中国航天

HEVC 低复杂度算法与码率控制算法研究

Research on HEVC low complexity algorithm and rate control algorithm

（申请中国空间技术研究院工学硕士学位论文）

培养单位：中国空间技术研究院西安分院

学　　科：　　信息与通信工程

研究方向：　　空间通信技术

研 究 生：　　黄浩

导　　师：　　马伊民（研究员）

2023 年 6 月 05 日

摘 要

随着我国航天科技的发展，与视频相关的卫星业务在应用领域不断增加的同时，对视频的分辨率、帧率等方面也提出了更高的要求。而传输这些高质量的视频需要占用大量带宽资源，但星上带宽资源有限，所以星上视频需要一种压缩比更高的视频编码标准。相比于 H.264/AVC，H.265/HEVC 在相同的重建视频质量下可以降低大约一半的码率。较高的压缩比依赖于许多新的编码工具的引入，但同时也带来了巨大的计算量，限制了 HEVC 的实时应用。此外，在实际的应用场景中，用于传输视频的码率是上下波动的，选择合适的编码参数使压缩码流能够在目标码率下顺利传输且失真尽可能小是码率控制算法的目标。因此，性能更好的码率控制算法可以提高 HEVC 的实用性。

针对以上两个问题，本文基于 HEVC 编码标准，对 HEVC 中复杂度较高的编码单元划分流程和码率控制流程进行了详细研究。在此基础上提出了一种帧间预测编码单元快速划分算法，并对 HEVC 中自带的码率控制算法进行了改进。本文的主要工作如下：

（1）设计了一种帧间预测编码单元快速划分算法。

通过实验证明了相邻帧同位编码树单元（Coding Tree Unit, CTU）的二值绝对误差和(Binary Sum of Absolute Difference, BSAD)与编码单元深度分布之间的相关性，同时通过实验证明了当前 CTU 的编码单元深度分布与时空相邻的 CTU 的编码单元深度分布之间的相关性，然后通过联合使用这两种关系对当前 CTU 划分过程中的遍历深度范围进行修剪。实验结果表明，本文提出的算法可以平均提高 29.40%的编码速度，而码率仅仅增加 1.30%。

（2）设计了一种基于视觉显著性检测的 LCU 层码率控制算法。

设计了一种新的最大编码单元（Largest Coding Unit, LCU）层目标比特分配权重计算方法，该权重综合考虑了当前 LCU 的视觉显著性以及参考 LCU 残差信息的离散余弦变换的变换系数绝对值之和。同时利用当前 LCU 编码完成后真正需要的比特数、失真以及不同帧之间不同的率失真特性提出了一种新的模型参数更新方法。实验结果表明，相比于 HM-10.1 中自带的码率控制算法，本文所提算法可以平均提高 2.89%的率失真性能，码率控制误差由 1.083%降低至

0.709%。此外，重建视频中视觉显著性较高的区域也给人更好的观看体验。

关键词：HEVC　　帧间预测　　快速划分　　码率控制

Abstract

With the development of aerospace technology in China, the number of satellite services related to video is constantly increasing in application fields. At the same time, higher requirements are also put forward for video resolution, frame rate, and other aspects. The transmission of these high-quality videos requires a large amount of bandwidth resources, but the bandwidth resources on satellite are limited, so on satellite videos require a video encoding standard with a higher compression ratio. Compared to H.264/AVC, H.265/HEVC can reduce the bit rate by about half with the same reconstructed video quality. The high compression ratio depends on the introduction of many new coding tools, but also brings a huge amount of computation, which limits the real-time application of HEVC. In addition, in the actual application scenario, the bit rate used for video transmission fluctuates up and down. The goal of the rate-control algorithm is to select the appropriate encoding parameters so that the compressed stream can be transmitted smoothly under the target bandwidth and the distortion is as small as possible. Therefore, a better rate control algorithm can improve the practicability of HEVC.

In view of the above two problems, based on the HEVC coding standard, this thesis makes a detailed study of the coding unit division process with high complexity and rate control process in HEVC. On this basis, a fast division algorithm of inter prediction coding unit is proposed, and the rate control algorithm in HEVC is improved. The main work of this thesis is as follows:

(1) A fast division algorithm of inter prediction coding unit is proposed.

The relationship between the Binary Sum of Absolute Difference (BSAD) of co-located Coding Tree Unit (CTU) of adjacent frames and the depth of coding unit division is proved by experiments. At the same time, the relationship between the coding unit division depth of the current CTU and the coding unit division depth of the spatio-temporal adjacent CTU is proved by experiments. Then, the traversal depth range in the current CTU partition

process is trimmed by combining these two relationships. experimental results show that the proposed algorithm can increase the coding speed by 29.40% on average, while the bit rate is only 1.3% higher.

(2) A LCU layer code rate control algorithm based on visual saliance detection is designed.

A new bit allocation weight for Largest Coding Unit (LCU) layer was designed, which considered both the visual saliency of the current LCU and the sum of the absolute values of the Discrete Cosine Transform (DCT) coefficients of the residual information of the reference LCU. At the same time, a new model parameter updating method is proposed based on the real required bit number, distortion and different rate-distortion characteristics between different frames after the current LCU encoding is completed. The experimental results show that compared with the rate control algorithm in HM-10.1, the proposed algorithm can improve the rate distortion performance by 2.89% on average, and the rate control error is reduced from 1.083% to 0.709%. In addition, the region with higher visual significance in the reconstructed video also gives people a better viewing experience.

Keywords: HEVC inter prediction fast division rate control

目　录

主要符号对照表

AMVP	Advanced Motion Vector Prediction（高级运动矢量预测）
AMP	Asymmetric Motion Partition（非对称运动划分）
AVC	Advanced Video Coding（高级视频编码）
AVS	Audio Video coding Standard（音视频编码标准）
BDBR	Bjontegaard Delta Bit Rate
BSAD	Binary Sum of Absolute Difference（二值绝对误差和）
CB	Coding Block（编码块）
CTB	Coding Tree Block（编码树块）
CTU	Coding Tree Unit（编码树单元）
CU	Coding Unit（编码单元）
DC	Direct Current（直流）
DCT	Discrete Cosine Transform（离散余弦变换）
GOP	Group of Picture（图像组）
HEVC	High Efficiency Video Coding（高效视频编码）
HR	Hit Rate（击中率）
HVS	Human Visual System（人类视觉系统）
ISO	International Organization for Standards（国际标准化组织）
ITU	International Telecommunication Union（国际电信联盟）
JCTVC	Joint Collaborative Team on Video Coding（视频编码联合专家组）
LCU	Largest Coding Unit（最大编码单元）
MPM	Most Probable Mode（最可能模式）
MSE	Mean Square Error（均方误差）
MV	Motion Vector（运动矢量）
MVD	Motion Vector Difference（运动矢量差值）
PCM	Pulse Code Modulation（脉冲调制编码）
PSNR	Peak Signal to Noise Ratio（峰值信噪比）
PU	Predicting Unit（预测单元）

QP	Quantization Parameter（量化参数）
RDO	Rate Distortion Optimization（率失真优化）
RMD	Rough Mode Decision（粗略模式选择）
ROI	Region of Interest（感兴趣区域）
SAD	Sum of Absolute Difference（绝对误差和）
SSE	Sum of Square Error（平方误差和）
TU	Transform Unit（变换单元）

第 1 章　绪论

1.1　课题研究背景与意义

随着我国航天科技的发展，与视频相关的卫星业务的应用领域已经非常广泛，如电视广播、赛事直播、军事侦察、卫星遥感图像、空间站建造、空间科学实验等。此外，视频也与人们的生活息息相关。在生活娱乐方面，短视频平台、视频播放器、直播平台已经成为当代人们工作之余的主要放松工具。在工作方面，由于 3 年疫情的特殊需求，推动了线上面试、线上会议、线上授课等实时视频通信业务的发展，且疫情过后由于相关业务的便利性仍被广泛使用。此外，智能交通、安防监控以及工业制造领域，视频作为一种强有力的工具也已经被使用了很久。

随着多媒体技术的发展以及显示设备的升级，在视频应用邻域不断增加的同时，视频的质量也在不断的提高。从标清到高清、从高清到全高清，以及现在耳熟能详的超高清 4K 和超高清 8K，高分辨率的视频能带给人们极致的视觉震撼和细节显示。在多个视频应用领域，超高清视频正在快速的占有市场，如超高清影院、超高清电视直播、超高清远程医疗等。如 2022 年，利用中星 9B 卫星，我国成功实现北京冬奥会 4K 实时转播和 8K 超高清试播。因此，视频传输系统中与超高清视频处理、传输相关的硬件、软件、标准等成为近年来研究的热点。

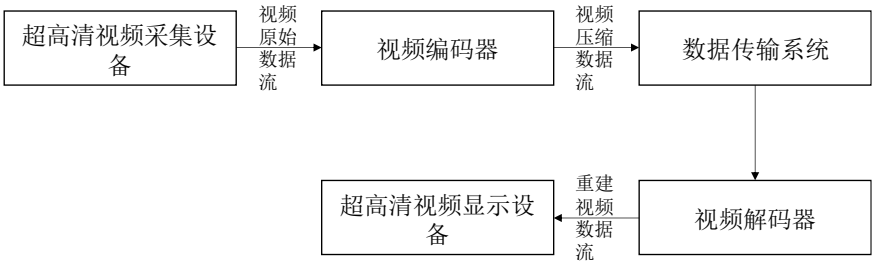


图 1-1　超高清视频传输系统

超高清视频传输系统大致可以分为 5 个模块，如图 1-1 所示。目前，该视频传输系统生态链上的超高清视频采集设备和超高清视频显示设备

早已不再是问题。但是相比于高清视频，在相同帧率以及量化深度的情况下，4K 视频数据量为高清的 9 倍，而 8K 视频数据量为高清的 36 倍。陡然增加若干倍的带宽需求对通信网络来说是一个极大的挑战，成为制约超高清视频传输系统的软肋之一。因此，一种压缩比更高的且能保证良好的重建视频效果的编码标准需要被应用到超高清视频的编码中。

目前市场上高级视频编码标准（Advanced Video Coding, AVC[1]）仍然是视频压缩格式的主流，但在超高清视频数据量陡然增加的面前压缩率开始显得不足。原因是 AVC 在最初设计时，并没有考虑针对超高清视频的压缩，因此导致很多超高清视频特有的性质没有被有效的利用。为了高清和超高清视频压缩，视频编码联合专家小组在 2013 年提出了高效视频编码标准（High Efficiency Video Coding, HEVC），该视频编码标准相较于 AVC 在相同的重建视频质量下，可以节省大约 50%的码率。无论是对于超高清视频的存储还是对于超高清视频的传输，HEVC 都更加的友好[2]。

HEVC 压缩性能的提升依赖于许多新的编码工具的使用，如灵活的四叉树划分结构、更多的预测角度等[3]，这些新的编码工具在带来较高压缩比的同时也带来了巨大的计算量，同一视频文件使用 HEVC 进行压缩的计算复杂度是 AVC 的若干倍[4]，在一定程度上限制了 HEVC 在计算能力有限的设备上的实时应用。因此，在保证压缩效率基本不变的前提下降低 HEVC 的计算复杂度，提高编码速度可以有效地推动 HEVC 的普及应用。

此外，在实际应用场景中，用于传输视频的码率并不是不变的，甚至随着时间推移不断上下波动。如何选择编码参数使压缩码流能够在目标码率下顺利传输且视频失真尽可能较小是码率控制算法的设计目标。因此，性能更加优越的码率控制算法可以极大提高 HEVC 的实用性。

