman 序列差错隐藏主观图像恢复效果比较, 丢包率 5%

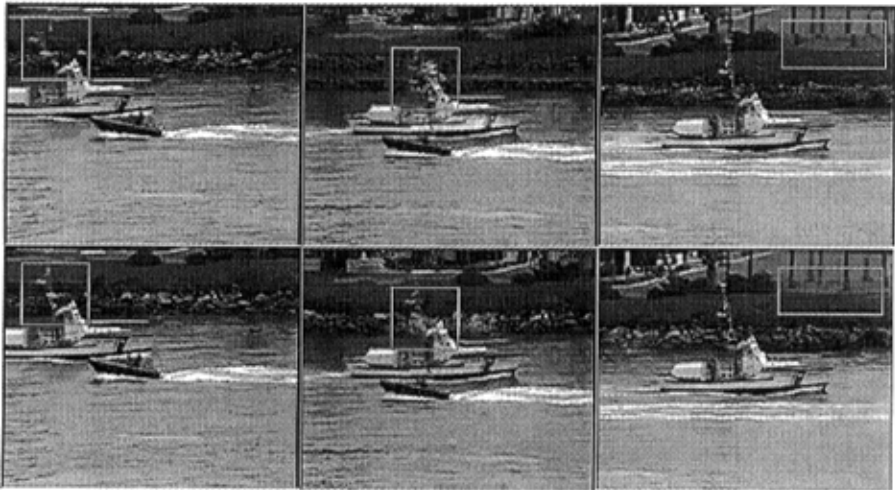


图 2-19 Coastguard 序列差错隐藏主观图像恢复效果比较, 丢包率 5%

2.3.2 改进方案

尽管采用差错隐藏解码在大多数情况下图像恢复效果比不采用差错隐藏要好, 但是也不排除有的情况下效果相反。当丢失 Slice 包的帧为 I 帧时, 差错隐藏会用像素平均法进行差错隐藏, 由图 2-20 和 2-21 可以看出, 此时使用差错隐藏得到的图像效果反而不是很

好, 因为丢失的数据包含有的数据量较多, 而使用像素平均实际上是将丢失数据区域模糊化, 对大范围的图像数据进行模糊处理, 实际效果肯定不会好。而且由于是 I 帧, 其它数据帧以它为参考进行编码, 因此会影响以后的解码, 造成误码扩散, 使后续解码图像也有很大的模糊区域, 如图 2-20 和 2-21 下面一排后一幅图。此时如果不使用像素平均法, 而是将前一个参考帧的相应区域复制到丢失区域, 虽然在丢失区域边缘有分裂的现象, 但总体效果反而更好。



图 2-20 Foreman 序列中 I 帧丢失 Slice 时差错隐藏效果

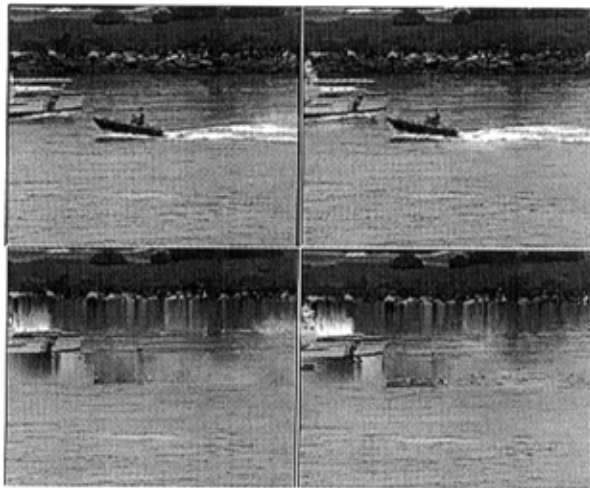


图 2-21 Coastguard 序列中 I 帧丢失 Slice 时差错隐藏效果

根据以上情况, 本文对差错隐藏解码策略进行了改进。以往的视频解码都是 P 帧以 I 帧为参考进行解码, 而在 I 帧出错时影响很大。反过来考虑当 P 帧能正常解码, 而且错误比较少, 图像质量比较高时, 以它作为错误 I 帧的参考对错误 I 帧进行恢复似乎效果更好。虽然和常理有些不同, 但是通过实验的效果发现这种方法是很有效的。

改进策略如下:

- (1) 整个图像序列第一帧 (肯定为 I 帧) 如果丢失 Slice, 则用像素平均法进行差错

隐藏。因为此时不可能有前向参考帧可以使用复制方式进行差错恢复。

(2) 除第一个 I 帧外，如果在图像序列当中其它 I 帧有丢失 Slice，就使用前一个可以正常解码的 P 帧为参考帧，将参考帧中相应区域的数据复制到当前丢失区域以恢复当前丢失的 Slice。

通过仿真实验表明采用本文提出的改进差错隐藏解码策略，图像质量的恢复有了较大提高，仿真结果平均 PSNR 值对比如表 2-3 所示。另外，通过测试不同数据率下图像差错隐藏的效果，得到了如图 2-21 所示的对比结果，可见改进的差错隐藏解码策略也是适用于各种数据率情况的，在各种数据率下一般都有 0.3-0.7dB 的性能提升。

表 2-3 改进差错隐藏性能对比

Foreman 256kbps(30f/s)	Packet Loss Rate				Coastguard 128kbps(30f/s)	Packet Loss Rate			
	3%	5%	10%	20%		3%	5%	10%	20%
直接复制	32.01	30.09	26.81	21.58	直接复制	29.31	28.32	25.76	22.25
差错隐藏	32.39	30.77	27.81	23.04	差错隐藏	29.68	28.32	25.83	22.35
改进差错隐藏	33.32	31.54	28.40	23.82	改进差错隐藏	29.85	29.09	26.80	22.99

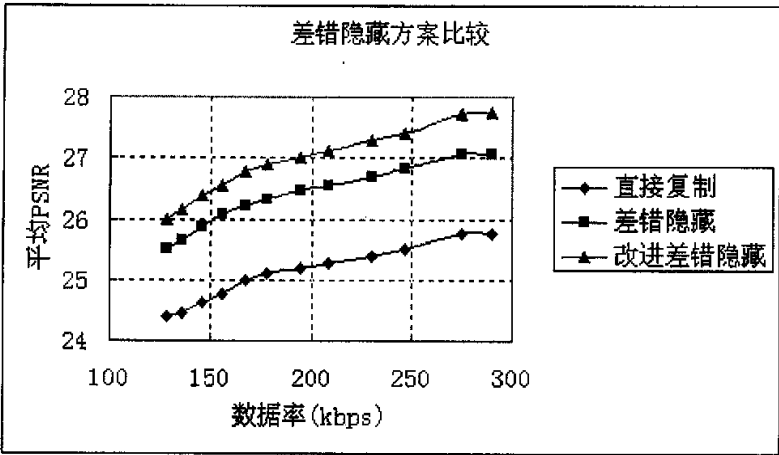


图 2-22 各种数据率下差错隐藏恢复图像平均 PSNR 比较

§ 2.4 小结

本章介绍了 H.26L 标准中最基本的差错恢复和差错隐藏技术。仿真实验的结果显示了对于丢失数据包情况下主观和客观上视频质量的改善。根据实验结果还能发现 H.26L 标准中的差错隐藏算法还有一定缺点，因此提出了改进建议，取得了较好的改进结果。但是差错隐藏适合于大量数据丢失的情况，例如网络丢包，而对于网络上经常出现的各种随机错和突发错，它并不适合，因此本文接下来研究了信道编码，它对于处理随机错和突发错有很好的效果。

隐藏。因为此时不可能有前向参考帧可以使用复制方式进行差错恢复。

(2) 除第一个 I 帧外, 如果在图像序列当中其它 I 帧有丢失 Slice, 就使用前一个可以正常解码的 P 帧为参考帧, 将参考帧中相应区域的数据复制到当前丢失区域以恢复当前丢失的 Slice。

通过仿真实验表明采用本文提出的改进差错隐藏解码策略, 图像质量的恢复有了较大提高, 仿真结果平均 PSNR 值对比如表 2-3 所示。另外, 通过测试不同数据率下图像差错隐藏的效果, 得到了如图 2-21 所示的对比结果, 可见改进的差错隐藏解码策略也是适用于各种数据率情况的, 在各种数据率下一般都有 0.3-0.7dB 的性能提升。

表 2-3 改进差错隐藏性能对比

Foreman 256kbps(30f/s)	Packet Loss Rate				Coastguard 128kbps(30f/s)	Packet Loss Rate			
	3%	5%	10%	20%		3%	5%	10%	20%
直接复制	32.01	30.09	26.81	21.58	直接复制	29.31	28.32	25.76	22.25
差错隐藏	32.39	30.77	27.81	23.04	差错隐藏	29.68	28.32	25.83	22.35
改进差错隐藏	33.32	31.54	28.40	23.82	改进差错隐藏	29.85	29.09	26.80	22.99

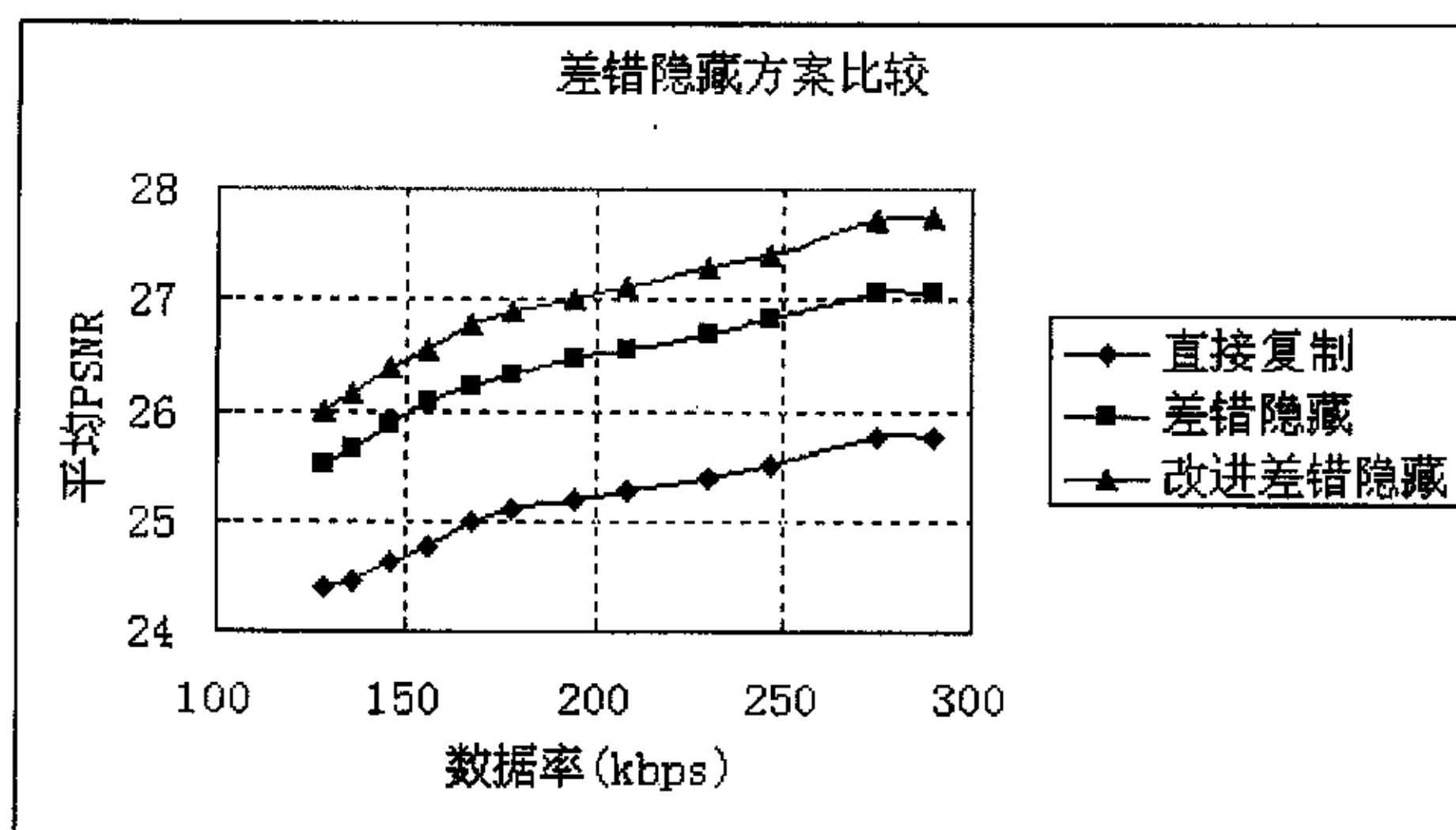


图 2-22 各种数据率下差错隐藏恢复图像平均 PSNR 比较

## § 2.4 小结

本章介绍了 H.26L 标准中最基本的差错恢复和差错隐藏技术。仿真实验的结果显示了对于丢失数据包情况下主观和客观上视频质量的改善。根据实验结果还能发现 H.26L 标准中的差错隐藏算法还有一定缺点, 因此提出了改进建议, 取得了较好的改进结果。但是差错隐藏适合于大量数据丢失的情况, 例如网络丢包, 而对于网络上经常出现的各种随机错和突发错, 它并不适合, 因此本文接下来研究了信道编码, 它对于处理随机错和突发错有很好的效果。

## 第三章 信道编码技术—RS 码研究与实现

### § 3.1 信道编码介绍

#### 3.1.1 信道编码概述

数据在网络中传输时可能由于线路故障或外界影响产生噪声，噪声作用于数据就形成差错。噪声可分为两类，一类是热噪声，另一类是冲击噪声，热噪声引起的差错是一种随机差错，亦即某个码元的出错具有独立性，与前后码元无关。冲击噪声是由短暂原因造成的，例如电机的启动、停止，电器设备的放弧等，冲击噪声引起的差错称为突发错，其差错持续时间称为突发错的长度。以保证数字信息传输和处理的可靠性为目的的差错控制编码就是信道编码，常称为纠错编码。

信道编码在数字通信系统中的位置如图 3-1 所示。使信源编码器输出的数字序列  $M$  经过信道编码器后输出具有某种规律性的数字序列  $C$ ，当  $C$  经过调制解调器和信道（传输媒体）传输后，得到可能有差错的数字序列  $C'$ ，而信道译码器能发现或者纠正在序列  $C'$  中出现的错误，使得根据  $C'$  得到的  $M'$  尽可能与  $M$  相等。采用信道编码方式时，不需要反馈信道，也无需反复重发而延误传输时间，对实时传输有利，特别适合于视频点播、电视会议等实时性要求比较高的应用中的差错控制。

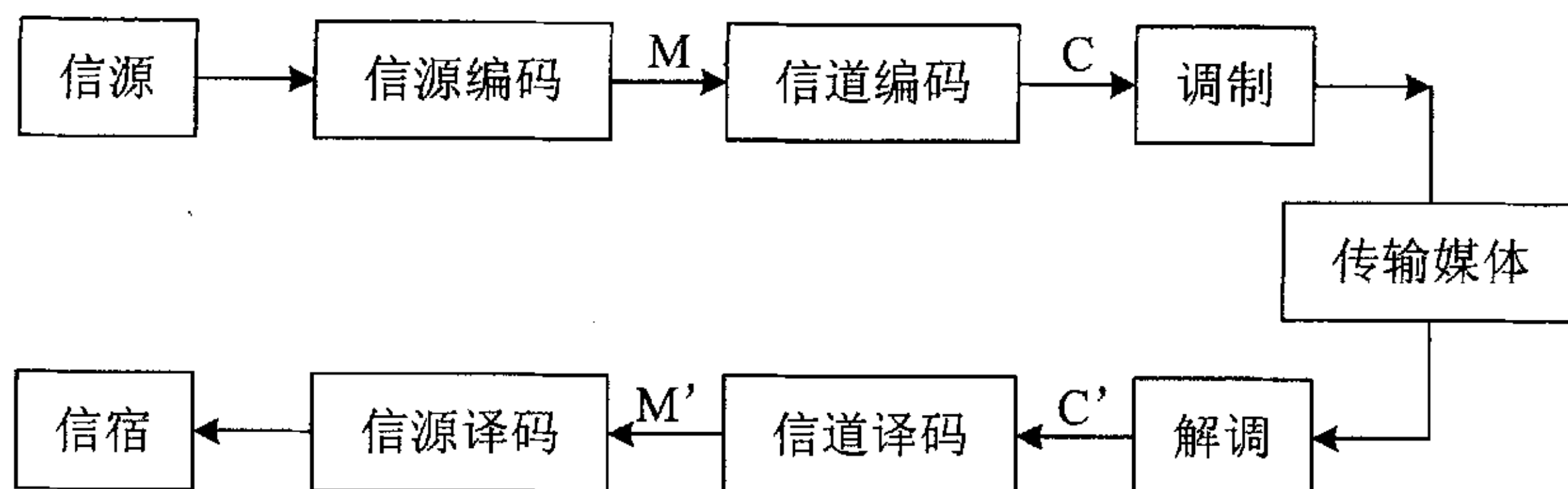


图 3-1 信道编码在数字通信系统中的位置

信道编码主要是为了发现和减少通信中的传输差错，因此，差错率是一个重要的参数。其定义有如下几种：

- (1) 码元差错率  $P_e$ ：发生差错的码元数与被传输的码元总数之比，简称误码率。
- (2) 比特差错率  $P_{be}$ ：发生差错的比特数与被传输的比特总数之比，简称误比特率（BER, Bit Error Rate）。在二进制传输系统中， $P_{be} = P_e$ 。

(3) 码组差错率：发生差错的码组数与被传输的码组总数之比。帧差错率（FER, Frame Error Rate）即误帧率，在计算机网络通信中常用来衡量信道性能和纠错能力，它属于码组差错率的一种。

显然，信源编码的中心任务是消除冗余，实现码率压缩，可是为了检错与纠错，又不得不增加冗余，这又必然导致码率增加，传输效率降低；显然这是个矛盾。分析信道编码的目的，正是为了寻求较好的编码方式，能在增加冗余不太多的前提下来实现检错和纠错。

### 3.1.2 信道编码基本概念

#### (1) 信息码元与监督码元:

信息码元又称信息序列或信息位, 这是发端由信源编码后得到的被传送的信息数据比特, 通常以  $k$  表示。由信息码元组成的信息组为:  $M = (m_{k-1}, m_{k-2}, \dots, m_0)$ , 在二码情况下, 每个信息码元  $m$  的取值只有 0 或 1, 故总的信息码组数共有  $2^k$  个, 即不同信息码元取值的组合共有  $2^k$  组。

监督码元又称监督位或附加数据比特, 这是为了检纠错码而在信道编码时加入的判断数据位。通常以  $r$  表示, 即为:  $n=k+r$  或  $r=n-k$ 。

经过分组编码后的码又称为  $(n, k)$  码, 即表示总码长为  $n$  位, 其中信息码长 (码元数) 为  $k$  位, 监督码长 (码元数) 为  $r=n-k$ , 通常称其为长为  $n$  的码字 (或码组、码矢)。

#### (2) 许用码组与禁用码组

信道编码后的总码长为  $n$ , 总的码组数应为  $2^n$ , 即为  $2^{k+r}$ 。其中被传送的信息码组有  $2^k$  个, 通常称为许用码组; 其余的码组共有  $(2^n - 2^k)$  个, 不传送, 称为禁用码组。发端误码控制编码的任务正是寻求某种规则从总码组  $2^n$  中选出许用码组; 而收端译码的任务则是利用相应的规则来判断及校正收到的码字符合许用码组。通常又把信息码元数目  $k$  与编码后的总码元数目 (码组长度)  $n$  之比称为信道编码的编码效率或编码速率, 表示为:  $R = k/n = k/(k+r)$ , 这是衡量信道编码性能的一个重要指标, 一般情况下, 监督位越多 (即  $r$  越大), 检纠错能力越强, 但相应的编码效率也随之降低了。

#### (3) 码重与码距

在分组编码后, 每个码组中码元为 “1” 的数目称为码的重量, 简称码重。两个码组对应位置上取值不同 (1 或 0) 的位数, 称为码组的距离, 简称码距, 又称汉明距离, 通常用  $d$  表示。例如: 000 与 101 之间码距  $d=2$ ; 000 与 111 之间码距  $d=3$ 。对于  $(n, k)$  码, 许用码组为  $2^k$  个, 各码组之间距离最小值称为最小码距, 通常用  $d_0$  表示。最小码距  $d_0$  的大小与信道编码的检纠错能力密切相关, 主要有以下三条。

在一个码组内为了检测  $e$  个误码, 要求最小码距应满足:

$$d_0 \geq e + 1 \quad (3-1)$$

在一个码组内为了纠正  $t$  个误码, 要求最小码距应满足:

$$d_0 \geq 2t + 1 \quad (3-2)$$

在一个码组内为了纠正  $t$  个误码, 同时能检测  $e$  个误码 ( $e > t$ ), 要求最小码距应满足:

$$d_0 \geq e + t + 1 \quad (3-3)$$

## § 3.2 RS 信道编码介绍

Reed-Solomon (简称 RS) 码是多元 BCH 码的一种。BCH 码是能纠正多个错误的信道编码, 纠错能力强, 构造方便, 编译码简单, 易于实现, 在短码和中等码长的条件下性能接近 Shannon 极限。RS 码是多元 BCH 码, 它能纠正的突发错误比特数是相同复杂度 BCH 码的  $m$  倍, 因此纠正突发错的能力很强。目前还有一些常用的信道编码, 比如卷积码、Turbo 码等, 它们都有各自的优点, 但是 RS 码凭借其极强的纠突发错的能力, 被广泛应用于各



种通信领域, 文献[16][17][18]提出了使用 RS 码作为网络视频传输的信道编码。

伽罗华域 (简称 GF) 又称有限域, 是指按域的构成规则构成的有限个元素的集合, 要求满足加法和乘法两种运算的代数系统。如果域中有  $q$  个元素, 则表示为  $GF(q)$ 。常用到的是  $GF(2)$ , 即二元域。如果将元素个数按幂次扩展则有  $GF(2^m)$  称其为  $GF(2)$  的  $m$  次扩域, BCH 码是基于  $GF(2)$  的, 但 RS 码是基于  $GF(2^m)$  的。 $(n, k)$  RS 码中, 输入信息分成  $km$  比特一组, 每组包括  $k$  个符号, 每个符号由  $m$  比特组成,  $(n, k)$  RS 码相当于  $(nm, km)$  BCH 码。

一个纠  $v$  个符号错误的 RS 码有如下参数:

- 码长  $n = 2^m - 1$  符号 或  $m(2^m - 1)$  比特
- 信息段  $k$  符号 或  $km$  比特
- 监督段  $n - k = 2v$  符号 或  $m(n - k)$  比特
- 最小码距  $d = 2v + 1$  符号 或  $m(2v + 1)$  比特

对于纠  $v$  个错误符号的 RS 码生成多项式如公式 3-4 所示, 其中  $a$  为  $GF(2^m)$  上的本原元。显然,  $g(x)$  的全部根为  $a, a^2, \dots, a^{2v}$ , 它的系数是  $GF(2^m)$  的元素, 由  $g(x)$  生成的码是  $(n, n - 2v)$  循环码, 它包括所有那些能被  $g(x)$  除尽, 系数是  $GF(2^m)$  的元素, 次数不大于  $n - 1$  的多项式。RS 码的编译码是基于码元而不是一组码元而单独的 0 或 1, 这也是 RS 码纠错能力特别强的原因, 并且这种特点使得 RS 码特别适合处理突发成片的错误。

$$g(x) = (x + a)(x + a^2)(x + a^3)(x + a^4) \cdots (x + a^{2v}) \quad (3-4)$$

由于 RS 码是循环码的一种, 因此, 它的编码方法与一般循环码的编码方法完全一致。用信息码多项式  $m_{k-1}x^{k-1} + m_{k-2}x^{k-2} + \dots + m_0$  升  $x^{n-k}$  位后去除生成多项式  $g(x)$ , 所得余式  $r(x)$  为监督多项式, 将监督多项式置于升  $x^{n-k}$  位的信息多项式之后, 就形成 RS 码。因此 RS 码的编码变成用除法求余的过程, 由于纠  $v$  个符号错误的生成多项式为

$$g(x) = \prod_{i=1}^{2v} (x + a^i) = \sum_{i=0}^{2v} g_i x^i$$

$a$  是有限域  $GF(2^m)$  的元素, 设输入信息码为  $m(x)$ , 编码后的码组为  $c(x)$ , 则

$$c(x) = x^{n-k} m(x) + x^{n-k} m(x) \bmod g(x)$$

RS 码的译码比编码要复杂, 分为以下几个步骤:

(1) 计算多个伴随式值  $S_k$

假设  $T(x) = t_0 + t_1 x + \dots + t_{n-1} x^{n-1}$  是发送的码字, 设  $R(x) = r_0 + r_1 x + \dots + r_{n-1} x^{n-1}$  是相应的接收码字, 则由信道引起的错误图样为

$$e(x) = R(x) - T(x) = e_0 + e_1 x + \dots + e_{n-1} x^{n-1}$$

这里  $e_i = r_i - t_i$  是  $GF(2^m)$  的元素。设错误图样  $e(x)$  含有  $v$  个错误 (非零元素) 分别位于  $x^1, x^2, \dots, x^v$ , 则

$$e(x) = e_1 x^1 + e_2 x^2 + \dots + e_v x^v$$

因此为决定  $e(x)$ , 需知道错误位置  $x^i$  及错误值  $e_i$ 。  $2v$  个伴随式元素是通过把  $a^i$  代入接

收多项式  $R(x)$  而得到, 如果  $S_k = 0$ , 认为接收无误; 若  $S_k \neq 0$ , 由  $S_k$  找出错误图样。因为  $R(a^i) = e(a^i) + T(a^i) = e(a^i)$  ( $T(a^i) = 0$ ), 因此

$$\left\{ \begin{array}{l} S_1 = R(a) = e_1 x^1 + \dots + e_v x^v \\ S_2 = R(a^2) = e_1 (x^1)^2 + \dots + e_v (x^v)^2 \\ \dots\dots\dots \\ S_{2^v} = R(a^{2^v}) = e_1 (x^1)^{2^v} + \dots + e_v (x^v)^{2^v} \end{array} \right\} \quad (3-5)$$