针对以上两个问题，本文基于 HEVC 编码标准，着力开展了低复杂度算法与码率控制算法的研究。

1.2　国内外研究现状

1.2.1　低复杂度算法研究现状

AVC 和 HEVC 都使用了基于块的混合编码框架，相比于 AVC，HEVC 的变化可以分为两个方面。一是添加了新的技术，比如环路滤波中的样点自适应补偿、

用于运动矢量（Motion Vector, MV）预测的 Merge 技术和高级运动矢量预测（Advanced Motion Vector Prediction, AMVP）技术、特殊的脉冲调制编码（Pulse Code Modulation, PCM）模式等等，二是对 AVC 中原有技术进行扩展，比如更加灵活的块划分方式、帧内预测中更多的预测角度、运动估计中的 TZSearch 搜索算法、用于生成亚像素的 7/8 抽头插值滤波器等。两方面的变化虽然都提升了 HEVC 的计算量，但对原有技术的扩展是 HEVC 计算量大幅提升的主要原因。如为了提升预测的精准度，帧内预测模式中将角度预测模式由 9 种提升至 33 种；帧间预测模式中，使用更多的相邻像素来生成亚像素；为了更好的利用高清和超高清视频的特性，将编码单元的尺寸扩展至 64×64，与之对应的是预测单元和变换单元尺寸的提升和多样化。因此，多数低复杂度算法的处理对象为这些耗时较高的技术扩展模块并且这些算法具有通用的处理思路。即在基本不降低 HEVC 压缩率的前提下，计算某些特征值并判断是否满足预设的标准，若满足，则终止继续遍历或者缩小需要遍历的范围，以此来提高编码速度。

针对 HEVC 帧内预测的 35 种预测模式，文献[5]~[11]提出了相应的简化方法。文献[5]中，Hosseini 等人联合利用了当前待处理图像的纹理信息以及图像预测模式的统计信息，对当前待处理图像的预测模式进行分析和筛选。在基本不降低压缩率的同时减少了计算量。文献[6]中，石敏等人利用图像纹理信息、预测单元的尺寸来减少进入粗略模式选择算法的预测模式数量，从而降低计算复杂度。文献[7]提取了编码单元的纹理特征，并利用机器学习中的随机森林设计了一种帧内预测模式快速决定算法。文献[8]中，进入粗略模式选择和最终预测模式选择的预测模式数量均被减少。Zhao 等人在文献[9]中不仅利用了率失真值来提前终止编码单元的划分，而且利用相邻编码单元的最佳预测模式来减少需要遍历的预测模式数量。文献[10]和[11]均利用了图像的纹理信息来加速预测模式的选择过程。

针对 HEVC 中复杂度较高的编码单元划分模块，文献[12]~[22]利用待划分 CTU 深度与时域相邻、空域相邻的已编码 CTU 深度划分之间的相关性来修剪需要遍历的深度，达到降低计算复杂度的效果。这些文献中既包括帧间预测也包括帧内预测。如文献[13]中首先通过实验证明了当前 CTU 划分深度与时空相邻 CTU 划分深度之间的相关性，然后利用前帧同位 CTU、同帧的左侧、左上、上方的 CTU 的划分深度来决定当前 CTU 的遍历范围。文献[17]在利用时空相关性修剪遍历范围后，为了进一步提高编码速度，又使用去块滤波器的边界检查和

CTU 的绝对误差和值（Sum of Absolute Difference, SAD）按照预设的规则再次修剪遍历深度，大大提高了编码速度。文献[18]和文献[19]将编码单元深度空间相关性与率失真代价相结合来确定是否需要提前结束遍历，且为了避免误差积累，文献[19]每隔固定帧数使用 HEVC 标准划分方式进行刷新。

除了利用时空相关性，许多其他的参数也可以做为编码单元（Coding Unit, CU）快速划分的依据。如文献[23]对不同深度编码单元的率失真代价进行统计，且利用贝叶斯估计方法来确定是否提前结束遍历。文献[24]首先对标准序列中不同深度的编码单元的率失真代价进行统计，基于此确定编码单元不再继续划分的阈值，达到节约编码时间的效果，但只对特定序列有较好的结果。文献[25]~[27]对编码单元的预测残差进行处理，将处理结果做为判断是否结束遍历的依据。文献[28]~[32]利用编码单元的纹理信息或者内容特征来加速 CU 的划分。文献[28]利用 Sobel 算子分析编码单元的纹理，依据结果跳过一些概率较小的深度。文献[29]中，纹理比较复杂的区域的深度遍历范围不包括较小深度，而较平坦区域的深度遍历范围不包括较大深度。文献[30]将全局和局部的边缘方向作为编码单元快速划分的依据。文献[31]从 CU 的像素梯度、块均值和块方差三个方面统计分析了视频内容的特征，然后将这些特征和时空相邻 CU 的预测模式相结合，实现快速 CU 深度级别决策和快速预测模式决策。以上各种快速算法均需要研究人员进行实验来总结使用的参数与 CU 划分之间的相关性，然而将已使用 HM 编码过的视频数据作为样本数据，利用深度学习算法来寻找参数与 CU 划分之间的内在规律也是一个非常热门的研究方向，如文献[33]~[37]。

编码单元划分是由多个模块组成的，通过简化某些复杂度较高的模块也可以达到提高编码速度的效果，如前文提到的帧内预测编码单元划分过程中的从 35 种预测模式中选择最佳预测模式。在帧间预测中，预测单元最佳匹配块的搜索占据了编码单元划分过程时间的 40%以上[4]，所以一些学者对最佳匹配块的搜索进行了简化研究。对应的算法主要可以分为三类[38]，搜索提前终止策略[39][40]，搜索窗口决策[41][42]以及搜索模式设计[43][44]。

1.2.2　码率控制算法研究现状

码率控制不属于视频编码标准的范畴，信道中传输的压缩码流中并不包含关于码率控制的语法信息，使用者可以根据特定的场景设计适合的码率控制算法。但码率控制模块是编码器中必不可少的关键模块，因为在实际应用场景中用

于传输视频的码率一般都是有限且浮动的，所以需要一个算法来控制压缩码流的大小，使视频能够在不同目标码率下传输，该算法即为码率控制算法。因此，每当新的视频编码标准发布时，许多学者便开始研究相应的码率控制算法，如 MPEG-2 的 TM5、H.263 的 TMN8 和 AVC 的 JVT-G012 等等。此外，码率控制算法也在随着视频编码标准的更新而更新，如在 H.261 和 MPEG-1 等较早的编码标准中，仅仅通过缓存器的反馈值来调节编码量化参数[45]，达到码率控制的目的。该算法的码率控制精度不高，在之后的编码标准 H.263、H.264/AVC、HEVC 以及我国拥有自主知识产权的音视频编码标准（Audio Video coding Standard, AVS[46]）中均使用了基于率失真模型的码率控制算法。目前主要的码率控制模型有 $R-Q$ 模型和 $R-\lambda$ 模型。

$R-Q$ 模型是最早出现且应用最为广泛的码率控制模型，从 MPEG-2 的 TM5 到HEVC测试软件的早期版本中，均为视频编码标准建议使用的码率控制模型。该模型建立了码率和量化参数之间的函数关系，在给定目标码率后，可以通过带入相应的 $R-Q$ 模型求得对应的量化参数，以此达到控制码率的效果。目前应用比较广泛的 $R-Q$ 模型是 Zhang Yaqin 等[47]提出的基于拉普拉斯分布的二次模型。基于此模型，文献[48]提出的码率控制算法被 MPEG-4 采纳，并将该算法集成在其标准测试软件 VM8 中。文献[49]进一步优化该算法，提高了率失真性能和码率控制精度。在 H.264/AVC 的码率控制提案 JVT-G012[50]中，Li Zhengguo 等提出了一个基于此二次模型的码率控制算法。该算法利用了缓存器漏斗模型和平均绝对误差线性预测模型，该算法是 H.264/AVC 标准建议码率控制算法且被集成到 H.264/AVC 标准测试软件 JM 中。针对 H.265/HEVC 标准码率控制，提案 JCTVC-H0213[51]提出了统一码率量化模型，该模型被集成在 H.265/HEVC 测试软件 HM 的 6.2 至 9.0 版本。

相比于 $R-Q$ 模型，$R-\lambda$ 模型在率失真性能、码率控制精度两方面均有较大提升。因此，自 HM 的 9.1 版本起，基于 $R-\lambda$ 模型的码率控制算法就取代了原本的统一码率量化模型，成为了 H.265/HEVC 推荐的码率控制模型。该模型在提案 JCTVC-K0103[52]中被提出。该模型建立了码率与拉格朗日因子 λ、λ 与量化参数之间的函数关系，并根据实际编码结果对模型参数进行更新，同时也包括了与 $R-Q$ 模型类似的分层分配目标比特的思想[53]。之后，针对 $R-\lambda$ 模型的改进提案不断被提出，如处理对象是 I 帧的 JCTVC-M0257[54]、提出了一种自适应目标比特分配算法的 JCTVC-M0036[55]，该算法被集成在 HM 的 11.0 以及之后的

版本中。$R-\lambda$模型包括三个模块，分别为目标比特分配、量化参数确定、模型参数更新。后续的码率控制算法的改进多集中于目标比特分配和模型参数更新两个模块。如文献[56]~[58]通过二次编码的方式来获得更加准确的模型参数，文献[59]、[60]也改进了模型参数的更新方式。针对不同层次的目标比特分配问题，文献[61]~[62]、[63]~[64]、[65]分别对 CTU 级、帧级、图像组（Group of Picture, GOP）级的目标比特分配方案进行了改进。此外，还有一些算法[66][67]在进行目标比特分配时结合了感兴趣区域（Region of Interest, ROI）技术，为包含特定对象的部分分配更多的比特，而其余部分分配较少比特，以提高特定对象的显示质量。

1.3　本文主要工作及结构安排

第一章是本文的绪论部分。在 1.1 节中介绍了本文所研究的两种算法的研究背景与意义。1.2 节对两种算法的研究现状进行了阐述，其中 1.2.1 节对 HEVC 低复杂度算法的研究对象和研究方法进行了总结，并列举了大量相关研究文献。1.2.2 节对码率控制算法的发展以及主要的研究方向进行了介绍。

第二章详细介绍了 HEVC 中与本文研究课题联系比较紧密的预测编码关键技术。

第三章提出了一种帧间预测编码单元快速划分算法。3.1 节对帧间预测编码单元的划分过程和复杂度进行了分析。3.2 节详细介绍了本文所提低复杂度算法的设计过程和理论基础，首先通过实验观察标准测试序列的编码单元深度分布特点，然后通过实验观察 BSAD 与编码单元深度之间的相关性，最后为了提高预测准确性，也通过实验观察当前 CTU 深度划分与时空相邻 CTU 深度的相关性。3.3 节利用以上两种相关性设计了一种编码单元快速划分算法。3.4 节对提出算法的率失真性能进行了验证。

第四章提出了一种基于视觉显著性检测的 LCU 层码率控制算法。4.1 节详细介绍了视频编码率失真曲线和λ域码率控制算法。4.2 节介绍了本文所提算法中使用的视觉显著性算法。4.3 节介绍了本文所提码率控制算法的实现细节，其中，4.3.1 节提出了一种新的基于视觉显著性的 LCU 层目标比特分配权重计算公式，4.3.2 节提出了一种新的 LCU 层模型参数更新方式。4.4 节对所提算法的性能进行了验证。

第五章是对全文的总结和对后续工作的展望。

第 2 章　HEVC 预测编码关键技术

HEVC 标准的第一个版本在 2013 年由国际电信联盟（International Telecommunication Union, ITU）和国际标准化组织（International Organization for Standards, ISO）共同发布。在 ISO 中，该标准被称为 HEVC;在 ITU 中，该标准被称为 H.265。

2.1　HEVC 混合编码框架

自 H.261 编码标准使用混合编码框架之后，所有的视频编码标准都继承了混合编码框架的特点，HEVC 也是如此。HEVC 标准包括了 3 大模块，分别为预测编码、变换编码和熵编码。其中预测编码和熵编码均为无损编码，变换编码中的量化是失真的主要原因，也直接决定了压缩率。HEVC 混合编码框架如图 2-1 所示。