(2) 由伴随式值确定差错定位多项式

由公式 3-5 直接求解  $e^i$  和  $x^i$  非常困难, 因为它是一个非线性联立方程组, 一般不直接求解, 而是将其转换为一组线性方程。为求出  $v$  个错误位置  $x^i$ , 可构造以  $v$  个错误位置  $x^i$  为根的一元  $v$  次线性方程

$$\delta(x_i) = x_i^v + \delta_1 x_i^{v-1} + \dots + \delta_v = 0 \quad i=1, 2, \dots, v \quad (3-6)$$

显然, 以  $x_i$  的任何次幂与公式 3-6 相乘后该式仍为零, 现以  $e_i x_i^j$  与其相乘, 可写出

$$e_i x_i^{v+j} + \delta_1 e_i x_i^{v+j-1} + \dots + \delta_v e_i x_i^j = 0 \quad i=1, 2, \dots, v \quad (3-7)$$

将公式 3-7 对  $i=1, 2, \dots, v$  相加, 由公式 3-5 可得

$$S_{v+j} + \delta_1 S_{v+j-1} + \dots + \delta_v S_j = 0 \quad (3-8)$$

由公式 3-8 定义的方程式叫牛顿恒等式, 以纠  $v$  个差错的非二元 RS 码为例, 已计算出  $2v$  个伴随式值  $S_1, S_2, \dots, S_{2^v}$ 。由公式 3-8, 令  $j$  由 1 取至  $v$ , 可构成  $v$  个线性系数的联立方程。

$$\left\{ \begin{array}{l} S_{v+1} + \delta_1 S_v + \dots + \delta_v S_1 = 0 \\ S_{v+2} + \delta_1 S_{v+1} + \dots + \delta_v S_2 = 0 \\ \dots\dots\dots \\ S_{2^v} + \delta_1 S_{2^v-1} + \dots + \delta_v S_v = 0 \end{array} \right\} \quad (3-9)$$

通过解联立方程可求得  $\delta(x)$  的系数  $\delta_1 \dots \delta_v$ 。这种方法运算量虽然比较大, 但是很适合利用计算及编程实现, 并且容易理解, 简单易行。

(3) 求出错误位置值

求得一元  $v$  次线性方程的系数后, 可以解出其根, 即错误位置, 但不宜直接求解, 而是采用直接代入的方法, 对于

$$\delta(x_i) = x_i^t + \delta_1 x_i^{t-1} + \dots + \delta_t = 0 \quad i=1, 2, \dots, t$$

设先考虑发送的高阶次比特, 以便先对  $a^{n-1}$  定位式进行根测试, 当代入  $a^i$  ( $i=n-1, n-2, \dots, 0$ ) 求得  $\delta(a^i)$  为零时, 说明第  $i$  个码元发生错误。这种循环的方法非常适合于用计算机实现。

(4) 由所得的错误位置值计算错误值

把上面计算的错误位置值代入公式 3-5, 以与求  $\delta_1 \dots \delta_v$  相同的方法解联立方程组可得错误值。

(5) 对所求的差错位置及错误值进行纠正

在对应的错误位置，加上相应的错误值，即完成纠错。

### § 3.3 RS 信道编码实现

本文在  $GF(2^8)$  上实现了 RS (255, 205) 信道编码，每个码符号含有  $m = 8$  个比特，对应一个字节，因此码长  $n = 2^m - 1 = 2^8 - 1 = 255$  符号（字节），信息段  $k = 205$  符号（字节），监督段  $n - k = 50$  符号（字节），能纠正 25 个符号（字节）错误，纠正突发错误的能力很强。同时根据 RS 编码的特点，在实际应用时可以根据具体情况改变信息段和监督段的符号长度，使其具有灵活的纠错能力。比如在网络传输中当带宽紧张时，使用过多的监督段会进一步降低网络传输的效率，减少了实际数据带宽，这时就应该适当减少监督段符号数，例如可以实现 RS (255, 235) 纠正 10 个符号错误，使带宽和纠错能达到一个平衡点，这将是下一章研究的问题。

本文信道编码的基本工作过程是：

(1) 对输入的源数据包进行编码，编码前对源数据包进行分割，分成 205 个字节一个信息段编码单元，不足 205 字节的尾部通过填补特定的标志数据补足成 205 字节，然后对每一个 205 字节的编码单元进行编码，编码后得到了对应于该编码单元的监督段数据，将这两部分结合作为已编码数据单元，已编码单元应具有 255 个字节，将同属于一个源数据包的已编码数据单元重新组合成已编码数据包，用于实际传输。

(2) 在接收端收到数据包后，对接收数据包按照 255 字节进行分割，得到待译码数据单元，对待译码数据单元进行译码，如果发生的错误字节数小于纠错能力 25 字节，则可以完成纠错，得到正确的译码数据单元，将译码数据单元尾部的填补标志去除即得到信息段数据，所有信息段数据再组合就还原了源数据包。

整个编码流程工作图如图 3-2 所示。

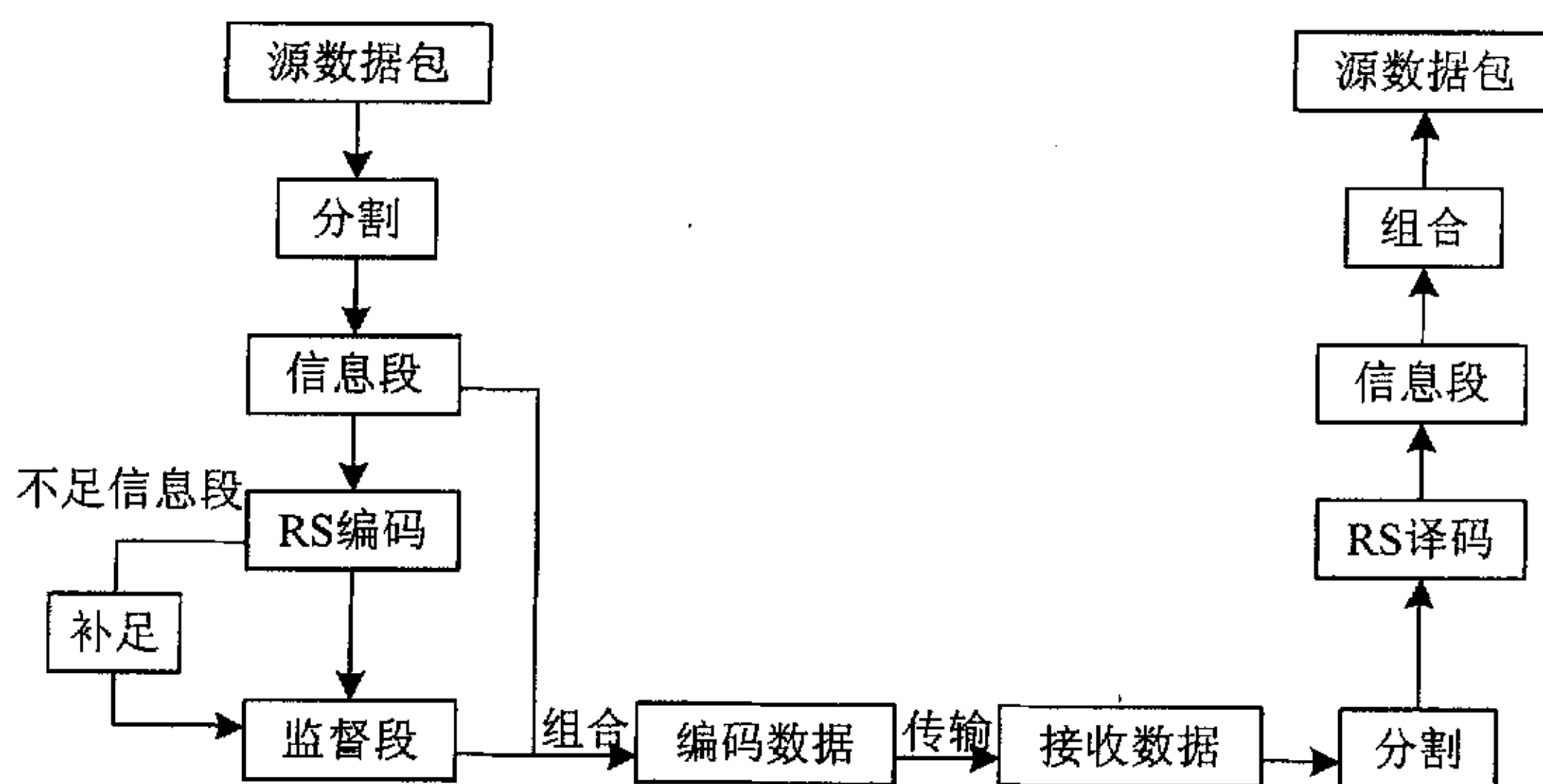


图 3-2 RS 编码流程工作图

RS 译码的软件实现与上一节所述译码步骤类似，只是涉及到具体的软件处理方法不同，分为以下几步：

- (1) 由接收码字计算得到伴随式；
- (2) 用伯利坎普迭代算法 (Berlekamp iterative algorithm) 从伴随式得到差错定位多

项式;

- (3) 钱搜索求差错定位多项式的根, 其倒数为差错位置数;
- (4) 计算错误值;
- (5) 接收多项式减差错多项式, 得到正确的码字。

在本文 RS 编码软件实现中借鉴了文献[17]中提到的通过查表法降低“模乘”复杂度的方法, 提高了软件的编码效率。通过建立二进制字与对应  $GF(2^m)$  域元素的生成多项式指数的映射表和逆映射表来实现。当表的长度不大时 (本文使用  $GF(2^8)$ , 表长 256), 表查找时间几乎可忽略不计, 大大减少了 RS 编码中大量存在的“模乘”运算的运算时间, 运算效率提高明显。

在仿真实验中对视频测试序列进行了编译码速度试验, 与原来没有采用查表算法进行对比。实验采用普通 PC 机, CPU 为奔腾 IV 1.6G, 内存为 256M DDR。实验所得数据如表 3-1 所示, 可以发现使用可以纠正 25 个符号错的 (255, 205) RS 码时, 编码效率可达到 9.7Mbps, 译码效率可达到 14.6Mbps; 而使用可以纠正 10 个符号错的 (255, 235) RS 码时, 编码效率和译码效率分别达到了 52.1Mbps 和 86.5Mbps, 完全可以满足普通视频传输的要求。反观不使用查表法的编译码性能, 平均只有使用查表法速度的 1/5, 性能比较低, 在处理大量数据编译码时会遇到困难。

表 3-1 查表法性能测试对比

纠错数(符号)		使用查表法	不使用查表法
25	编码	9.7Mbps	1.9Mbps
	译码	14.6Mbps	2.9Mbps
10	编码	52.1Mbps	11.4Mbps
	译码	86.5Mbps	18.7Mbps

### § 3.4 小结

基于 FEC 的信道差错控制具有控制时延低、扩展性好、包丢失恢复效率高的优点。本章用软件实现了 Galois 域上 Reed-Solomon 码; 通过查找表实现多项式运算, 有效地降低多项式运算的复杂度, 提高了 RS 信道编码的效率。实验结果表明, 本文实现的 RS 信道编码具有较高的编/译码效率, 能够满足大多数网络应用的输出速率要求。而且, 随着计算机硬件技术的快速发展, 本文实现的软件 RS 信道编码性能可望得到进一步提高。

使用 RS 码显然会带来冗余, 对网络带宽造成新的压力, 在带宽受限时使用 RS 码进行传输数据保护就必须牺牲一定的视频码率, 从而导致视频质量的下降, RS 码监督段越长码率牺牲越多。然而使用 RS 码又能增强视频数据对差错的鲁棒性, 在易损信道中又能对视频质量起到一定的改善作用, 监督段越长保护能力更强, 纠正差错更多。因此, RS 码中究竟采用多少监督段数据能使最终视频质量达到最优就是一个值得探讨的问题, 下一章将对这个问题进行研究。



## 第四章 端对端的视频传输差错控制模型研究

为了在噪声信道上传输视频,一般都要使用信源编码和信道编码。信源信道联合编码在实际应用中是很有用的,现在是一个研究的热点。尽管研究取得了很大进展,但是信源信道联合编码还处于其幼年期。现在比较实际的技术成果是将信源编码和信道编码分开处理,但是对他们的参数进行联合优化。这种优化的一个关键点就是信源和信道编码的码率分配问题。信源信道码率分配的灵活性增加了系统的复杂性,使参数优化变得更困难。

视频传输系统总体的性能取决于多种因素,比如率失真性能和信源的差错恢复编码、信道编码的纠错能力以及信道特性。由于这些系统部件集成在一起组成视频传输系统,单个参数的影响很难确定,整个系统的设计可能会成为一个极其困难的任务。以往的研究经常用仿真来研究整个系统的性能[19]。但是仿真得到的数据有限且只能对基本问题提供有限的帮助。而且在参数范围较大时仿真将变得十分复杂。因此需要使用适当的模型来分析和研究系统参数间的相互作用和平衡点。

通过上两章分别介绍 H.26L 编解码器和 RS 信道编码,本文借鉴 H.263 的视频传输模型,研究了基于 H.26L 的视频传输系统差错控制模型,通过实验证明该模型对实际情况的模拟比较准确,并通过这个模型分析了系统的端对端总体性能,找到了特定环境下系统参数的最佳值。

### § 4.1 视频传输系统

#### 4.1.1 系统概述

如图 4-1 所示,整个视频传输系统包含三个部分:视频编码器、差错控制信道、视频解码器,其中差错控制信道由信道编码和信道模型组成[20]。该模型系统中使用到的参数在表 4-1 中列出。

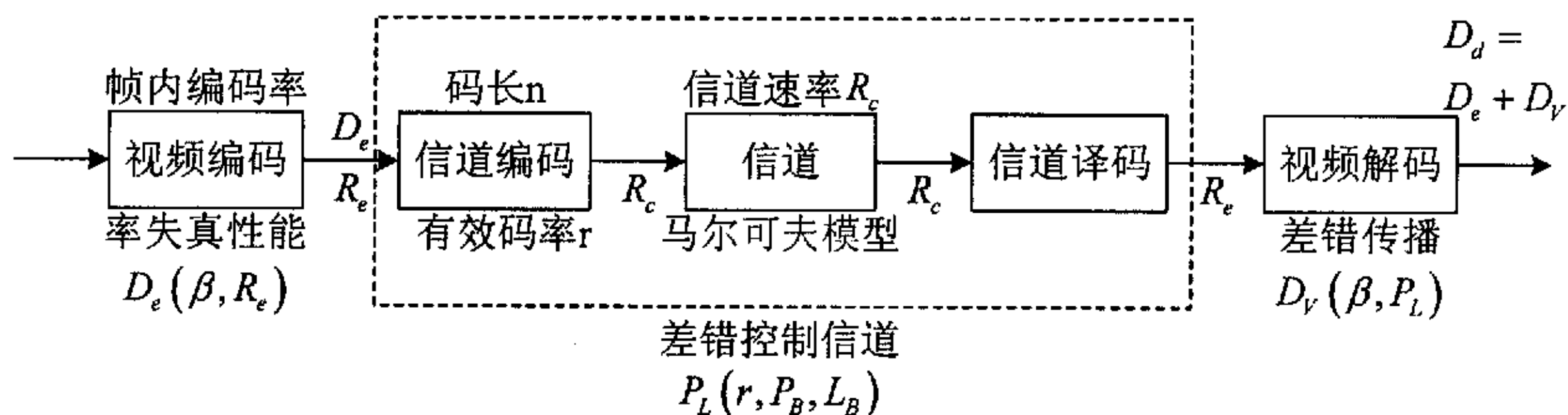


图 4-1 视频传输系统构成图

视频源输入到 H.26L 视频编码器中。H.26L 视频编码器的特性通过率失真性能函数  $D_e(\beta, R_e)$  反映,平均失真  $D_e$  表示为视频码率  $R_e$  和帧内刷新率  $\beta$  的函数。由于帧内刷新率对于差错恢复影响显著,在率失真关系中它是最重要参数。 $\beta$  也是整个系统中的一个重要参数。

经过信源编码,压缩后的视频流通过信道编码以便于传输。具体来讲,就是使用第三章讲到的  $(n, k)$  RS 码,码长为  $n$ , 包含信息段长为  $k$ 。另一个系统优化的重要参数就是

信道编码效率  $r = k/n$ 。通过降低信道编码效率，也就是减少信息段长，更多的信道编码冗余被加入到每个编码码字中，增强了信道编码的纠错能力但同时降低了信道吞吐量。

经过信道编码，RS 编码的码字通过信道进行传输。本文用两状态马尔可夫模型描述信道错误模式，用平均符号错误率  $P_B$  和平均突发错长度  $L_B$  作为信道模型的参数。这两个参数和信道速率  $R_c$  一起描述视频传输信道的特性，用于研究突发错和随机错的影响。根据马尔可夫模型的参数和信道编码效率可以计算信道译码后的残余错误率  $P_L(r, P_B, L_B)$ 。

最后来考虑残余错误对解码视频质量的影响。当信道译码不能完全纠正错误时，错误的译码码字被丢弃，这可能导致严重的图像失真。视频码流的重同步和差错隐藏是两个能减轻残余差错影响的重要手段，这在第二章中已经讨论过了。另一个重要问题是帧间误码扩散，因为使用了帧间预测，某一帧参考图像出错会导致后续图像也跟着出错。因此在系统中加入了帧间误码扩散模型用以描述解码器端的这种失真  $D_V(\beta, P_L)$ ，它有帧内刷新率  $\beta$  和残余错误率  $P_L$  两个参数。

表 4-1 视频传输系统参数列表

自由参数		
$\beta$	帧内刷新率（宏块的百分比）	
$r$	信道编码效率	
$n$	信道编码码长	$n = 255$
图像序列参数		取决于
$R_0$	速率分量	编码器, $\beta$
$D_0$	失真分量	编码器, $\beta$
$\theta$	率失真因素	编码器, $\beta$
$\sigma_{u0}^2$	每个丢失数据包的差错能量	分包方式, 差错隐藏
$\gamma$	环路滤波效率	编码器, 分包方式
信道参数		缺省设定值
$R_c$	信道速率	$R_c = 200\text{kbps}$
$P_B$	符号错误率	
$L_B$	平均突发错长度	$L_B = 8$
其他相关参数		取决于
$t_c$	信道纠错能力	$n, k$
$k$	每个码字中信息段长	$r, n$
$R_e$	视频码率	$R_c, r$
$D_e$	视频编码失真	$\beta, R_e, R_0, D_0, \theta$
$\sigma_u^2$	差错能量	$\sigma_{u0}^2, P_L$
$D_V$	解码端误码扩散造成的失真	$\sigma_{u0}^2, \gamma, \beta$
$P_D$	码字差错密度函数	$n, P_B, L_B$
$P_L$	残余差错率	$P_D, n, r$
$D_d$	解码端总失真	$D_e, D_V$

经过对系统各个部分的简要介绍,可以把各系统联合起来分析,寻找他们之间的相互关系和平衡点,因为这影响总失真  $D_d = D_e + D_v$ 。首先考虑信道编码效率  $r$  的变化,当给定信道速率  $R_c$ ,信道编码效率控制了信源编码和信道编码码流的比率。这对图像质量有两方面影响。首先,减小  $r$  就减小了分配给信源编码的比特率,增加了信源编码部分的失真  $D_e(\beta, R_c)$ 。另一方面,根据差错控制信道错误特性  $P_L(r, P_B, L_B)$ ,如果减小  $r$  就降低了残余错误率  $P_L$ 。最后,  $P_L$  的降低就相应地减小了解码器端失真  $D_v(\beta, P_L)$ 。考虑视频解码端总失真  $D_d = D_e + D_v$ ,整个系统几个部分的相互作用使设置最优参数变得很困难,但从根本上讲,使解码端视频质量达到最优的系统参数应该存在且可以找到。

接下来考虑帧内刷新率  $\beta$  的变化。由于帧内编码宏块不依赖于前一个参考帧,因此增加帧内编码宏块数量可以降低误码扩散的影响,也就降低了  $D_v$ 。但是帧内编码也降低了基于运动预测的编码效率,需要更多比特率,在固定视频码率  $R_c$  时就增加了编码失真  $D_e$ 。无论  $\beta$  的增加是否有利,系统总失真  $D_d = D_e + D_v$  取决于各个部分失真的增/减量的比值。这表明为了获得整个系统的优化参数,每个部分的模型应力求准确,特别是本章将要研究的  $r$  和  $\beta$  的优化。

#### 4.1.2 仿真环境

本文用于证明模型可行性的仿真环境介绍如下。

使用 QCIF 格式的测试视频序列 Foreman 作为视频源,编解码使用 150 帧。对于信源编码使用第二章介绍的 H.26L 编码。每个视频帧运用 Slice 分割,每一行宏块分为一个 Slice,每个 Slice 头部都包含重同步信息。编码器使用固定的视频码率  $R_c$ ,  $R_c$  的码率控制使用如下所述的简单速率控制。每一帧视频图像使用相同的量化步长进行编码,然后根据目标码率动态调整每帧的量化步长。量化步长的调整按下面的步骤进行,首先选择某一帧的视频编码模式,然后对运动预测后的残差进行变换并按照不同的量化步长进行量化。最后,使编码后传输码率和目标码率差值最小的量化步长就被采用。这种速率控制方法能够降低码流缓冲区变化,使传输控制在一个比较固定的码率。在实际应用中,其他一些速率控制算法也被用于进一步降低码流缓冲区的变化,但是本文关注的重点不在这里,因此上述方法也足够使用了。

另一个关于视频编码的控制问题是帧内宏块刷新的方式。以往的文献中提到了一些方式,包括图像活动区域刷新[21][22],多形状模式帧内刷新[23]或者率失真优化编码框架中的帧内编码模式选择[24][25][26]。在通常情况下,如果采用帧间编码模式时,每个宏块都有一个不断增长的编号。当编号达到某一个阈值  $T$  (刷新间隔)时,该宏块就用帧内编码模式,并把编号重置为零。本文中的仿真使用一种类似的方式,但是  $T$  并不是像某些标准建议的一样是固定值(例如 H.263 建议中  $T=132$ ),而是一个可变值,因为它对应的是  $\beta$  的变化。

信道的参数设定如下。对于 Foreman 参考序列使用信道速率  $R_c = 200\text{kbps}$ 。这个参数使帧内刷新率  $\beta$  和信道编码效率  $r$  能够在一个比较宽的区域内变化并且不会带来很大的失真。平均符号错误率  $P_B$  设为集合  $P_B \in \{10, 7.5, 5.0, 2.5, 1.0, 0.5\}[\%]$ ,平均突发错长度设为  $L_B = 8$ 。

RS 信道编码的参数设定如下。使用最常用的  $GF(2^8)$  上的 RS 码,码长 255,每个码符号包含 8 个比特,也就对应一个字节。信息段长根据信道编码效率  $r$  的变化而定。



最后来考虑视频解码端。如果在传输过程中出错较多, RS 译码失败, H.26L 解码器检测到错误的码流语法结构, 不能正常解码, 则丢弃错误数据, 采用第二章所述差错隐藏技术对丢失数据的 Slice 进行差错隐藏处理。

#### 4.1.3 失真计算

为了评估视频传输系统有必要对整个视频序列求平均失真度以给出一个评定优劣的数字。本文使用视频序列所有帧的均方差 (MSE) 来衡量视频失真度。PSNR 是一个编码中常用的计算标准, 本文使用

$$PSNR = 10 \log_{10} (255^2 / MSE)$$

来评价仿真结果。平均 PSNR 一般是先计算每一帧的 PSNR 再求平均值。

由于需要区分编码端和解码端的视频质量, 所以使用  $D_e$  描述整个视频序列编码后的 MSE, 得到

$$PSNR_e = 10 \log_{10} (255^2 / D_e) \quad (4-1)$$

描述编码端 PSNR 值。在解码端用总失真  $D_d$  表示 MSE, 解码端 PSNR 值计算如下

$$PSNR_d = 10 \log_{10} (255^2 / D_d) \quad (4-2)$$

为了保证解码端的失真在一个稳定状态, 只有后 100 个编码帧用于计算  $D_e$  和  $D_d$ 。

如上所述, 总失真  $D_d$  实际是由两部分失真叠加组成, 一部分是由视频编码端压缩带来的  $D_e$ , 另一部分是由信道译码后的残余差错和帧间预测导致的误码扩散带来的  $D_v$ 。根据文献[27]的研究, 假设  $D_e$  和  $D_v$  是不相关的, 就用公式 4-3 计算总失真度 MSE。

$$D_d = D_e + D_v \quad (4-3)$$

## § 4.2 视频编解码分析

### 4.2.1 视频编码分析

在这一节中将借鉴 H.263 视频传输模型[27]来研究基于 H.26L 的速率-失真 (RD) 分析模型, 由于 H.26L 和 H.263 一脉相承, 因此模型的使用也具有相似性, 后面的实验证明也确实如此。这个模型是一个经验模型, 并不是从理论分析得出的。本文关注的是视频编码的输入和输出, 重点在于寻找一种简单实用的模型来描述速率和失真的关系。一方面, 本文要分析的是一个完整的视频传输系统, 每个部分的复杂程度应尽量控制在可以接受的范围。另一方面, 通过实验发现基于简单假设的理论分析得出的模型经常不能准确地和实验结果匹配。例如文献[28]中通过理论分析得出的模型尽管想法很新颖, 但是在用于分析 H.263 的 RD 性能时不能提供足够的准确度。类似的问题在文献[29]中分析 DCT 变换编码的 RD 性能时也遇到了。因此本文使用了一个简单的方程来反映编码失真  $D_e$  和其他相关参数之间的关系。针对仿真环境, 有两个参数对  $D_e$  影响较大, 即视频码率  $R_e$  (分配给视频编码的码率) 和帧内宏块编码率  $\beta$  (增强对差错鲁棒性)。本文采用文献[27]中给出的 H.263 编码视频源失真模型, 该经验模型根据给出的  $R_e$  和  $\beta$  描述整个频序列的平均失真。



$$D_e = \frac{\theta}{R_e - R_0} + D_0 \quad (4-4)$$

其中  $D_e$  是编码视频序列的失真,  $R_e$  是视频编码器输出的码率。剩下的几个 RD 模型的参数  $(\theta, R_0, D_0)$  取决于编码序列和帧内刷新率  $\beta$ 。通过实验发现发现这几个参数和  $\beta$  的关系近似于一个线性关系, 如下所示:

$$\begin{aligned} \theta &= \theta_p + \Delta\theta_{IP}\beta \\ R_0 &= R_{0P} + \Delta R_{0IP}\beta \\ D_0 &= D_{0P} + \Delta D_{0IP}\beta \end{aligned} \quad (4-5)$$