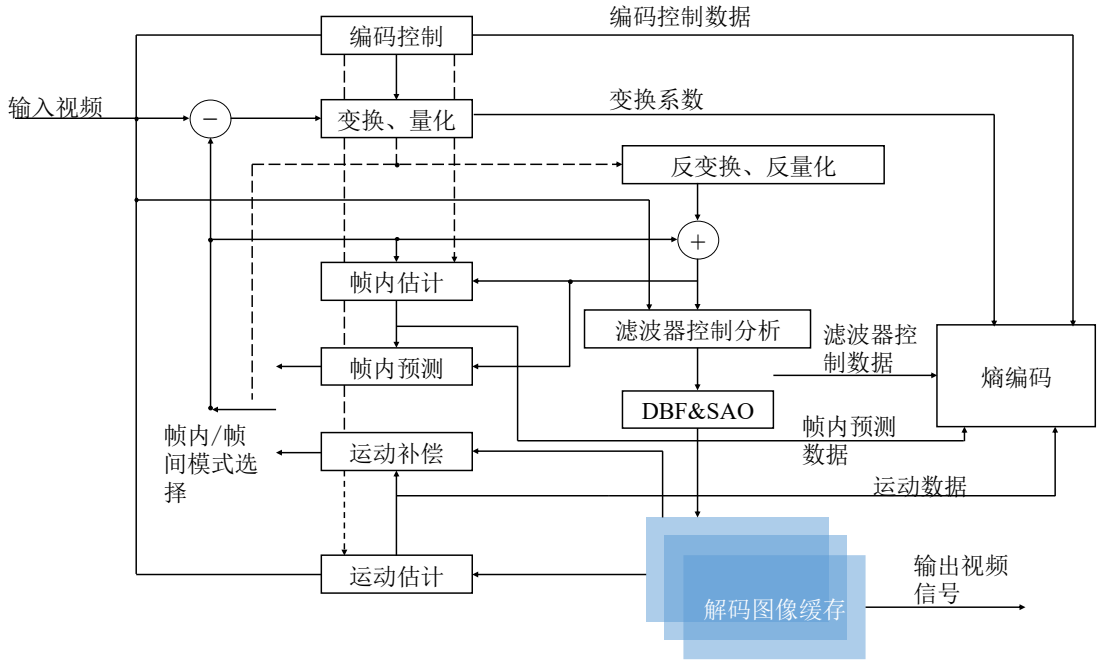


图 2-1　HEVC 标准框架

图 2-1 为 HEVC 编码器框架，其中解码图像缓存中的图像是作为帧间预测

模式中的参考图像，输出视频信号和解码端解码图像是完全相同的。

2.2　四叉树划分

视频序列在进入 HEVC 编码器后首先会被划分为同等大小的 CTU。CTU 类似于 AVC 中的宏块，但是为了利用高清及超高清视频序列的高分辨率特点，CTU 尺寸的取值更加灵活，其值可由配置文件中相关参数决定，最大可为 64×64。在高分辨率序列中，有许多像素变化比较缓慢的部分，但同时也包含了纹理比较丰富的部分，为了更加有效地表示这些不同特点的对象，可将 CTU 进一步划分为编码单元，划分方式被称为四叉树划分。该处理方式极大提高了 HEVC 在高分辨率视频的压缩性能。

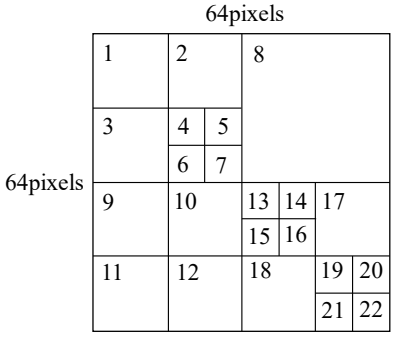


图 2-2　CTU 划分

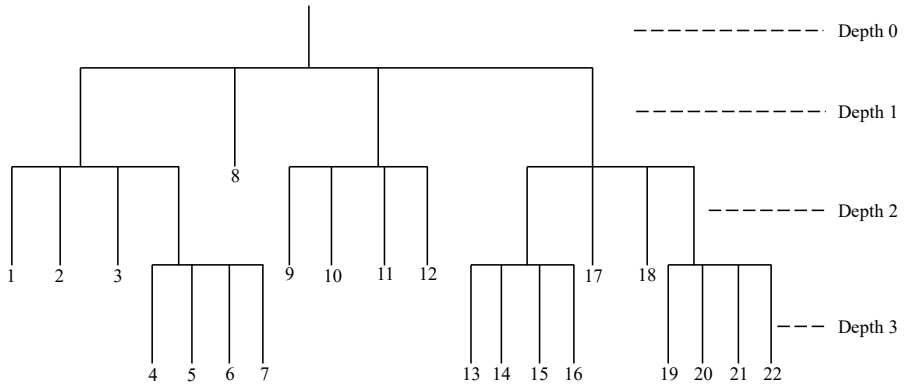


图 2-3　CU 深度分布

当 CTU 的尺寸为 64×64 时，CU 的尺寸可以为 8×8、16×16、32×32、64×64。若 CU 的尺寸为 64×64，则表示 CTU 没有被划分；若 CU 的尺寸为其他，则表

示 CTU 被划分为了多个 CU。为了更加方便地表述 CU 的大小，将其尺寸与深度相关联，如 CU 尺寸为 64×64，则表示该 CU 深度为 0；CU 尺寸为 32×32，则表示该 CU 深度为 1，依此类推。如图 2-2 所示，进行四叉树划分的 CTU 的尺寸为 64×64，其中数字代表编码的顺序，扫描方式为 Z 扫描。图 2-3 中显示了各个 CU 的深度。

2.3　帧内预测

帧内预测利用的是像素之间的空间相关性，将当前像素块周围的像素按照一定规则生成当前像素块的预测块，预测块与当前像素块的差值称为预测残差。预测残差是后续变换、量化的对象，以此来去除图像的空域冗余，提高压缩率。

CU 在进行帧内预测处理时会先被划分为预测单元（Predicting Unit, PU），帧内预测的 PU 划分方式有两种，如图 2-4 所示。其中，2N 表示被划分的 CU 的尺寸。

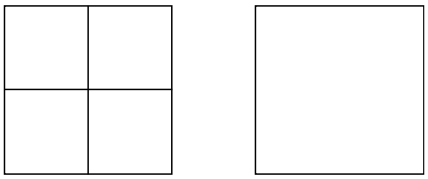


图 2-4　帧内预测 PU 划分方式

当 CU 划分为 PU 之后，针对 PU 生成相应的预测块。生成预测块的方式由 AVC 中的 9 种增加至 35 种，被称为 35 种预测模式，模式名称及模式编号如表 2-1 所示。

表 2-1　帧内预测 35 种模式编号

帧内编码模式	帧内模式名称
0	Planar 模式
1	DC 模式
2 到 34	33 种角度模式

不同的预测模式是为了应对具有不同内容特征的 PU，如 DC 模式针对的是

内容比较平坦的 PU，而 Planar 模式针对的是内容缓慢变化的 PU。不同的预测模式虽然在生成预测块时使用不同的计算方法，但是却使用通用的预测模板，如图 2-5 所示。其中，$P_{x,y}$ 代表的是生成的预测像素值，$R_{x,y}$ 代表的是用于生成预测像素值的参考像素值，PU 的尺寸为 N×N。如 DC 模式中，$P_{x,y}$ 的值等于 $R_{0,1}$ 至 $R_{0,N}$，$R_{1,0}$ 至 $R_{1,N}$ 这 2N 个值的均值。

对于一个 PU 而言，最佳预测模式只有一个。该模式的选择过程大致如下，首先通过粗略模式选择算法（Rough Mode Decision, RMD）从 35 种预测模式中选择 N 种，N 的取值与 PU 的尺寸相关。然后确定 PU 的 3 个最可能模式（Most Probable Mode, MPM），MPM 是利用相邻 PU 之间最佳预测模式的相关性来确定的。最后，从 N+3 种预测模式中通过率失真优化（Rate Distortion Optimization, RDO）确定最佳预测模式。相比于 AVC，虽然 HEVC 最佳预测模式的选择过程计算复杂度提高了很多，但是更多的预测模式也提高了预测精度，所以其压缩性能得到了提升。

$R_{0,0}$	$R_{1,0}$	$R_{2,0}$	.....	$R_{N,0}$	$R_{N+1,0}$	.....	$R_{2N,0}$
$R_{0,1}$	$P_{1,1}$	$P_{2,1}$	.....	$P_{N,1}$			
$R_{0,2}$	$P_{1,2}$						
⋮	⋮						
$R_{0,N}$	$P_{1,N}$			$P_{N,N}$			
$R_{0,N+1}$							
⋮							
$R_{0,2N}$							

图 2-5　HEVC 帧内预测模板

2.4　帧间预测

帧间预测利用的是像素之间的时域相关性。一秒钟的视频序列包含多帧，相邻帧同位或邻位的像素值是非常接近的，所以将前帧图像块作为预测块，可以去除视频的时域冗余。

针对同一视频序列，相比于帧内预测，帧间预测的压缩率要更高，因此实用性更好。所以，本文研究了帧间预测编码单元快速划分算法。帧内和帧间预测的压缩率对比如表 2-2 所示。

表 2-2　帧间预测和帧内预测压缩率对比

分辨率	序列名	编码帧数	压缩率比值（帧间/帧内）
1920×1080	ParkScene	30	3.966
1280×720	Johnny	30	9.318
832×480	BasketballDrill	30	4.006
416×240	RaceHorses	30	1.582

帧间预测的 CU 在进行预测处理时也会首先被划分为 PU，而且划分的方式至多达到 8 种，如图 2-6 所示。其中上面 4 种划分方式为对称划分，下面 4 种划分方式为非对称划分。当配置文件中的参数 AMP（Asymmetric Motion Partition, 非对称运动划分）为真时，对称划分和非对称划分都会被遍历，否则仅仅遍历 4 种对称划分方式，从中选择率失真代价最小的划分方式。

帧间预测过程包含许多模块，接下来将介绍搜索算法、亚像素精度运动估计以及 MV 预测技术。相比于 AVC，这 3 个模块均进行了改进。

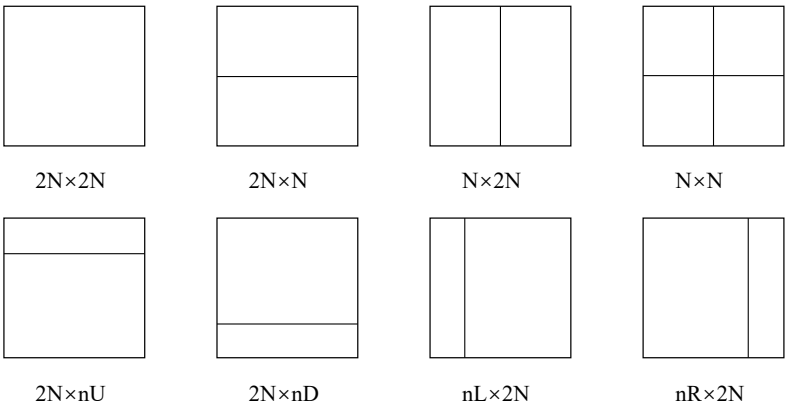


图 2-6　帧间预测 PU 划分方式

2.4.1　搜索算法

在将 CU 划分为 PU 之后，需要为 PU 在前帧中寻找最佳匹配块，这个过程使用的算法称之为搜索算法。

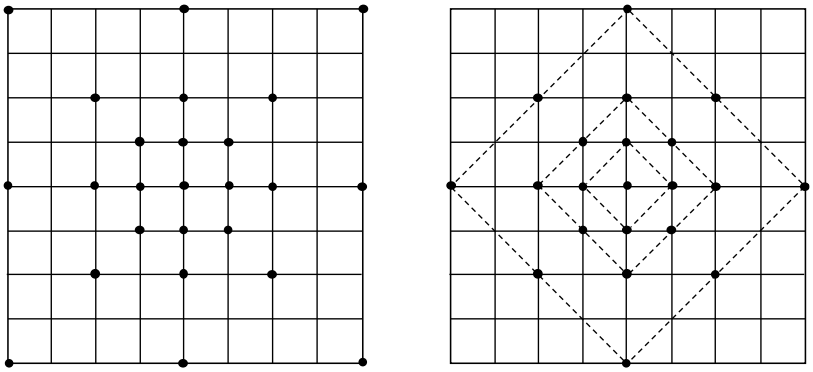


图 2-7　TZ 算法搜索模板

HEVC 标准测试软件中有两种搜索算法，分别为全搜索和 TZ 搜索。使用哪一种搜索算法由配置文件中的 FastSearch 参数决定。全搜索即搜索一定范围内的所有点，并按照一定的运动估计准则（如绝对误差和（Sum of Absolute Difference, SAD）、率失真代价、均方误差（Mean Square Error, MSE）等）选取最佳匹配块。但由于其复杂度较高，HEVC 参考模型 HM 中默认采用了 TZ 搜索，即搜索一定范围内指定的若干个点。TZ 搜索算法的搜索模板分为菱形搜索模板和正方形搜索模板，如图 2-7 所示。