因此模型参数是  $\theta_p, \Delta\theta_{IP}, R_{0P}, \Delta R_{0IP}, D_{0P}, \Delta D_{0IP}$ , 总共 6 个, 根据公式 4-5 只需要两个不同的帧内刷新率  $\beta$  值就足够计算出 RD 曲线。图 4-2 显示了公式 4-4 和 4-5 能够比较准确地和视频编码的 RD 性能近似。

通过实际测量  $\beta = 1\%$  和  $\beta = 55\%$  时不同码率  $R_e$  下的  $D_e$  值, 得出  $\beta = 1\%$  和  $\beta = 55\%$  时的 RD 曲线, 然后根据公式 4-4 拟合模型曲线, 使模型曲线和实验测量曲线在所有测量点处 MSE 值的差的总和最小。因此就得到了分别对应于两条 RD 曲线的两组  $(\theta, R_0, D_0)$ , 再根据公式 4-5 可计算出  $\theta_p, \Delta\theta_{IP}, R_{0P}, \Delta R_{0IP}, D_{0P}, \Delta D_{0IP}$ , 这六个参数就可以通过插值计算得出其他帧内刷新率  $\beta$  下的 RD 曲线。图 4-2 中  $\beta = 3\%, 6\%, 11\%, 22\%, 33\%, 44\%$  时的 RD 曲线就是根据这六个参数通过公式 4-4 和 4-5 得出的。模型拟合曲线和实际测量曲线之间最大的 PSNR 误差只有 0.3dB。由此可见, 采用 H.263 的 RD 模型也是可以模拟 H.26L 的失真状况的, 不同之处只在于模型的参数。

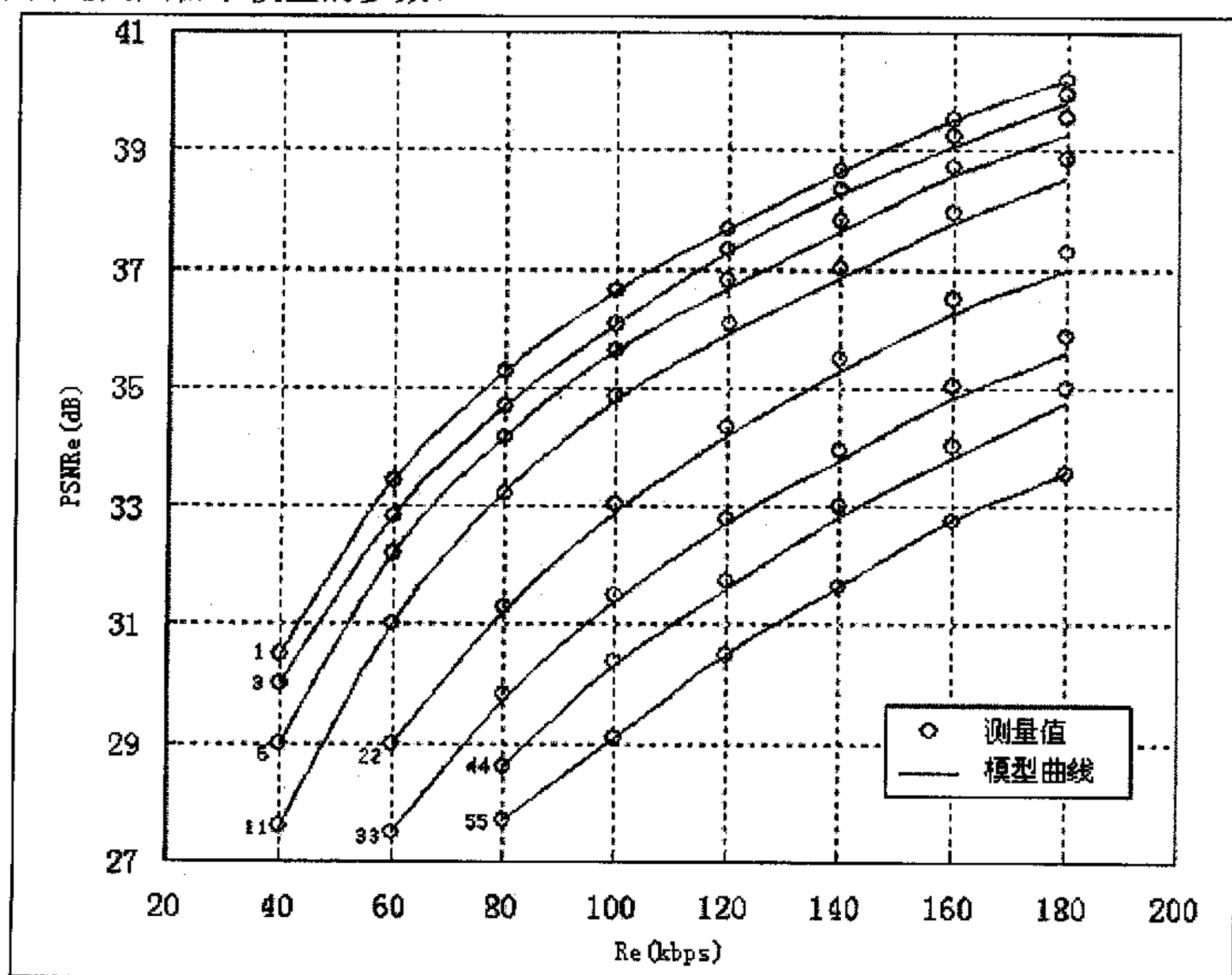


图 4-2 不同  $\beta$  值的 RD 曲线

#### 4.2.2 视频解码分析

视频解码端的总失真主要由两类差错引起。一种是编码端视频压缩引入的  $D_c$ ，另一种是信道译码不能纠正的残余误差带来的失真。对于第一种差错的描述在上一节的公式 4-4 中已经给出，这一节研究第二种差错失真，用变量  $D_v$  表示。

首先定义信道译码后的残余差错的方差为  $\sigma_u^2$ ，显然  $\sigma_u^2$  直接和残余差错率  $P_L$  相关，丢包数量的增加也会增加引入差错的方差  $\sigma_u^2$ 。除此之外，还和其他因素相关，比如重同步和差错隐藏技术的使用。对于一个给定的视频序列，在包长、解码环境给定的情况下，差错的方差可以如下表示为

$$\sigma_u^2 = \sigma_{u0}^2 P_L \quad (4-6)$$

这种线性关系只在较低残余差错率的情况下适用[27]，比如  $P_L < 0.1$ 。由于在高差错率情况下即使使用先进的差错恢复技术也很难保证合适的图像质量，因此在本文实验环境下使用该线性关系也足够了。 $\sigma_{u0}^2$  可以被当作一个常量，独立于其它模型参数。它描述了视频解码对差错率增长的敏感度。相对于一般的视频解码，本文实验环境中 H.26L 解码包含的重同步和差错隐藏技术能够减轻残余差错的影响，因此  $\sigma_{u0}^2$  的值相对比较小。

时域上的误码扩散在基于运动补偿预测的视频编码中是很常见的。虽然运动补偿预测获得了很高的编码效率，但是在传输出现差错时它也会导致帧间的误码扩散。由于这种误码扩散衰减的很慢，因此对视频质量影响较大。但如果不采取措施减少差错影响，随着解码环路的差错积累，即使很小的  $\sigma_u^2$  也会导致图像质量令人难以接受。如果为了优化整个视频传输系统的性能，研究误码扩散的影响就很重要了。

由于视频解码的预测环路中的空间滤波可以去除一些高频系数，帧内编码宏块也能终止一部分误码扩散，因此某一帧图像中的差错随着时间会衰减。设  $t=0$  时刻差错的方差为  $\sigma_u^2$ ，则在  $t$  时刻，传播的差错方差如公式 4-7 所示[27]

$$\sigma_v^2[t] = \sigma_u^2 \frac{1 - \beta t}{1 + \gamma t} \quad (4-7)$$

其中  $0 \leq t < T$ ， $T$  是帧内刷新时间间隔，每隔  $T$  个编码帧，就采用帧内编码方式对每个宏块进行编码。当  $t \geq T$  认为差错的能量已经被帧内编码宏块消除了，得到  $\sigma_v^2[t] = 0$ 。 $T$  和帧内刷新率  $\beta$  的关系由公式 4-8 给出。

$$\beta = \frac{1}{T} \quad (4-8)$$

环路滤波效率  $\gamma$  描述了环路滤波器消除差错的能力。它取决于环路滤波器的强度和差错能量谱线密度。如果不使用空间滤波， $\gamma = 0$ ，差错能量的消减只依靠帧内编码。当使用较多的空间滤波或差错包含比较容易消除的高频空间系数时  $\gamma$  的值会增加。一般情况下  $\gamma$  的值满足  $\gamma \in (0, 1)$ 。

由于解码器是线性的，单独的误码扩散延续  $T$  个后续帧，因此得到解码端平均失真  $D_v$  是  $T$  个差错的叠加。在残余差错率比较低时，单独的差错在时间和空间上一般都是离散的，因此可以假设帧与帧之间的差错信号是不相关的。但是在高差错率情况下，该假设就不准确了。与公式 4-6 适用条件类似，在残余差错率较低情况下给出  $D_v$  的计算公式[27]

$$D_V = \sigma_{u0}^2 P_L \sum_{t=0}^{T-1} \frac{1 - \beta t}{1 + \gamma t} \quad (4-9)$$

根据公式 4-9 计算  $D_V$  需要其它几个模型参数。帧内刷新率  $\beta$  是编码器的控制参数, 事先已知。参数残余差错率  $P_L$  取决于信道特性和信道编码, 将在下一节中介绍如何获得。剩下两个参数  $\gamma$  和  $\sigma_{u0}^2$  根据给定的视频编解码器和视频源估计得到。与上一节中拟合 RD 曲线的方式类似, 通过测量一组数据点使模型曲线匹配实验数据。按理来说, 只需要测量两个点就可以计算出  $\gamma$  和  $\sigma_{u0}^2$ 。使用  $\beta^{(1)} = 1\%$  和  $\beta^{(2)} = 33\%$  作为两个测量点, 然后选择适当信道编码效率和符号差错率使  $D_V$  落在  $[10, 140]$  区间内 (大约相当于 PSNR 在 27dB 和 38dB 之间)。图 4-3 所示为不同符号差错率下  $D_V$  实验测量数据和模型拟合曲线图。由图可见公式 4-9 模型能够比较准确地近似传输差错失真。实心星号是用来拟合模型曲线的测量点。

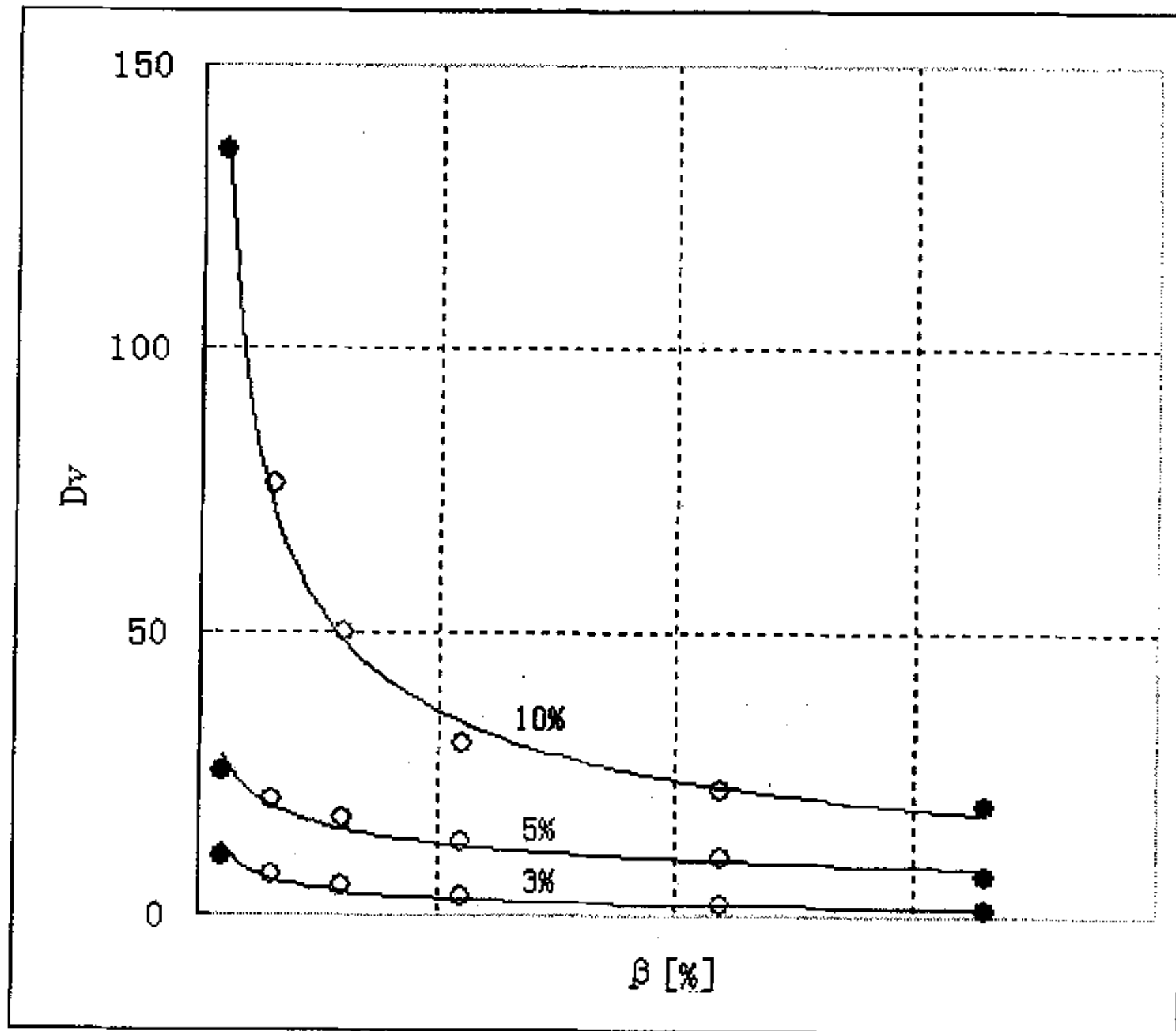


图 4-3  $D_V$  实验数据和模型拟合曲线图

### § 4.3 差错控制信道分析

通过信道编码可以改善传输的可靠性, 因为可以降低残余差错率  $P_L$ , 根据公式 4-9 相应地也就降低了  $D_V$ 。但是, 给定的信道速率为常数, 也就意味着信道速率限制, 信源编码的数据率就必须降低为

$$R_e = rR_c \quad (4-10)$$

其中信道编码效率  $r \in [0, 1]$ , 根据公式 4-4 信源编码失真  $D_e$  就会增加。因此就必须在信源编码失真  $D_e$  和信道带来的失真  $D_V$  之间寻找折衷的平衡点。为了优化总失真  $D_d = D_e + D_V$  就必须弄清降低一定的信道编码效率到底能够带来多大的可靠性。

本文中使用 RS 码作为信道编码。对于  $(n, k)$  RS 码，码长为  $n$  个符号，信息段长  $k$  个符号，监督段长  $n - k$  个符号，于是信道编码效率为

$$r = k/n \quad (4-11)$$

对于码符号长为  $m$  个比特的 RS 码，码长为  $n = 2^m - 1$ ，对源数据的编码需要将数据划分为长度为  $n$  的数据块。本文中使用  $m = 8$ ，因此每个码符号对应一个字节， $n, k$  也是以字节为单位。 $(n, k)$  RS 码能够纠正的最大错误符号数为

$$t_c = \left\lfloor \frac{n - k}{2} \right\rfloor \quad (4-12)$$

超过  $t_c$  个符号错误就不能完全正确地解码，在视频解码时就会由于视频语法结构错误而被检测出来，相应的数据块就解码失败，因此认为差错总是可以被检测出的。

残余差错率  $P_L$  等价于数据块不能被纠错的概率，基于公式 4-12， $P_L$  的计算如下[27]

$$P_L = \sum_{k=t_c+1}^n P_D(n, k) \quad (4-13)$$

$P_D$  是码字差错密度函数， $P_D(n, k)$  表示在  $n$  个传输符号中发生  $k$  个符号错的概率。对于记忆信道计算  $P_D$  比较复杂，本文使用文献[30]中介绍的简单的两状态马尔可夫模型描述信道差错，并利用该文献中的公式计算  $P_D$ ，得出如图 4-4 所示的差错控制信道的性能。

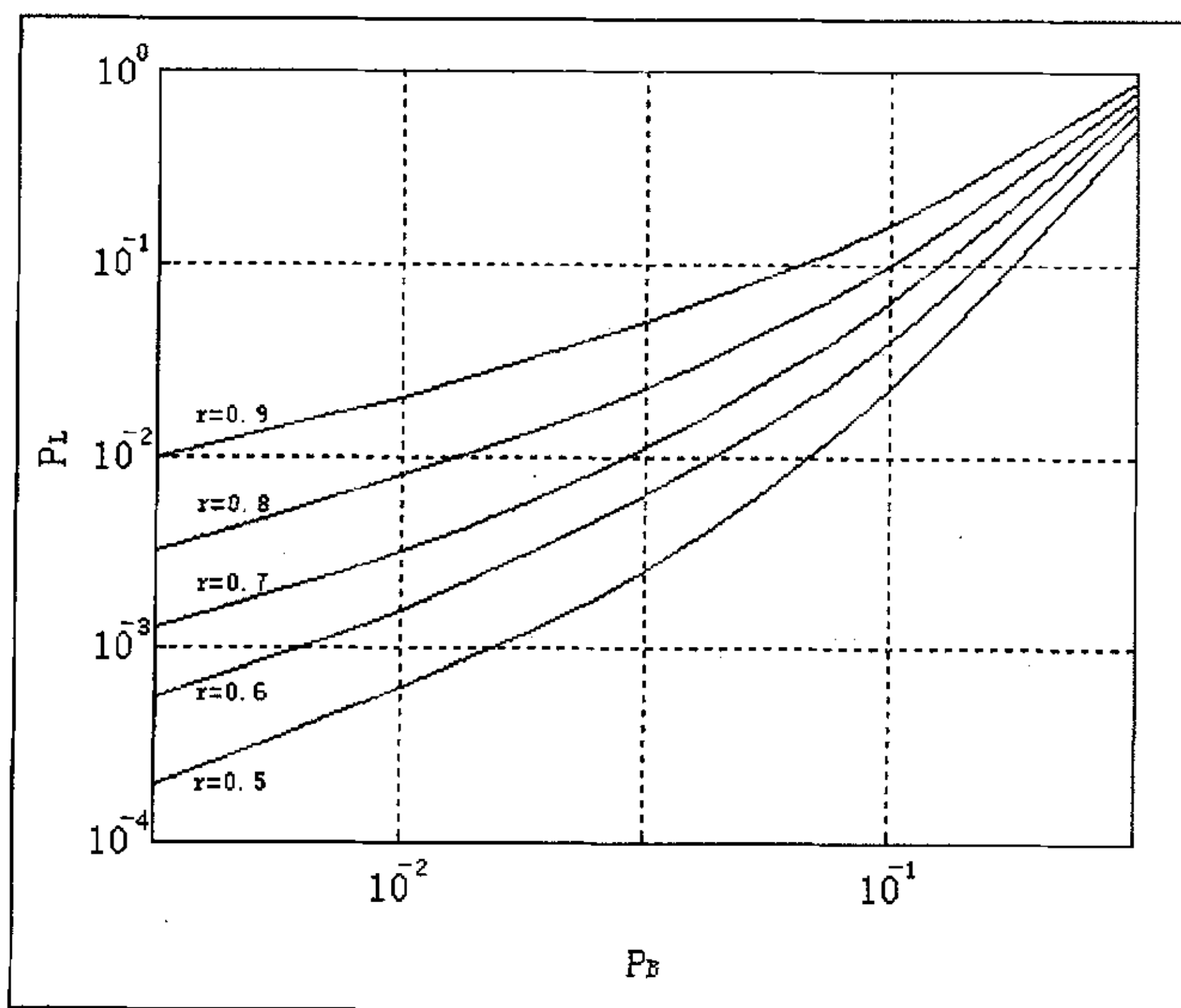


图 4-4 差错控制信道性能

## § 4.4 视频质量优化

根据上两节所述, 已经研究了视频传输系统各个部分的单独模型, 最后需要对整个系统进行仿真, 通过调整帧内刷新率  $\beta$  和信道编码效率  $r$  使系统总失真  $D_d$  最小。首先分析每个参数单独的影响, 然后联合考虑  $\beta$  和  $r$  的优化。

先研究帧内刷新率  $\beta$  对  $D_d$  的影响, 固定信道编码效率  $r$ 。显然, 帧内刷新率  $\beta$  肯定有一个达到性能最优的平衡点。一方面, 增加帧内编码宏块的比例可以减少误码扩散, 根据公式 4-9 可知减小了  $D_v$ ; 另一方面, 根据公式 4-4 和 4-5 可知, 在固定编码目标码率情况下增加帧内刷新率会增加视频编码失真  $D_e$ 。

图 4-5 所示为四种符号差错率下解码端视频质量  $PSNR_d$  随帧内刷新率  $\beta$  变化的曲线。可见模型与实际测量值近似度很高。当无差错时 ( $P_b = 0\%$ ),  $\beta$  的增加导致  $PSNR_d$  不断减小, 完全是受到固定码率的限制。当有差错时,  $\beta$  的增加就能对  $PSNR_d$  起到一定的帮助作用, 在一定范围内可以使  $PSNR_d$  上升, 但是达到最优后, 继续增大的  $\beta$  带来的质量提高就会不足以抵消码率限制带来的质量下降, 导致  $PSNR_d$  的下降。显然在某一个最优的  $\beta$  值时,  $PSNR_d$  能达到最大。当  $\beta \rightarrow 0$  时, 根据公式 4-8 也就意味着  $T \rightarrow \infty$ , 因此由公式 4-9 可知  $D_v \propto \sum_{i=0}^{\infty} \frac{1}{1+i^2} \rightarrow \infty$ ,  $PSNR_d$  下降很快, 因此如果有差错发生时, 必须有一定的帧内刷新率才能保证视频质量。另外, 由图可以看出  $\beta$  在最优值处曲线很平坦, 因此对最优  $\beta$  值的选择不是十分确定, 只要  $\beta$  在一个范围内变化, 都可以使视频质量达到几乎最优。