2.4.2　亚像素精度运动估计

物体在连续帧间的运动是连续的，而像素本身是离散的，这种现象带来了一个问题，当前帧中图像块的最佳参考块不一定位于参考帧的整数像素点位置。为了更加准确地预测当前待编码的图像块，有必要在非整数位置进行运动估计，即亚像素精度运动估计。亚像素精度运动估计由两个步骤组成，首先通过对整数像素进行插值滤波生成亚像素参考图像，然后在亚像素参考图像中寻找最佳匹配块。

HEVC 继承了 AVC 的亚像素精度运动估计并对其进行了改进。两者相同点是最高精度均为 1/4 像素。不同点是 HEVC 使用了 8 抽头的滤波器生成了 1/2 像素点，使用了 7 抽头的滤波器生成 1/4 和 3/4 像素点，而 AVC 使用了 6 抽头的滤波器生成半像素点，使用两点内插生成 1/4 和 3/4 像素点。HEVC 抽头系数由表 2-3 给出。

表 2-3　亮度插值滤波器抽头系数

亚像素位置	抽头系数
1/4	{-1,4,-10,58,17,-5,1}
1/2	{-1,4,-11,40,40,-11,4,-1}
3/4	{1,-5,17,58,-10,4,-1}

2.4.3　MV 预测技术

类似于编码单元深度分布的时空相关性，当前 PU 的 MV 与空域相邻或者时域相邻的 PU 的 MV 也具有很大的相关性。因此，类似于预测编码中对编码单元的预测残差进行变换和量化，HEVC 不直接传输 PU 的 MV，而是传输当前 PU 的 MV 与预测 MV 之间的差值，这样可以节约编码 MV 的比特数。HEVC 中包含了两种 MV 预测技术，一是 merge 技术，二是 AMVP 技术。

这两种 MV 预测技术之间有两个不同点，一是 merge 技术不仅仅是一种 MV 预测技术，它还是一种编码模式。在构建了当前 PU 的时域和空域候选 MV 列表后，当前 PU 的 MV 将在此列表中选择，并且仅仅传输当前 PU 的 MV 在候选列表中的序号。AMVP 技术也会构建时域和空域候选 MV 列表，但仅仅从该列表中选择一个 MV 作为预测 MV 和搜索起始点，最终传输的是预测 MV 与 PU 的 MV 的差值。第二个不同之处是两种技术的时域和空域候选列表的构建方式以及数量是不同的，如 merge 候选列表长度为 5，而 AMVP 候选列表长度只有 2。

2.5　本章小结

本章首先介绍了 HEVC 编码框架，然后简要介绍了 HEVC 中的四叉树划分。最后介绍了预测编码中的帧内预测和帧间预测技术，在帧内预测中介绍了亮度分量对应的 35 种预测模式，在帧间预测中对第 3 章中提到的技术细节进行了补充，简要介绍了帧间预测编码单元划分过程中的 3 种关键技术，分别为最佳匹配块的搜索算法、亚像素精度运动估计以及 MV 预测技术。

第 3 章　帧间预测编码单元快速划分算法

帧间预测是 HEVC 混合编码框架中非常重要的一步，其充分利用了图像在时域上的相关性。由帧间预测得到的图像块残差信息是之后的变换操作、量化操作的对象。若帧间预测过程中寻找的预测图像块与待编码图像块相似度越高，则得到的残差信息就会越小，经过之后的变换和量化，该残差信息就可以用极少的比特来表示，能够明显地提高压缩率。

HEVC 编码标准相较于上一代编码标准 AVC，在相同的视频质量下压缩率提高了两倍，这得益于许多新的编码工具的引入，其中贡献较大的就是灵活的四叉树划分结构。该划分结构在提高编码效率的同时也极大地增加了 HEVC 的计算复杂度，使得相对于 AVC 标准，HEVC 需要更多的编码时间，在一定程度上限制了 HEVC 在计算能力有限的设备上的实时应用。针对此问题，本章提出了一种帧间预测编码单元快速划分算法，以在基本不降低编码效率的前提下，减少编码时间。

3.1　帧间预测编码单元划分过程分析

3.1.1　帧间预测编码单元划分流程

复杂的块划分结构和预测模式的选择等面临的一个问题是如何选择最优的块划分结构和预测模式，即需要为块划分和预测模式的选择指定一个标准。视频压缩的目的就是在一定的视频质量下尽可能降低码率，或者在一定码率下可以传输更高质量的视频。因此，在视频压缩中视频质量和需要的比特数是评价压缩标准性能的重要指标，所以也成为了块划分和预测模式选择的判定参数。

失真和编码比特数属于不同的单位，无法直接比较大小。为了解决这个问题，Sullivan[3]等将视频编码看作凸优化，并利用拉格朗日优化将失真和编码比特数统一成率失真代价，如式 3-1。

$$J = (SSE_Y + \omega_{chroma} \cdot SSE_{chroma}) + \lambda_{md} \cdot B_{md} \qquad (3\text{-}1)$$

其中，J 是率失真代价；SSE（Sum of Square Error，平方误差和）表示失真，

SSE_Y 表示亮度分量的失真，SSE_{chroma} 表示色度分量的失真；ω_{chroma} 代表色度分量所占的权重；λ_{md} 表示拉格朗日因子。率失真代价代表编码某个图像块或者语法元素所产生的编码代价，它将失真和编码比特数统一成可以直接比较大小的量纲，率失真代价越小，表示压缩性能越好。

根据率失真代价的含义，可以通过遍历的方法来确定最优的块划分结构和预测模式。对于一个编码单元来说，由式 3-1 可以计算出 SKIP 模式、merge 模式和 INTER 模式的率失真代价 J。其中，率失真代价最小的模式就是当前编码单元的最佳预测模式。

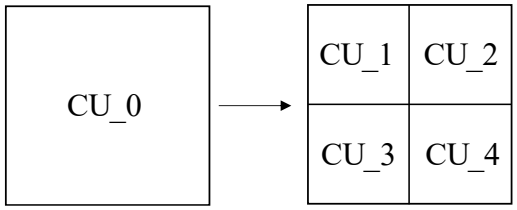


图 3-1　CU 划分

CTU 的块划分过程是由一个个如图 3-1 的基本模块组成的，划分过程需要利用式 3-1 的率失真代价作为是否划分的判断准则，步骤如下。

步骤一：首先计算 CU_0 的率失真代价，记为 J_0。

步骤二：依次计算 CU_1、CU_2、CU_3、CU_4 的率失真代价，分别记为 J_1、J_2、J_3、J_4，且记 4 个率失真代价之和为 J_{sum}。

步骤三：若 J_0 小于 J_{sum}，则 CU_0 不进行划分；否则，CU_0 划分为 4 个子 CU, 即图 3-1 中的 CU_1、CU_2、CU_3、CU_4。

该基本模块的执行遵循自下而上的原则，先计算所有深度为 2 的 CU 的划分情况。当计算完成后，开始计算所有深度为 1 的 CU 的划分情况，最后计算深度为 0 的 CU 的划分。最终 CTU 会被划分为若干个 CU，这些 CU 的率失真代价之和在所有的划分情况中是最小的。

3.1.2　帧间预测编码单元划分复杂度分析

帧间预测编码单元划分过程是 HEVC 标准中复杂度最高的一个模块，本小节将从以下 4 个方面来分析该模块复杂度较高的原因，包括需要遍历的 CU 的数量、预测模式的多样化、最佳匹配块的搜索以及率失真代价的计算。

首先分析编码单元划分过程中需要遍历的 CU 的数量，在 HEVC 标准中，CU 的尺寸可以有 4 种选择，分别为 8×8、16×16、32×32、64×64，单位为像素。这 4 种尺寸对应的深度分别为 3、2、1、0。如图 3-2 所示，一个尺寸为 64×64 的 CTU 可以划分成 1 个深度为 0 的 CU、4 个深度为 1 的 CU、16 个深度为 2 的 CU 和 64 个深度为 3 的 CU，这表示完成一个 CTU 的划分需要遍历这 85 个 CU。

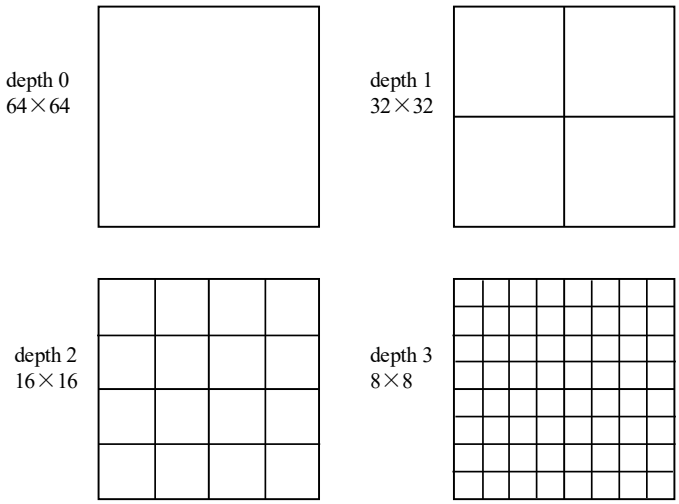


图 3-2　不同深度 CU 分布

对于任意一个 CU，都需要从多种预测模式中选择最优的预测模式。帧间预测模式包括 SKIP 模式、merge 模式和 INTER 模式。

在 SKIP 模式中，PU 的大小是和 CU 相同的，SKIP 模式下把预测信号作为重构图像。SKIP 模式的特点是仅仅传输当前 CU 的标志位以及匹配块的运动矢量，而其他模式则传输的是预测残差变换量化后的值。

merge 模式也是 HEVC 帧间预测采用的新技术，PU 的大小与 CU 也是相同的。从构建的时域、空域候选 MV 列表中选择率失真代价最小的 MV 作为 PU 的 MV，并传输该 MV 在候选列表中的序号。

INTER 模式下，CU 划分为 PU 存在 8 种划分方式，如图 2-6 所示。

对于 INTER 模式中的任意一种 PU 划分方式，都包含了 PU 最佳匹配块搜索过程。搜索过程更加详细的介绍位于 2.4.1 节。此外，为了进一步提升运动估计的准确性，除了进行整数像素运动估计，还添加了亚像素精度运动估计，导致了最佳匹配块搜索的计算复杂度进一步提高。