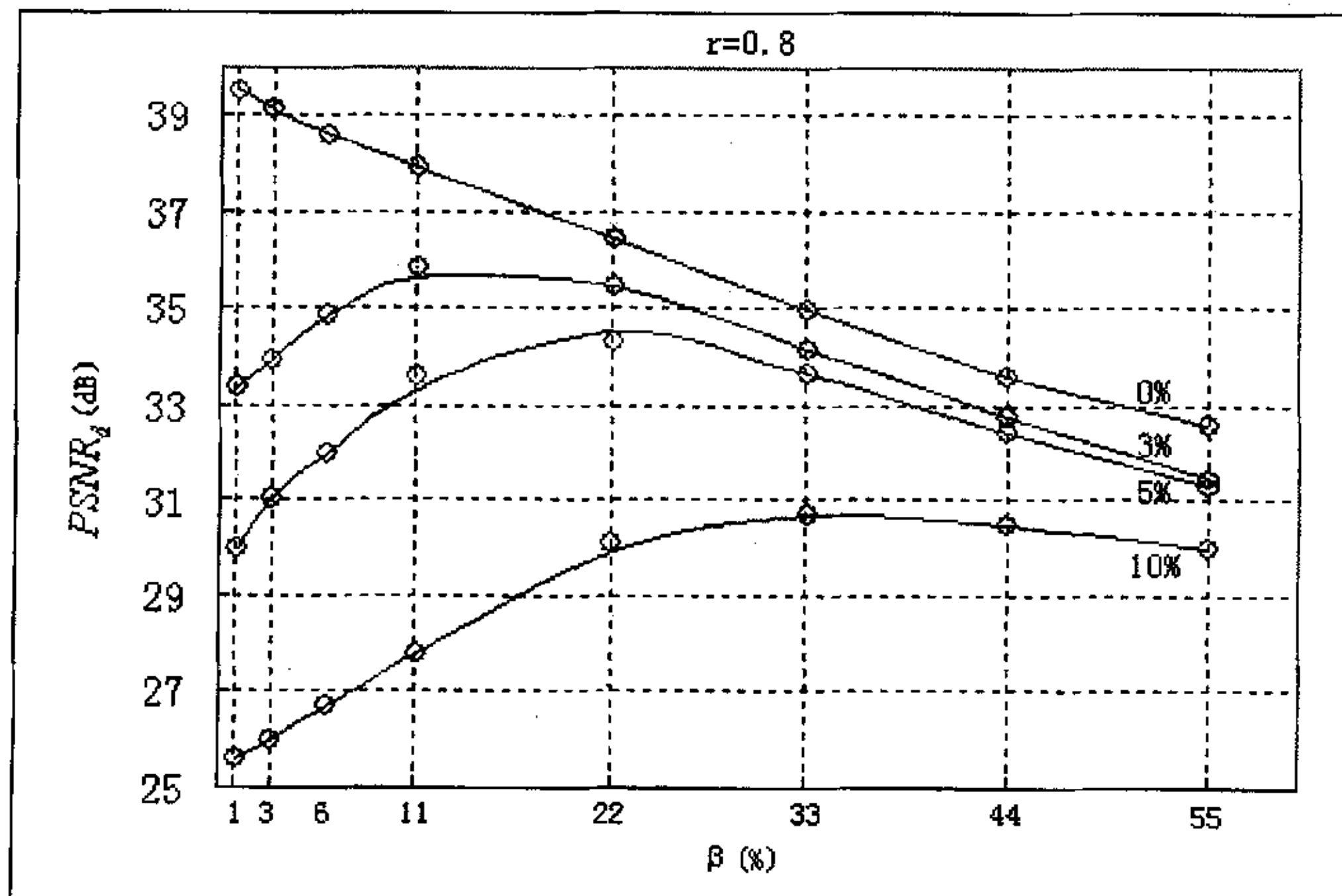
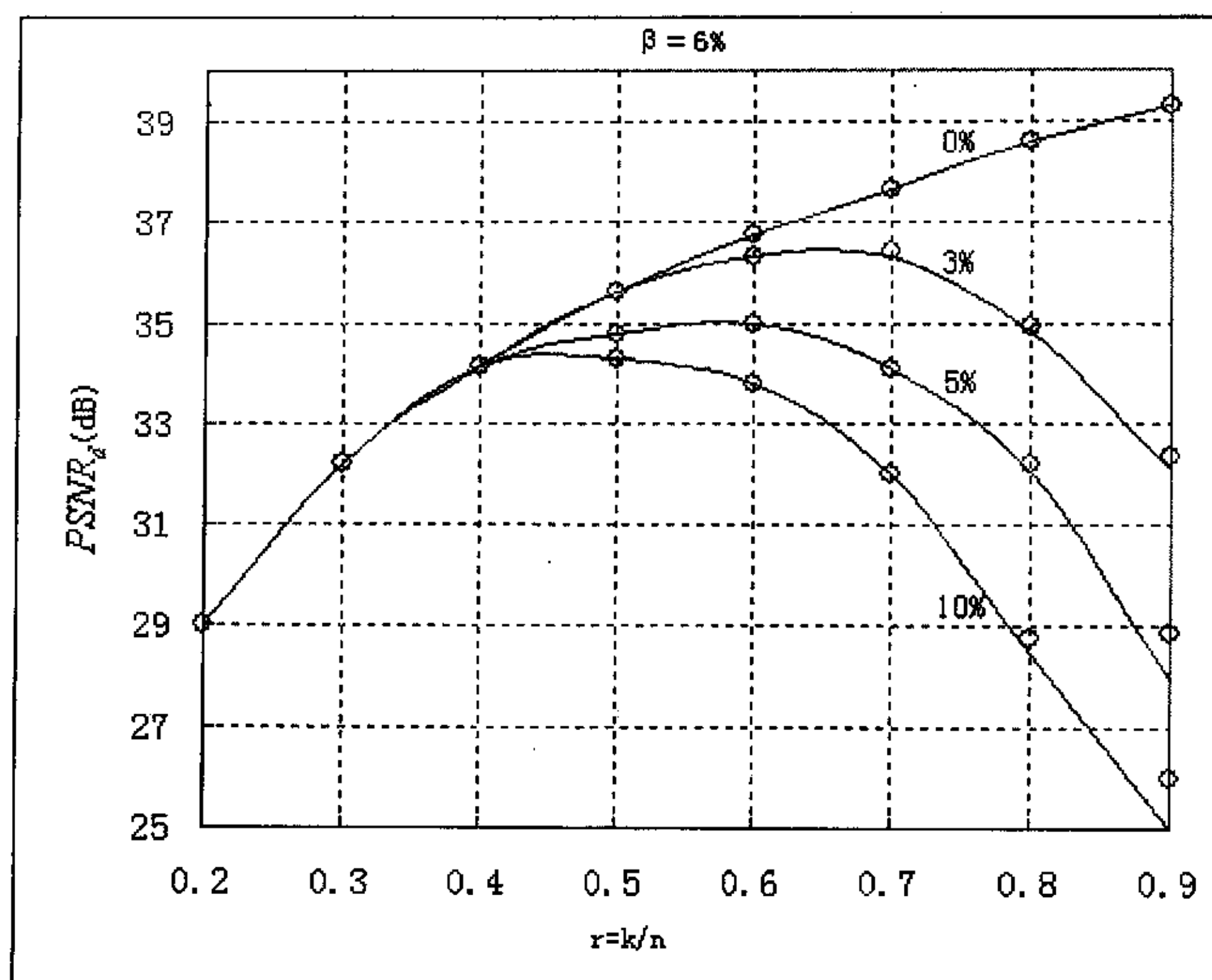


图 4-5  $PSNR_d$  随  $\beta$  变化曲线



图 4-6  $PSNR_d$  随  $r$  变化曲线

接下来研究信道编码效率  $r$  对  $D_d$  的影响, 固定帧内刷新率  $\beta$ 。图 4-6 所示为四种符号差错率下解码端视频质量  $PSNR_d$  随信道编码效率  $r$  变化的曲线。模型计算值和实际测量值也十分接近。只是在差错率较大时模型准确率有所下降。当  $r$  很小时, 信道编码几乎能纠正所有错误, 使得不同符号差错率下的曲线重合。与图 4-5 类似,  $r$  在最优处的曲线也比较平坦, 因此  $r$  在某一个范围内变化, 也能使视频质量达到几乎最优。

最后把  $\beta$  和  $r$  联合起来分析, 如图 4-7 所示, 对每一个  $\beta$  值, 将不同  $r$  值对应的  $PSNR_d$  放在一幅图中, 可见总有一个最大值, 比较每个  $\beta$  对应的最大  $PSNR_d$ , 可得到不同  $\beta$  和  $r$  组合下的最大  $PSNR_d$ , 该  $PSNR_d$  最大值对应的  $\beta$  和  $r$  即为当前情况下 (图 4-7 中符号差错率为 3%) 的最优值。由图可知, 模型得到的最优值曲线和实验测量的最优值很接近。根据同样的方法, 可以得到不同的符号差错率下  $\beta$  和  $r$  的最优组合, 如图 4-8 所示。

至此就找到了基于 H.26L 的端对端视频传输模型在各种差错率情况下的  $\beta$  和  $r$  的最优组合。如果网络状况发生变化, 就可以根据新的网络环境重新测定网络状况参数, 得到适应不同网络环境的模型, 对差错控制进行优化。

## § 4.5 小结

本章借鉴 H.263 的视频传输模型, 研究了基于 H.26L 的视频传输系统差错控制模型, 用于分析通过差错信道传输的压缩视频解码后的恢复质量。该模型不单考虑了视频编码端的率失真问题, 而且考虑了信道上使用信道编码, 以及采用差错恢复和差错隐藏技术对误码扩散带来的性能影响, 模型涵盖面比较广; 但同时能够用有限的参数描述模型, 应用起来比较简单。通过实验证明该模型能够比较准确地对应仿真实验结果。利用该模型可以决

定对于特定的信道使用最优的帧内刷新率  $\beta$  和信道编码效率  $r$ ，使视频传输质量达到最优。

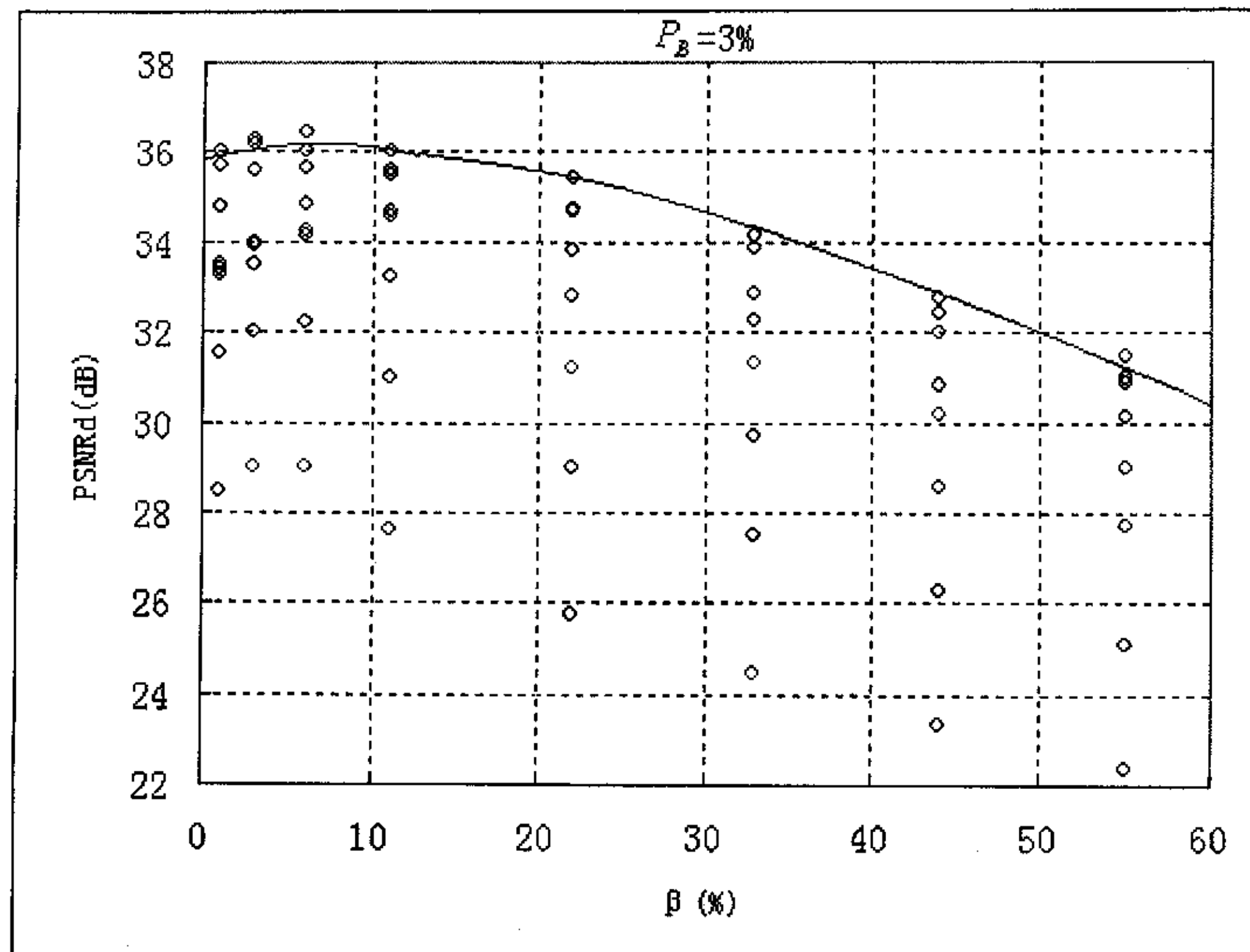


图 4-7  $\beta$  和  $r$  联合优化

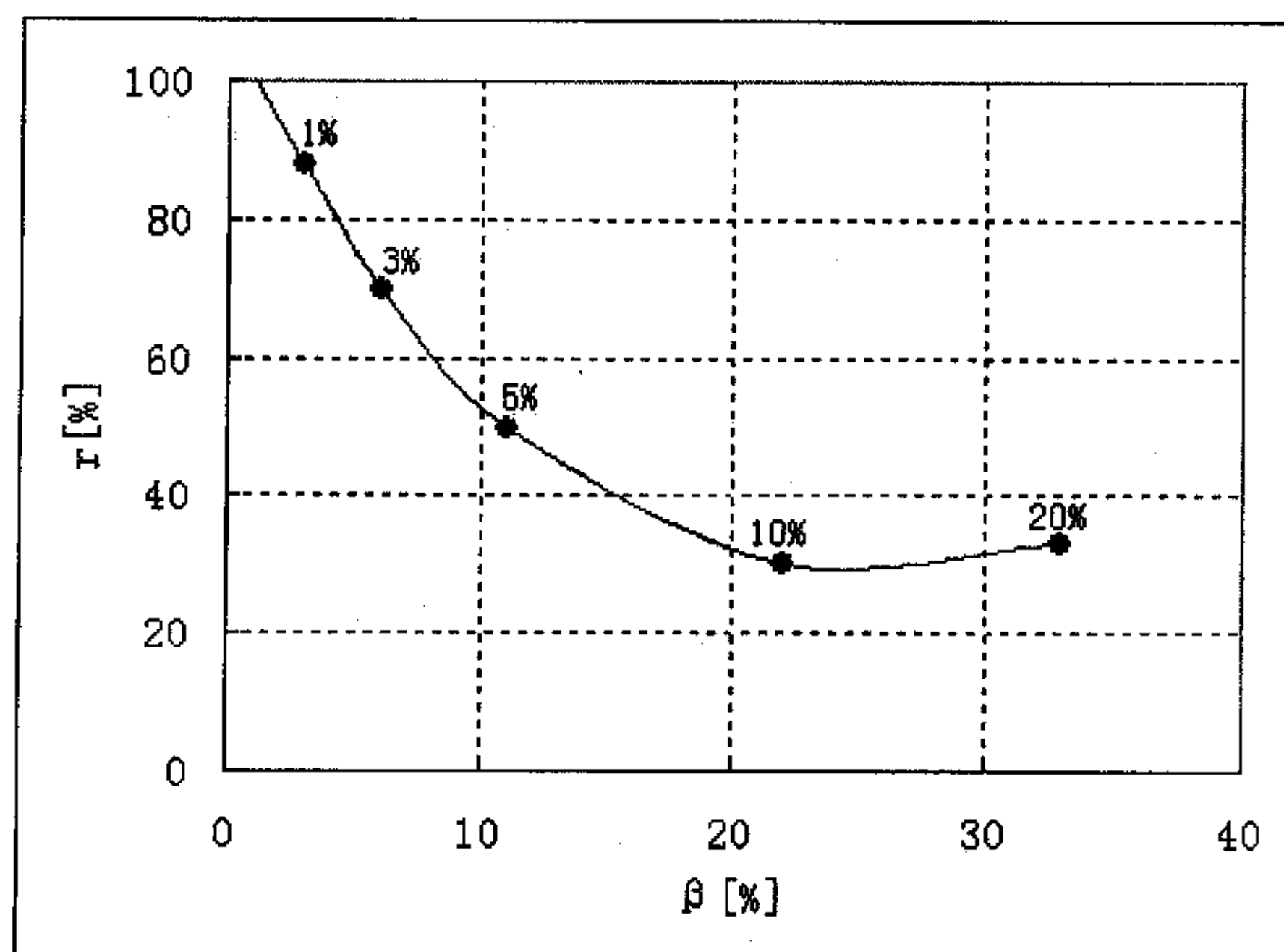


图 4-8 不同差错率下  $\beta$  和  $r$  的最优组合

## 第五章 总结

H.26L 是一个新提出的视频编码标准, 它的高压缩率、网络适应性以及对差错鲁棒性使它很适合于作为易损信道中的视频传输压缩标准。在易损信道中进行视频传输, 差错控制是十分重要的, 如果不加控制, 用户接收到的视频质量将是令人难以接受的。本文围绕在易损信道中的差错控制技术进行了探讨, 着重研究了基于 H.26L 的差错控制技术, 结合信道编码技术, 设计了一个完整的端对端的视频传输系统模型, 包含了从信源、信道到信宿的整个差错控制流程。

本文在第二章介绍了 H.26L 视频编码标准。编码端采用了一些差错恢复技术, 如重同步、数据分区、帧内宏块刷新, 限制了差错范围, 减小了误码扩散的影响。解码端采用了差错隐藏技术, 利用人眼的容错性, 对丢失的数据区域进行填补掩盖, 尽最大可能保证解码的视频质量。针对 H.26L 中原有差错隐藏技术的不足, 提出了一种不同于常理但比较有效的改进方案, 通过实验验证了改进方案的效果还是比较明显的。

本文在第三章介绍了一种实用的信道编码——RS 码。通过编程实现了 RS 码的编译码, 然后应用于视频传输系统的信道差错控制, 对于纠正部分随机错和突发错起到了重要作用, 为视频传输提供了一定的质量保证。而且在编程过程中实现了一种快速查表算法, 使以往困扰软件信道编码实现的效率问题得到了解决, 为实际应用提供了保证。

本文在第四章借鉴 H.263 的视频传输模型, 研究了基于 H.26L 的视频传输系统差错控制模型, 包含了视频源编码端差错恢复技术, 信道上信道编码技术和视频解码端差错隐藏技术。通过该模型分析了帧内刷新率  $\beta$  和信道编码效率  $r$  的最优化问题, 这两个参数是影响模型性能的最主要参数, 它们的变化对视频解码质量都有直接的影响, 而且它们都存在速率和失真的平衡问题。通过仿真实验验证了模型在较低误码率情况下的准确性, 并根据模型寻找了特定网络环境下的这两个参数的最优值, 从而使解码端视频质量在固定码率的情况下达到最好。

关于本文的研究工作, 今后还有一些需要继续研究和改进的地方:

1. 由于受时间和环境的制约, 本文没有合适的网络应用背景, 因此下一步研究将结合基于 H.323 的视频电话应用实例做具体的差错控制研究, 进一步验证本文模型的实用性。
2. 基于 H.26L 的差错控制技术还不是很完善, 今后的研究将在更有效的差错恢复以及基于视频内容(对象、运动)的第二代差错隐藏等方面做一些工作。
3. 对于 RS 码的应用研究来说, 针对分层编码的非平衡保护也是一个研究的热点。
4. 对视频传输系统模型的研究将逐步细化, 建立基于更合理的理论分析的模型, 对于公式 4-3 在编码失真和传输失真之间权重选择将是模型研究的重点。另外, 将模型扩展到支持非平衡保护的分层视频传输系统也是值得研究的内容。

总之, H.26L 标准作为一个新生事物, 它的发展刚刚起步, 无线视频传输的应用需求也为它的发展开辟了空间, 今后的研究还有很多工作要做, 任重道远。



## 参考文献

- [1] 刘轶, H.323 标准及其在 Internet 上的应用, 微型电脑应用, 1999 年第 15 卷第 10 期
- [2] 胡瑞敏, 李德仁, 喻占武, 沈未名, 支持 ITU-T H.324 协议的桌面式可视电话系统, 武汉大学学报, 2002 年 2 月第 27 卷第 1 期
- [3] 谭月东, 彭强, H.263+视频差错跟踪恢复技术研究, 计算机应用研究, 2002 年第 9 期, P45
- [4] 张继东, 徐焱, 徐义明, MPEG-4 中的视频误码复原方法, 数字电视与数字视频, 2002 年第 3 期, P9
- [5] Yuwen He, Feng Wu, Shipeng Li, Yuzhuo Zhong, Shiqiang Yang, "H.26L-BASED FINE GRANULARITY SCALABLE VIDEO CODING", Computer Science and Technology Department, Tsinghua University, Beijing
- [6] Thomas Stockhammer, Dimitrios Kontopodis, Thomas Wiegand, "Rate-Distortion Optimization for JVT/H.26L Video Coding in Packet Loss Environment", Proc. of 2002 Int. Packetvideo Workshop, Pittsburgh, USA, Apr. 2002
- [7] Trista Pei-chun Chen, Tsuhan Chen, "Second-generation error concealment for video transport over error-prone channels", Wirel. Commun. Mob. Comput. 2002, 2:607-624
- [8] R. Zhang, S. L. Regunathan, and K. Rose, "Video Coding with Optimal Inter/Intra-Mode Switching for Packet Loss Resilience," in IEEE JSAC, vol. 18, no. 6, pp. 966-976.
- [9] K. Stuhlmuller, M. Link, B. Girod, "Scalable Internet Video Streaming With Unequal Error Protection", Packet Video Workshop 99, April 1999, New York
- [10] Sorin Dumitrescu, Xiaolin Wu, Zhe Wang, "Globally Optimal Uneven Error-Protected Packetization of Scalable Code Streams", Proceedings of the DCC 2002
- [11] UB Video Inc, "Emerging H.26L Standard: White Paper", February 2002
- [12] Ye-Kui Wang, Miska M. Hannuksela, Viktor Varsa, Ari Hourunranta, Moncef gabbouj, "THE ERROR CONCEALMENT FEATURE IN THE H.26L TEST MODEL", TICSP, Tampere, Finland
- [13] Viktor Varsa, Miska M. Hannuksela, Ye-Kui Wang, "Non-normative error concealment algorithms", Video Coding Experts Group (VCEG) 14th Meeting, VCEG-N62, September, 2001
- [14] H.26L Test Model Software, Version JM 6.1, February 2003
- [15] S. Wenger, "Common conditions for wire-line, low delay IP/UDP/RTP packet loss resilient testing", ITU-T Video Coding Experts Group, VCEG-N79r1, September 2001
- [16] 张平, 王卫东, 陈月华, WCDMA 移动通信系统的信道编码技术的研究, 电子学报, 1999 年 11 月第 11A 期
- [17] 胡飞, 朱耀庭, 朱光喜, 基于 Galois 域 Reed-Solomon 码的数据包层 FEC 编码软件实现, 通信学报, 2002 年 3 月第 23 卷第 3 期
- [18] 袁东风, 姚旗, 王承祥, RS 码与 MPSK 结合的方案在 M 衰弱信道中的性能研究, 山东大学学报, 2000 年 12 月第 35 卷第 4 期

- 
- [19] Q. F. Zhu and L. Kerofsky, "Joint source coding, transport processing, and error concealment for H.323-based packet video", in Proc. Symp. on Visual Comm. and Image Proc., San Jose, CA, Jan.1999, SPIE , vol.3635, pp.52-62
  - [20] S. B. Wicker, Error Control Systems, Prentice-Hall, Englewood Cliffs, NJ, 1995
  - [21] J. Y. Liao and J. D. Villasenor, "Adaptive intra update for video coding over noisy channels", in Int. Conf. on Image Processing, Lausanne, Switzerland, Sept. 1996, vol. 3, pp.763-766
  - [22] P. Haskell and D. Messerschmitt, "Resynchronization of motion compensated video affected by ATM cell loss", in Int. Conf. on Accoust., Speech and Sig. Proc. (ICASSP), San Francisco, CA, Mar.1992, vol. 3, pp.545-548
  - [23] Q. F. Zhu and L. Kerofsky, "Joint source coding, transport processing, and error concealment for H.323-based packet video", in Proc. Symp. on Visual Comm. and Image Proc., San Jose, CA, Jan. 1999, SPIE, vol.3652, pp.52-62
  - [24] G. Cote and F. Kossentini, "Optimal intra coding of blocks for robust video communication over the internet", Signal Proc. Image Comm., vol.15, no.1-2, pp.25-34, Sept.1999
  - [25] T. Wiegand, N. Farber, K. Stuhlmuller, and B. Girod, "Error-resilient video transmission using long-term memory motion-compensation prediction", IEEE Journal on Sel. Areas in Com., vol. 18, no.6, pp.1050-1062, June2000, Special Issue on Error Resilient Image and Video Transmission.
  - [26] S. Wenger and G. Cote, "Using RFC2429 and H.263+ at low to medium bit-rates for low-latency application", in Packet Video Workshop, New York, NY, Apr.1999
  - [27] Klaus Stuhlmuller, Niko Farber, Michael Link, Bend Girod, "Analysis of Video Transmission over Lossy Channels", IEEE Journal on Selected Areas in Communications, vol. 18, no. 6, June 2000.
  - [28] B. Girod, "The efficiency of motion-compensating prediction for hybrid coding of video sequences", IEEE Journal on Sel. Areas in Com., vol.5, no.7, pp.1140-1154, Aug.1987.
  - [29] H. M. Hang and J. J. Chen, "Source model for transform video coder and its application", IEEE Trans. on Circ. and Sys. f. Video Tech., vol.7, no.2, pp.287-311, Apr.1997
  - [30] B. Girod, K. Stuhlmuller, M. Link and U. Horn, "Packet Loss Resilient Internet Video Streaming", SPIE VCIP99, Jan.1999, San Jose, CA.

## 发表论文、获奖、获专利情况

《一种新的 H.26L 差错隐藏改进方案》，入选为国防科技大学第二届研究生活动节会议论文，本人为第一作者。

## 致谢

首先，我要衷心感谢我的指导老师——王晖副教授！他那严谨的治学态度、可敬的学者风范和高昂的工作热情，深深地印在了我的心里。他的言传身教和为人师表的高尚品德极大地感染了我。本文从选题到研究计划的制定，从研究伊始到论文截稿，都无不倾注了王老师的心血。在与王老师的交流和讨论中，他经常能提出一些建设性的意见和建议，他的一言一语往往能点破我心中的谜团，让我的思路豁然开朗，论文研究得以顺利进行下去。此外，王老师在关心我的学习和研究工作的同时，还关注着我的生活，这让我很受感动。在此，向王老师所给与的所有热切关心和巨大帮助表示衷心感谢！

感谢实验室的黄英君、姜志宏、张军老师在我攻读硕士学位期间给与的指导和帮助。感谢张鑫、何伟、陶钧、陈波、郭亮、孙晓等同学对我的无私帮助。在日常的学习、交流、讨论中，他们给了我一些很有用的建议，让我的研究思路更加清晰。而且在具体研究时，他们还帮助我解决一些程序上的问题，指导了我的动手能力。还要感谢我的室友，他在日常生活中给我的帮助让我少了很多麻烦，节省了大量时间用于课题研究。

感谢学员队领导对我的关心，在日常工作生活中对我们毕业课题的支持。感谢我的父母，他们一直关注着我的学业和生活，谢谢他们对我的关心和爱护！我还要感谢我的女友，她对我学习工作的大力支持和理解以及无微不至的关心使我能保持一份快乐的心情进行研究学习。