综上所述，如果按照 HEVC 标准流程去完成一个 CTU 的划分，需要遍历 85

个不同尺寸的 CU。对于任意一个 CU，需要从 SKIP 模式、merge 模式和 INTER 模式中选择率失真代价最小的模式为最佳预测模式。且在 INTER 模式中还需要遍历 8 种不同的 PU 划分方式，对于任意一种划分方式，都需要从已经编码的帧中在整数像素精度和分数像素精度寻找最佳匹配块，且需要比较每一种划分方式的率失真代价以确定 INTER 模式的最小率失真代价。这导致 CTU 的划分具有极高的计算复杂度，是 HEVC 中最耗时的一个模块，也是国内外 HEVC 低复杂度算法研究的主要对象，同时也是本文的研究对象。

3.2　编码单元快速划分算法设计过程

3.2.1　编码单元深度分布分析

HEVC 完成一个 CTU 的划分需要遍历 85 个 CU，如果可以根据某些编码过程中产生的相关参数来减少遍历的 CU 的数量，则可以减少编码单元划分的时间。表 3-2 是测试序列中各个深度的 CU 所占的比例，该比例的定义如式 3-2，其中分子表示的是深度为i的像素的数量，分母表示的是总的像素数量。

$$probability\_i = \frac{the\ number\ of\ pixels\ at\ depth\ i}{total\ number\ of\ pixels} \times 100\% \quad i = 0,1,2,3 \quad (3\text{-}2)$$

表 3-1 是表 3-2 实验数据的实验参数，HM 是 HEVC 官方测试平台，其中集成了 HEVC 可以完成的所有功能，QP 表示量化参数（Quantization Parameter, QP）。除了所列出的实验条件之外，其他的参数为 encoder_lowdelay_main 配置文件默认参数。此外，表 3-2 统计的是从第 2 帧到第 50 帧的深度分布，因为第 1 帧是 I 帧，只会进行帧内预测，而本文算法针对的是帧间预测，所以仅仅统计进行帧间预测的图像帧的深度分布。

表 3-1　实验参数

实验参数	
测试平台	HM16.7
QP	32
编码帧数	50
编码模式	Lowdelay_main

表 3-2　各深度 CU 所占比例（%）

分辨率	序列名	depth0	depth1	depth2	depth3
2560×1600	PeopleOnStreet	19.3	30.5	35.7	14.5
	Traffic	60.7	26.2	10.7	2.4
1920×1080	ParkScene	49.3	29.1	16.0	5.6
	Kimono1	40.4	47.1	10.5	2.0
	Cactus	54.1	28.0	13.6	4.3
	BQTerrace	66.8	21.2	8.5	3.5
	BasketballDrive	56.5	29.6	10.6	3.3
1280×720	FourPeople	75.9	15.6	7.8	0.7
	Johnny	79.0	14.7	6.1	0.2
	KristenAndSara	76.9	15.9	6.8	0.4
	vidyo1	77.0	15.5	7.1	0.4
	vidyo4	74.5	17.9	7.0	0.6
832×480	RaceHorses	9.3	46.8	32.0	11.9
	PartyScene	22.0	39.3	25.1	13.6
	BQMall	37.6	33.7	20.4	8.3
	BasketballDrill	42.6	34.9	17.1	5.4
	BasketballDrillText	42.3	33.9	18.3	5.5
均值		52.0	28.2	14.9	4.9

由表 3-2 可知，17 个测试序列做为测试数据，深度为 0 的 CU 平均占比高达 52%，在测试序列 Johnny_1280x720_60 中深度为 0 的 CU 占比甚至高达 79%。如果可以提前判断 CTU 的深度为 0，那么就可以不用再遍历该 CTU 的其他深度，这样可以大幅度节约编码时间。对于其余的 48%的深度非 0 的 CTU，若可以提前判断其深度非 0，则可以避免深度 0 的遍历，也可以节约部分编码时间。虽然避免了深度 0 的遍历，但是仍需遍历深度 1 到 3，因此为了更进一步节约编码时间，对 48%的非零 CTU 的遍历深度再次进行修剪。以上措施的结合使用可以节约大量的编码时间，但是由于对 CTU 的深度遍历做了修剪，所以相对于 HEVC 标准划分流程，可能会找不到最佳匹配块，导致预测残差更大，编码当前 CTU 需要更多的比特。因此，精确的深度 0 检测以及深度非零的 CTU 的遍历深度修剪是该算法的关键，保证了既降低计算复杂度又不增加其码率的效果。

为了验证该想法具有减少视频编码时间的效果，表 3-4 中实验数据显示了相对于 HEVC 标准划分流程（遍历 CTU 的所有深度），如果避免遍历某些深度可以节约的时间百分比。表 3-3 是实验的参数表，其他的参数为 encoder_lowdelay_main 配置文件默认参数。由表 3-4 中的数据可知，如果 CTU 划分过程中仅仅遍历深度 0，则编码时间可以节约 75%，其他遍历区间相对于标准划分流程也都有不同程度的时间节约。

表 3-3　实验参数

实验参数	
计算机品牌	华硕 X555UQ
CPU 型号	Intel(R) Core(TM) i5-6200U CPU@2.30GHz
测试平台	HM16.7
编码帧数	20
量化参数 QP	32
测试序列	Traffic_2560x1600_30_crop

表 3-4　编码时间节约

遍历的深度范围	编码时间(单位秒)	时间节约百分比
[0, 3]	1942.936	0%
[0, 0]	493.380	74.6%
[0, 1]	934.238	51.9%
[0, 2]	1518.654	21.8%
[1, 2]	1273.646	34.4%
[1, 3]	1618.570	16.7%

3.2.2　BSAD 与编码单元深度相关性分析

视频是由一帧帧图像组成的，视频的帧率表示该视频一秒钟所包含的帧的数目。帧率越高，相邻帧之间的相关性就越强，当视频内的物体移动速度较慢或者静止时，相邻帧在相同位置的像素值是基本相同的。因此，相邻帧相同位置的 CTU 的划分具有很强的相似性。在划分当前 CTU 之前，可以先获取前一帧相同或相邻位置的 CTU 的划分情况，利用这些信息对当前 CTU 深度进行预测，以此

减少遍历那些概率较低的深度，这就是许多 CTU 快速划分算法中利用的 CTU 划分的时间相关性。除了时间相关性、空间相关性以外，编码过程中产生的一些中间参数，例如率失真代价、PU 划分方式、预测残差等也与 CTU 划分深度有一定的关系。本节通过实验验证和逻辑推理说明了 BSAD 和编码单元划分深度之间的相关性，为下一节中使用 BSAD 设计算法提供了理论基础。

$$B(x,y)=\begin{cases}1, & \left|P_{cur}(x,y)-P_{ref}(x,y)\right|>Th\\ 0, & \left|P_{cur}(x,y)-P_{ref}(x,y)\right|\le Th\end{cases} \tag{3-3}$$

$$BSAD=\sum_{x=0}^{63}\sum_{y=0}^{63}B(x,y) \tag{3-4}$$

式 3-3 中 $P_{cur}(x,y)$ 表示当前 CTU 中坐标为 (x,y) 的像素点的 Y 通道值，横纵坐标都是从 0 开始的。$P_{ref}(x,y)$ 表示前一帧中和当前 CTU 相同位置的 CTU 中坐标为 (x,y) 的像素点的 Y 通道值，这里前一帧指的是重构图像。Th 表示阈值，当两个 CTU 中同位置像素点的 Y 通道值的差大于此阈值时，认定两者差异较大，否则认定两者差异较小。BSAD 的定义如式 3-4 所示，它等于一个 CTU 中所有与前帧同位 Y 通道值差异较大的像素数目和。若 BSAD 的值较大，说明当前 CTU 和它的前帧同位 CTU 相似性较低，若 BSAD 的值很小，则表明两者基本相同。

表 3-5 中数据显示了深度为 0 的 CTU 和深度为非 0 的 CTU 的 BSAD 均值，实验参数同表 3-1 相同。由于第一帧是帧内预测，所以统计数据来自于第 2 到 50 帧，并且由于某些分辨率的视频的高度并不是 64 的整数倍，导致最后一行并不是完整的 64×64 的 CTU, 在 HM 中该部分深度是不为 0 的，所以为了数据的更加准确，这里将不再统计这些视频最后一行对应的 BSAD。这里 BSAD 所对应的阈值 Th 取值为 20，和量化参数对应的量化步长基本相同。

从表 3-5 可知，虽然不同视频的深度 0 和深度非 0 的 CTU 所对应的 BSAD 均值有较大差距，但是对于同一个视频而言，这两者的 BSAD 均值是有明显的差距的。例如，测试序列 BQMall、BasketballDrill、vidyo4 等等，深度非 0 的 CTU 对应的 BSAD 均值是深度 0 的 CTU 对应的 BSAD 均值的 20 倍以上，而两者的最小差距也在 2 倍以上。这说明 BSAD 值和 CTU 划分之间有着很强的相关性，可以作为提前预测 CTU 划分深度的依据，尤其是对深度为 0 的 CTU 的检测。

当某一 CTU 的 BSAD 值较小时，它的深度为 0 的概率会更大，当 BSAD 值非常大时，更倾向于深度非 0。

表 3-5　不同深度的 CTU 对应的 BSAD 的均值

分辨率	序列名	深度为 0	深度非 0
2560×1600	PeopleOnStreet	27.33	597.77
	Traffic	57.06	369.38
1920×1080	ParkScene	62.52	333.19
	Kimono1	267.37	464.51
	Cactus	50.68	583.76
	BQTerrace	235.59	412.33
	BasketballDrive	276.24	1005.24
1280×720	FourPeople	4.39	128.8
	Johnny	5.76	46.13
	KristenAndSara	4.54	71.37
	vidyo1	8.46	172.09
	vidyo4	56.63	232.21
832×480	RaceHorses	542.95	1054.60
	PartyScene	120.68	479.25
	BQMall	39.29	589.70
	BasketballDrill	23.46	691.79
	BasketballDrillText	22.58	652.18
均值		106.22	453.66

为了更加直观地显示两者的关系，图 3-3 选取了某一测试序列作为示例。（a）表示的是测试序列 BasketballDrill 的第 40 帧的 Y 通道图像中的某一块图像，图中每一个方格都是一个大小为 64×64 的 CTU；（b）图表示该图像所对应的四叉树划分，实验参数同表 3-1 相同；（c）图是一个二值图像，生成该图像的定义式如式 3-5。其中 $P_{cur}(x,y)$ 表示当前图像中坐标为 (x,y) 的像素点的 Y 通道值，$P_{ref}(x,y)$ 表示前一帧图像中坐标为 (x,y) 的像素点的 Y 通道值，Th 表示阈值，与求 BSAD 的阈值相同。

$$p(x,y)=\begin{cases}0, & \left|P_{cur}(x,y)-P_{ref}(x,y)\right|>Th\\ 255, & \left|P_{cur}(x,y)-P_{ref}(x,y)\right|\le Th\end{cases} \quad (3\text{-}5)$$

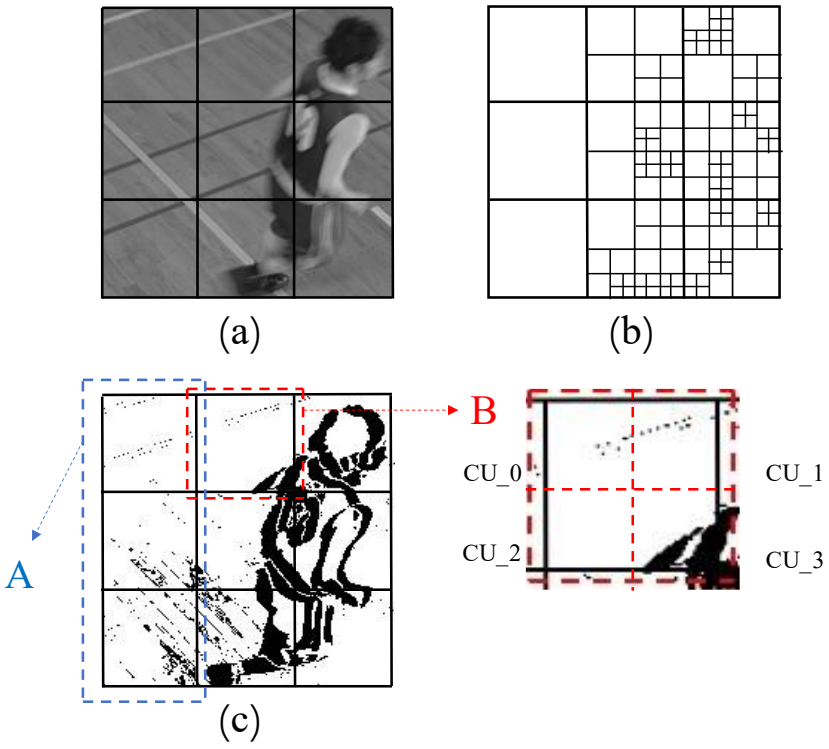


图 3-3　（a）Y 通道图像（b）四叉树划分（c）二值图像

由二值图像的定义式 3-5 可知，一个 CTU 的 BSAD 值是该 CTU 内黑色像素点的数目和。当一个 CTU 内黑色部分越多，其对应的 BSAD 值就越大，图 3-5（c）中的 A 区域内的黑色部分明显少于其他六个 CTU，即表示其 BSAD 值要远远小于其他六个 CTU 的 BSAD 值。（b）表示的是该部分的编码单元的四叉树划分，而 A 区内的三个 CTU 的深度都为 0，而其余 CTU 深度为非 0，结果与表 3-5 的结论相吻合，即 BSAD 值越大，其深度就越趋向于非 0。

除了在 CTU 尺度上利用 BSAD 来判断其是否应该进一步划分，32×32 尺寸的编码单元是否进行划分也与该编码单元对应的 BSAD 值有着很强的相关性，如 B 区域内的一个 CTU 的 4 个 32×32 的子 CU，其中 CU_0、CU_1、CU_2 的 BSAD 要远远小于 CU_3，而其余三个是不再进一步划分，CU_3 需要进一步划分。和表 3-5 具有相同的实验参数，表 3-6 中统计了 32×32 大小的 CU 划分与不

划分所对应的平均 BSAD 值。

表 3-6　32×32 大小的 CU 对应的 BSAD 均值

分辨率	序列名	不划分	划分
2560×1600	PeopleOnStreet	73.88	195.33
	Traffic	75.23	126.69
1920×1080	ParkScene	56.46	122.95
	Kimono1	107.37	158.16
	Cactus	117.81	194.87
	BQTerrace	89.94	130.52
	BasketballDrive	204.17	367.68
1280×720	FourPeople	20.40	63.29
	Johnny	8.06	28.66
	KristenAndSara	12.31	38.97
	vidyo1	29.37	89.71
	vidyo4	48.71	96.22
832×480	RaceHorses	184.73	333.07
	PartyScene	63.23	165.10
	BQMall	101.98	188.09
	BasketballDrill	96.35	263.21
	BasketballDrillText	88.15	248.35
均值		81.07	165.34

由表 3-6 数据得，虽然没有表 3-5 中深度 0 和深度非 0 的 CTU 所对应的 BSAD 差距那么大，但是对于任意一个测试视频，32×32 大小的继续划分的 CU 所对应的 BSAD 基本都是不划分的两倍。这表明了 BSAD 与是否继续划分之间有一定的相关性，可以作为预测 32×32 的 CU 是否进一步划分的一个依据。

以上部分通过实验数据说明了 BSAD 与 CU 是否进一步划分之间的相关性。由 3.1 节可知，一个 CU 是否进行划分取决于该 CU 的最小率失真代价与其四个子 CU 的最小率失真代价之和的相对大小。率失真代价由失真和编码比特数按照一定权重相加得到。帧间预测模式中，CU 从重构帧中搜索最佳匹配块，若匹配度较好，则对应的残差信息就会特别小，那么该 CU 的失真和编码比特数就会很

小，对应的率失真代价就会很小，因此不会继续划分。因为如果继续划分的话，添加许多语法元素，导致率失真代价变大。若当前 CU 的 BSAD 很小，则表明其与前帧相同位置的 CU 具有非常好的匹配度，因此更加倾向于不再继续划分，即解释了为什么不划分的CU所对应的BSAD均值要小于继续划分的BSAD均值。

3.2.3　时空相邻 CTU 深度相关性分析

上一小节分析了 BSAD 与当前 CTU 划分的相关性，但仅仅由 BSAD 来预测当前 CTU 的划分准确率并不足够高。为了进一步提高预测的准确率，本文算法结合了 CTU 划分的时空相关性，即利用当前 CTU 的前帧同位 CTU 和同帧相邻 CTU 的划分深度与当前 CTU 划分深度之间的相关性。如图 3-4 所示，图中的 CU_CUR 代表当前 CTU，CU_CO、CU_L 和 CU_U 是本文算法中利用到的编码单元，其中 CU_CO 是当前 CTU 在前一帧中的相同位置的 CTU，而 CU_L 和 CU_U 分别位于当前 CTU 的左边和上方。

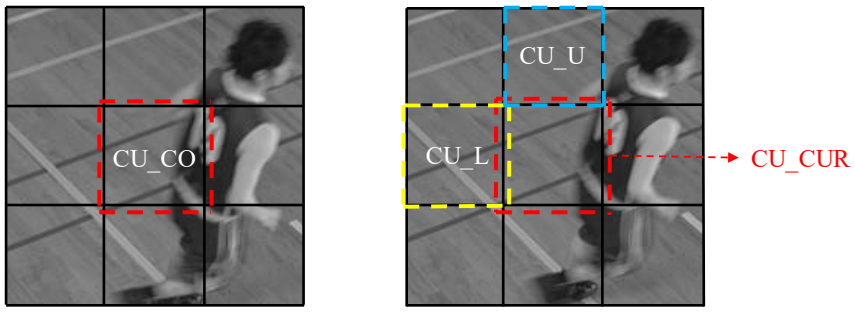


图 3-4　时空相邻 CTU 分布

由于 CTU 划分具有的时空相关性质，编码单元 CU_CO、CU_U、CU_L 和 CU_CUR 的划分具有相似性。表 3-7 中的数据分别显示了三个参考编码单元的最大划分深度大于或者等于当前 CTU 最大划分深度的概率。实验参数和表 3-1 相同。

由表中数据可知，若参考编码单元的最大深度为max_depth，则对于CU_CO、CU_U、CU_L，当前 CTU 的最大深度在区间[0，max_depth]的平均概率分别为 83.18%、82.32%和 82.10%，而对于分辨率为 1280×720 的测试序列，平均概率在

90%左右。这表示当已知某个参考编码单元的最大划分深度 max_depth 时，当前 CTU 的遍历深度如果设置为[0，max_depth]，当前 CTU 划分结果和 HEVC 标准流程划分结果有 80%以上的概率是完全相同的。而在那些不完全相同的情况中，该 CTU 中部分的编码单元划分仍然和 HEVC 标准流程划分结果是相同的。

表 3-7　CU_CO、CU_U、CU_L 和 CU_CUR 最大划分深度关系

分辨率	序列名	CU_CO	CU_U	CU_L
2560×1600	PeopleOnStreet	86.69%	80.44%	82.32%
	Traffic	80.73%	79.62%	82.08%
1920×1080	ParkScene	75.70%	80.79%	78.87%
	Kimono1	73.27%	73.44%	71.74%
	Cactus	81.98%	83.64%	82.35%
	BQTerrace	80.80%	83.40%	84.51%
	BasketballDrive	79.74%	81.37%	82.79%
1280×720	FourPeople	89.24%	89.22%	89.81%
	Johnny	90.30%	92.12%	90.67%
	KristenAndSara	89.38%	89.49%	88.02%
	vidyo1	90.33%	89.47%	89.74%
	vidyo4	88.70%	87.19%	87.70%
832×480	RaceHorses	80.86%	80.06%	75.29%
	PartyScene	76.21%	78.60%	82.29%
	BQMall	85.55%	76.84%	78.35%
	BasketballDrill	82.62%	77.45%	75.00%
	BasketballDrillText	81.93%	76.27%	74.17%
均值		83.18%	82.32%	82.10%

若仅仅使用单一参考编码单元来设置当前 CTU 的遍历深度，其准确率是非常低的，并且会出现遍历区间越来越小，最终只有[0, 0]这种情况。因此，本文算法利用 CTU 的 BSAD 以及三个参考编码单元来共同修剪当前 CTU 的遍历区间。

3.3　编码单元快速划分算法

3.3.1　深度为 0 的编码单元快速检测算法

由 3.2.1 节编码单元尺寸分析可得，深度为 0 的 CTU 平均占比为 52%。如果可以提前判断该 CTU 的深度为 0，就可以避免深度[1,3]的遍历，节约大量编码时间。由之前的实验分析知，一个 CTU 的 BSAD 值与其深度是否为 0 有着很强的相关性。当其深度为 0 时，对应的 BSAD 值要远远小于那些深度非 0 的 CTU，因此，BSAD 值是判断当前 CTU 深度是否为 0 的一个重要依据。由 3.2.3 节的实验可得，若参考编码单元的最大深度为 0，则当前 CTU 的深度为 0 的概率是很大的。三个参考编码单元 CU_CO、CU_L 和 CU_U 的深度与当前 CTU 深度相关性基本相同，因此三者在算法中所占权重是相同的。

深度为 0 的 CU 快速检测算法主要流程如下：

（1）计算当前 CTU 的 BSAD 值。

（2）将 CU_CO、CU_L 和 CU_U 的最大划分深度按照升序进行排序{d1，d2，d3}。

（3）若 BSAD 小于阈值 Threshold，且排序后的第二个值 d2 为 0，则当前 CTU 遍历深度范围设定为[0,0]。

其中阈值 Threshold 的大小与视频序列相关，由表 3-5 中的数据可知，不同视频序列中，深度为 0 和非 0 的 CTU 所对应的 BSAD 均值相差是非常大的，所以该值是与视频直接相关的。本算法中，每隔固定帧会选取一个测试帧，该帧使用 HEVC 标准流程进行划分，将该帧中的深度为 0 和非 0 的 CTU 所对应的 BSAD 均值之和的二分之一作为接下来若干帧的阈值。但是由于要使用测试帧，导致该帧完全按照 HEVC 标准流程划分，这将不会节约任何编码时间。测试帧除了用于确定 BSAD 的阈值外，还用于消除使用修剪的遍历深度范围所带来的积累误差。

为了测试深度为 0 的 CU 检测算法的效果，利用 DP（Detection Probability，检测概率）和 DA（Detection Accuracy，检测精度）来描述实验结果。其中 DP 和 DA 的定义如式 3-6 和式 3-7。其中 $N(A)$ 表示满足条件 A 的 CTU 的数目，$N(B|A)$ 表示在满足条件 A 的基础上又满足条件 B 的 CTU 的数目。在本文中，

$$DP = N(B|A) / N(A) \times 100\% \tag{3-6}$$

$$DA = N(A|B) / N(B) \times 100\% \tag{3-7}$$

条件 *A* 表示一个 CTU 按照 HEVC 标准流程划分，且其深度为 0；条件 *B* 表示一个 CTU 满足深度为 0 的 CU 快速检测算法的条件。因此，DP 表示一个深度为 0 的 CTU 能够被检测出来的概率，DA 表示被算法检测为深度为 0 的 CTU 的深度真正为 0 的概率。表 3-8 显示了实验结果，实验参数和表 3-1 相同。

表 3-8　DP 和 DA

分辨率	序列名	DP	DA
2560×1600	PeopleOnStreet	68.09%	81.94%
	Traffic	82.75%	88.47%
1920×1080	ParkScene	78.74%	85.35%
	Kimono1	52.13%	71.71%
	Cactus	84.63%	90.48%
	BQTerrace	66.02%	86.65%
	BasketballDrive	75.56%	86.91%
1280×720	FourPeople	95.49%	91.94%
	Johnny	89.63%	94.48%
	KristenAndSara	91.82%	91.94%
	vidyo1	93.36%	92.76%
	vidyo4	83.54%	92.37%
832×480	RaceHorses	9.90%	42.03%
	PartyScene	46.99%	64.80%
	BQMall	81.64%	78.64%
	BasketballDrill	81.37%	81.92%
	BasketballDrillText	81.50%	81.56%
均值		74.30%	82.59%

由表 3-8 可得，DP 的均值为 74.30%，表示一个视频序列中深度为 0 的 CTU 有 74.30%的概率可以被检测出来，且由于深度为 0 的 CTU 在视频序列中占有很大比重，所以可以节约大量编码时间。深度 0 检测算法的另一个评价标准为 DA，即被判定为深度 0 的 CTU 中有超过 80%是正确的，这在一定程度上保证了该算法在节约编码时间的同时不会过多提高码率。表中 RaceHorses 测试序列所对应的 DP 极低，只有 9.90%，这主要是由于该视频中深度为 0 的 CTU 本身概率就

非常低，由表 3-2 可知，只有 9.3%。而深度为 0 检测算法的条件要求三个参考 CTU 中至少有两个深度为 0，所以在深度 0 极少的视频中，这种检测条件就非常难以达到，也就造成了 DP 非常低的情况。对于包含深度为 0 的 CTU 较少的视频，深度 0 检测算法在编码时间节约上效果并不明显，因为只有很少的 CTU 避免了遍历[1, 3]，而接下来的快速划分算法可以弥补这一缺点。

3.3.2　相似区域和非相似区域的编码单元快速划分算法

相似区域和非相似区域通过 BSAD 与 Threshold 的相对大小来进行判断，若 BSAD 小于 Threshold，表明当前 CTU 和参考 CTU 较为相似，被称为相似区域，否则为非相似区域。针对相似区域和非相似区域的特点，设计了不同的 CU 快速划分算法。

相似区域 CU 快速划分算法：

（1）计算当前 CTU 的 BSAD 值。

（2）将 CU_CO、CU_L 和 CU_U 的最大划分深度按照升序进行排序{d1，d2，d3}。

（3）若 BSAD 小于阈值 Threshold，则当前 CTU 遍历深度范围设定为[0,d2]。

深度为 0 的 CU 快速检测算法是相似区域 CU 快速划分算法的一种特殊情况，当 d2 等于 0 时，就是深度 0 检测算法。

非相似区域 CU 快速划分算法：

AD（Average Depth，平均深度）的定义如式 3-8。其中 ω_1、ω_2、ω_3 分别代表 CU_CO、CU_L、CU_U 的权重，由表 3-7 可知，三者与当前 CTU 的深度关系基本相同，所以权重相同，都为三分之一。d_1、d_2、d_3 代表 CU_CO、CU_L、CU_U 的最大划分深度。

$$AD = \omega_1 d_1 + \omega_2 d_2 + \omega_3 d_3 \tag{3-8}$$

具体算法流程如下：

（1）若 $0.0 \le AD < 0.5$，当前 CTU 遍历深度范围设定为[0, 1]。

（2）若 $0.5 \le AD < 1.5$，当前 CTU 遍历深度范围设定为[0, 2]。

（3）若 $1.5 \le AD \le 3.0$，当前 CTU 遍历深度范围设定为[1, 3]。

为了检测本小节提出算法的准确性，以 HEVC 标准划分流程得到的深度作

为对照组，将利用该算法得到的划分深度作为实验组，用 HR（Hit Rate，击中率）作为评价指标，HR 的定义如式 3-9。该实验仅仅统计可以进入算法流程的 CTU 的击中率，不包含测试帧、前两帧、每一帧的第一行和第一列的 CTU。例如，第一行的 CTU 是不包含 CU_U 这个参考 CTU 的。实验结果如表 3-9 所示，实验参数和表 3-1 完全相同。

$$HR = \frac{the\ correct\ number\ of\ the\ CU\ depth\ estimation}{the\ total\ number\ of\ the\ CU\ depth\ estimation} \times 100\% \qquad (3\text{-}9)$$

表 3-9　CU 快速划分算法预测准确性

分辨率	序列名	相似区域	非相似区域		
			［0, 1］	［0, 2］	［1, 3］
2560×1600	PeopleOnStreet	90.84%	85.66%	93.64%	99.59%
	Traffic	89.84%	89.94%	96.90%	89.71%
1920×1080	ParkScene	88.42%	86.87%	96.23%	91.87%
	Kimono1	83.20%	95.78%	99.39%	87.56%
	Cactus	91.69%	95.96%	98.27%	97.04%
	BQTerrace	87.81%	97.78%	97.45%	83.92%
	BasketballDrive	88.98%	97.28%	98.29%	94.78%
1280×720	FourPeople	92.07%	80.17%	94.63%	94.56%
	Johnny	94.56%	96.78%	98.20%	92.47%
	KristenAndSara	92.22%	91.61%	97.44%	89.26%
	vidyo1	92.86%	92.80%	97.98%	94.72%
	vidyo4	92.45%	97.05%	98.36%	89.58%
832×480	RaceHorses	94.04%	100.00%	94.09%	97.17%
	PartyScene	87.56%	95.24%	96.45%	98.82%
	BQMall	84.95%	90.68%	84.69%	97.85%
	BasketballDrill	86.45%	75.00%	90.42%	99.06%
	BasketballDrillText	86.09%	73.07%	90.44%	99.26%
	均值	89.65%	90.68%	95.46%	93.95%

由表 3-9 可知，相似区域和非相似区域 CU 快速划分算法的击中率几乎都在 90%左右，这说明本小节提出的算法可以较好的跳过不必要的搜索深度，达到降

低编码时间的目的。

3.3.3　整体算法流程

本章提出的算法的流程图如图 3-5 所示：

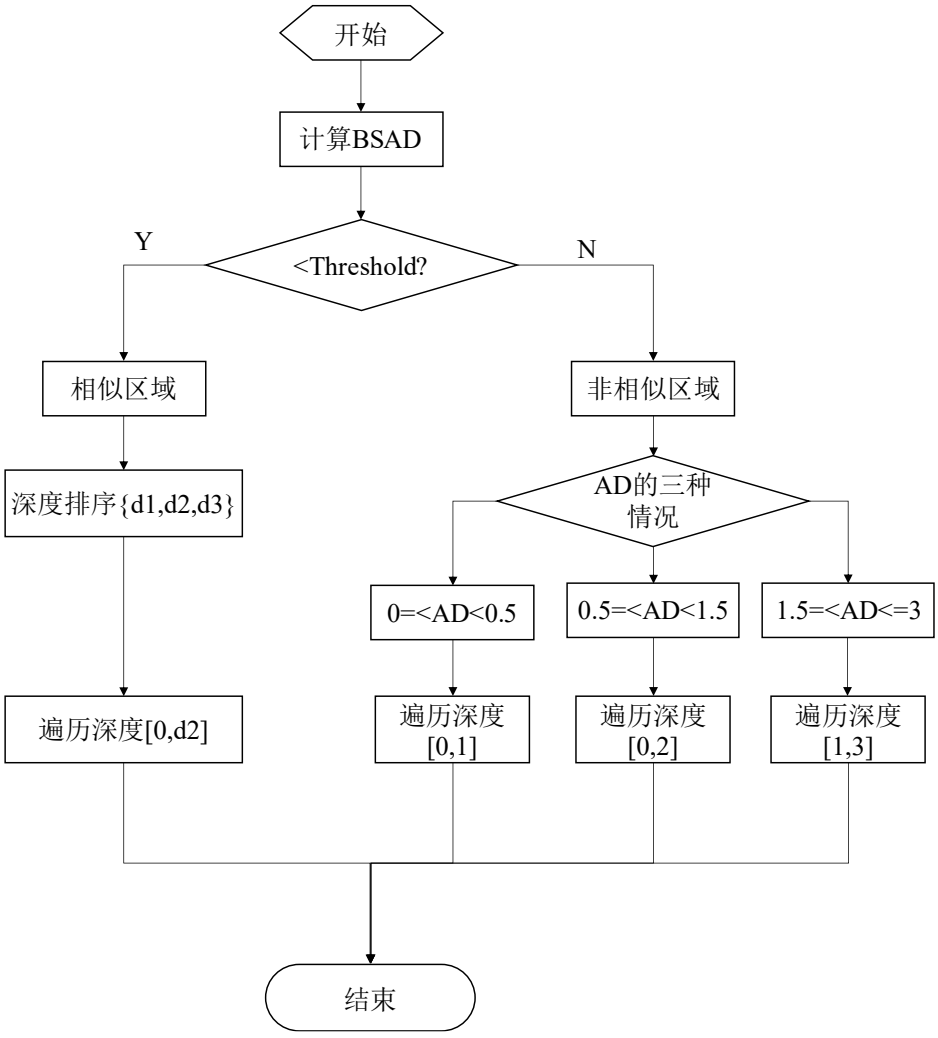


图 3-5　算法流程图

具体的处理流程如下：

（1）将输入的待编码的视频帧划分为同等大小的 CTU，计算当前待编码的 CTU 的 BSAD 值。

（2）若 BSAD 值小于阈值 Threshold，则当前 CTU 被判定为相似区域，后续处理执行相似区域编码单元快速划分算法，算法细节见 3.3.2 节。

（3）若 BSAD 值大于或者等于阈值 Threshold，则当前 CTU 被判定为非相似区域，后续处理执行非相似区域编码单元快速划分算法，算法细节见 3.3.2 节。

（4）进入（1），继续下一个 CTU 的划分过程。

3.4　实验结果与分析

3.4.1　编码单元快速划分算法复杂度定量分析

本文在 3.1.2 节中从 4 个方面分析了帧间预测编码单元划分计算复杂度较高的原因，第一个就是要完成一个 CTU 的划分需要遍历 85 个编码单元，而其余三个方面的操作是建立在每一个编码单元之上的。因此，本小节通过比较使用 HEVC 标准流程划分视频序列时遍历的编码单元的数量与使用本文提出的快速划分算法划分视频序列时遍历的编码单元的数量来直观显示本文所提算法对计算量的降低效果。

表 3-12 是表 3-10 和表 3-11 的配置参数表。

表 3-10 表示的是按照 HEVC 标准流程划分视频序列时需要遍历的不同深度的编码单元的数量。该数量只与视频的分辨率和编码帧数相关。

表 3-11 表示的是使用本文提出的快速划分算法划分视频序列时需要遍历的不同深度的编码单元的数量。

表 3-10　HEVC 标准流程

分辨率	depth0	depth1	depth2	depth3
2560×1600	50000	200000	800000	3200000
1920×1080	25500	102000	408000	1632000
1280×720	12000	48000	192000	768000
832×480	5200	20800	83200	332800

表 3-11　本文所提快速划分算法

分辨率	序列名	depth0	depth1	depth2	depth3
2560×1600	PeopleOnStreet	23629	173248	691904	2711104
	Traffic	42564	99848	390512	1335168
1920×1080	ParkScene	21835	61328	242144	888384

	Kimono1	23377	75552	282880	873664
	Cactus	21681	57636	224016	818752
	BQTerrace	24020	57432	184064	569920
	BasketballDrive	22106	59384	215232	718400
1280×720	FourPeople	11577	17576	69840	264512
	Johnny	11854	18748	66176	227648
	KristenAndSara	11730	18212	68128	240960
	vidyo1	11697	18056	69392	250688
	vidyo4	11713	22080	77648	256064
832×480	RaceHorses	3188	20524	82016	322176
	PartyScene	4010	18584	74000	289344
	BQMall	3899	15800	63200	244672
	BasketballDrill	4140	14864	59184	227008
	BasketballDrillText	4123	14900	59392	227776

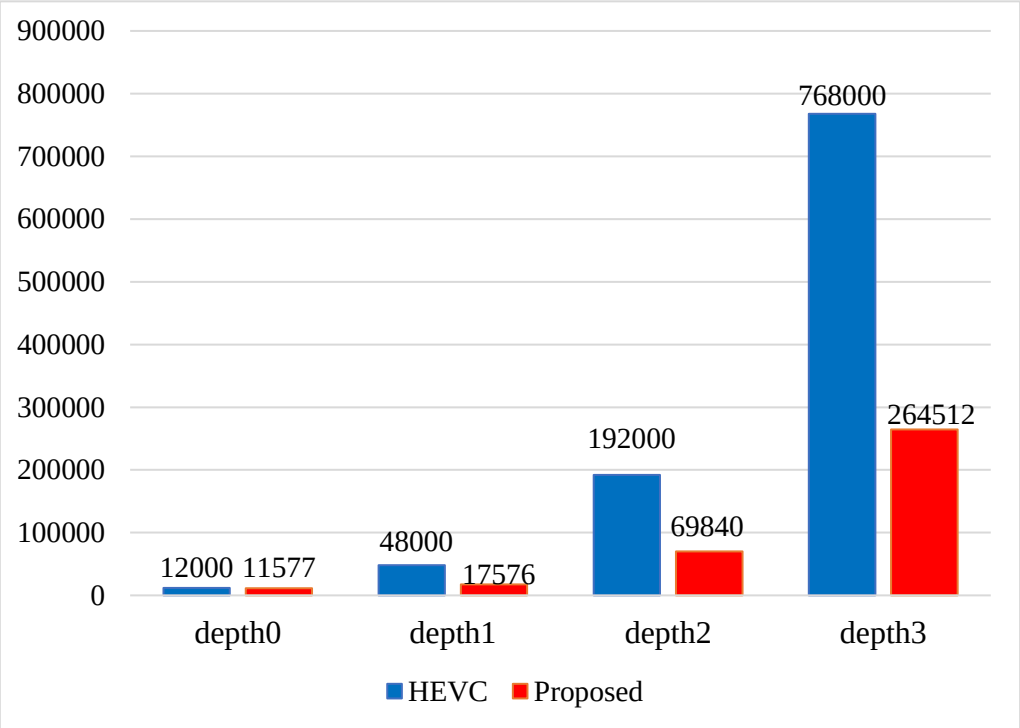


图 3-6　FourPeople 不同深度 CU 遍历数量对比

为了更加直观地展示 HEVC 标准划分流程和本文所提算法在编码同一视频

时遍历的 CU 的数量不同，以测试序列 FourPeople 为例，利用柱形图来展示该对比。由图 3-6 可得，对于深度为 1、2、3 的 CU，HEVC 标准流程遍历的数量要远多于本文所提算法。即本文所提算法主要通过减少遍历深度为 1、2、3 的 CU 来提高测试序列 FourPeople 的编码速度。

3.4.2　率失真性能与时间节约分析

为了检测提出的算法的综合效果，表 3-13 利用 BDBR（Bjontegaard Delta Bit Rate）、BDPSNR（Bjontegaard Delta Peak Signal to Noise Ratio）、ETR（Encoding Time Reducing，编码时间减少）和 HR 四个指标来进行衡量。其中 BDBR 表示和 HEVC 标准算法相比，在相同的重建视频质量下，本章提出的算法所造成的码率提高百分比。BDPSNR 表示和 HEVC 标准算法相比，在相同的压缩码率下，重建视频的峰值信噪比的变化情况。ETR 表示针对同一视频序列，编码相同帧数时本章提出的算法相比于 HEVC 标准算法节约的时间百分比，其定义如式 3-10。HR 定义如式 3-9 所示。

$$ETR=\frac{1}{4}\sum_{QP=22,27,32,37}\frac{ET_{reference}^{QP}-ET_{proposed}^{QP}}{ET_{reference}^{QP}}\times 100\% \qquad (3\text{-}10)$$

式 3-10 中，ETR 表示量化参数 QP 取四个不同值时的平均编码节约时间，当 QP 取不同的值时，量化步长是不同的，对应着不同的码率，其 CTU 划分结果也是不同的。$ET_{reference}^{QP}$ 表示按照 HEVC 标准算法编码视频花费的时间，$ET_{proposed}^{QP}$ 表示将本章提出的算法嵌入 HEVC 中后编码视频花费的时间。表 3-12 是表 3-13 的参数表，其中介绍了实验参数和设备参数，其余实验参数为 lowdelay_main 配置文件默认参数。

由表 3-13 得，嵌入本章提出的算法的 HEVC 对 CTU 的划分结果与原 HEVC 标准流程划分结果超过 90%是相同的，表明了提出的算法的预测准确性。由于预测较为准确，本章算法在平均节约编码时间 29.40%的情况下，编码比特率仅提高 1.30%，重建视频的峰值信噪比下降 0.06dB。

通过分析得，节约时间较多的视频序列，如 Traffic、BQTerrace 和分辨率为 1280×720 的视频序列，这些测试序列中深度为 0 的 CU 所占的比例都非常高，表明本章提出的深度为 0 的 CU 快速检测算法效果较好，可以检测出其中较高比

例的深度为 0 的 CTU，从而避免遍历深度[1,3]，仅仅遍历深度 0 则可以节约 75%的时间。RaceHorses 是 17 个视频中时间节约百分比最低的，只有 7.89%，主要是由两个原因造成的。一是其中含有的深度为 0 的 CTU 较少，并且由于深度 0 检测算法条件较高，导致其检测效率较低，这也是 PeopleOnStreet 和 PartyScene 的 ETR 较低的原因。二是由于视频分辨率较小，每一帧含有的 CTU 数目较少，而其中的第一行以及第一列是按照 HEVC 标准流程划分的，并不会节约任何时间，这些CTU数目占据了较大一部分比例，导致真正进入本章所提出算法的CTU的比例相对其他分辨率视频较少。并且进入非相似区域以及相似区域中非零的那些 CTU 需要遍历的深度仍然较多，时间节约效果没有仅仅遍历深度 0 那么显著。这个也是其他 832×480 分辨率视频 ETR 较低的原因。

表 3-12　实验参数和设备参数

参数	
CPU	Intel(R) Core(TM) i5-6200U CPU@2.30GHz
内存	8GB
操作系统	Windows 10 专业版
编译环境	Visual Studio 2017
HM 版本	HM16.7
量化参数 QP	22、27、32、37
编码帧数	50

表 3-13　实验结果

分辨率	序列名	HR(%)	BDBR(%)	BDPSNR(dB)	ETR(%)
2560×1600	PeopleOnStreet	96.03	0.59	-0.05	17.25
	Traffic	90.38	1.37	-0.10	33.97
1920×1080	ParkScene	89.48	1.06	-0.04	28.62
	Kimono1	87.49	1.57	-0.06	24.66
	Cactus	93.13	1.29	-0.05	31.81
	BQTerrace	90.10	2.16	-0.06	34.59
	BasketballDrive	91.47	1.45	-0.06	32.13
1280×720	FourPeople	92.21	1.02	-0.04	42.41
	Johnny	94.89	1.19	-0.03	43.10

	KristenAndSara	92.40	1.15	-0.09	41.84
	vidyo1	93.15	1.49	-0.05	42.49
	vidyo4	93.25	1.22	-0.11	41.12
832×480	RaceHorses	95.87	0.73	-0.03	7.89
	PartyScene	91.75	1.31	-0.07	12.42
	BQMall	89.80	1.92	-0.08	21.65
	BasketballDrill	90.44	1.14	-0.05	22.50
	BasketballDrillText	90.33	1.42	-0.04	21.39
	均值	91.89	1.30	-0.06	29.40

表 3-14　本文所提算法与文献[12]、[22]中算法性能对比

序列名	Proposed method			Ref.[12]			Ref.[22]		
	BDBR (%)	BDPSNR (dB)	ETR (%)	BDBR (%)	BDPSNR (dB)	ETR (%)	BDBR (%)	BDPSNR (dB)	ETR (%)
Cactus	1.29	-0.05	31.81	0.60	-0.01	22.62	1.35	-0.04	30.25
Kimono	1.57	-0.06	24.66	0.07	-0.01	21.57	0.42	-0.01	25.17
ParkScene	1.06	-0.04	28.62	0.52	-0.02	21.01	1.28	-0.03	26.46
BQTerrace	2.16	-0.06	34.59	0.34	-0.01	22.97	1.23	-0.03	29.68
FourPeople	1.02	-0.04	42.41	0.34	-0.01	23.54	0.54	-0.01	36.61
Johnny	1.19	-0.03	43.10	0.37	-0.01	23.95	0.96	-0.01	29.37
BasketballDrill	1.14	-0.05	22.50	1.21	-0.04	20.84	2.31	-0.16	33.58
BQMall	1.92	-0.08	21.65	1.01	-0.04	21.30	1.56	-0.07	27.62
PartyScene	1.31	-0.07	12.42	0.78	-0.03	21.91	0.76	-0.05	22.79
均值	1.41	-0.05	29.08	0.58	-0.02	22.19	1.16	-0.05	29.05

表 3-14 将本文所提算法与文献[12]、文献[22]中的算法的性能进行了比较。其中文献[12]利用了 CTU 划分之间的时空相关性来修剪当前 CTU 及其子 CU 的遍历深度范围。文献[22]则利用了 QP 的空间相关性来加速 CTU 划分过程。本文所提算法联合使用了 CTU 划分的时空相关性以及参数 BSAD。由表 3-14 得，与文献[12]相比，本文所提算法对编码速度的提升更多，但率失真性能较差一些。与文献[22]相比，两者对编码速度的提升以及率失真性能基本相同。

综上所述，本章提出的算法对具有较多静态背景和内容变化较缓慢的视频有着更好的时间节约效果，而对于含有较多动态内容的视频编码时间节约效果较差一些。该算法的提出有助于推动 HEVC 在卫星军事侦察、赛事直播等对实时性要求较高的卫星视频业务中的实际应用。

3.5　本章小结

本章首先分析 CTU 的 BSAD 与其深度是否为 0 的关系，发现当 BSAD 较小时，当前 CTU 深度更加倾向于 0。之后为了进一步提高深度预测的准确性，结合 CTU 深度划分的时空相关性提出了深度为 0 的 CU 快速检测算法以及相似区域和非相似区域的 CU 快速划分算法。实验结果表明，本章提出的算法能够较好地检测出视频中深度为 0 的 CTU，平均提高 29.40%的编码速度，而码率仅仅增加 1.30%。

第 4 章　基于视觉显著性检测的 LCU 层码率控制算法

视频压缩后的码率和视频的失真是相互制约的关系，即基于同一种视频压缩标准，压缩后的码率越大，视频的失真就越小；压缩后的码率越小，视频的失真就越大。码率控制算法的首要目标是使压缩后的码率尽可能等于目标码率或者给定的带宽，在此基础上使视频的失真达到最小。

HEVC 标准测试软件 HM10.1 中集成了 λ 域码率控制算法[53]，该算法在 LCU 层包含三个模块，目标比特分配、λ 和 QP 估计以及模型参数更新，本章对目标比特分配和模型参数更新两个模块做了些许改进。

4.1　率失真优化与码率控制技术

率失真优化技术在 HEVC 中是一个非常重要的工具，它是 HEVC 在选择各种编码参数时的判断依据，比如变换单元（Transform Unit, TU）的划分模式选择、PU 的划分模式选择、最佳匹配块的确定、帧内预测中预测模式的选择以及 CTU 的四叉树划分等等。同码率控制模块相同，率失真优化技术也不属于视频编码标准的范畴，编码器可以使用不同的率失真优化算法。HEVC 中使用的率失真优化技术表现为拉格朗日率失真代价公式，如式 4-1。如在选择预测模式时，不同的预测模式会得到不同的 J 值，选择 J 值最小的预测模式做为最优预测模式。其中 λ 是一个和量化参数 QP 相关的数值。

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

视频编码过程中若量化参数 QP 确定，则可以通过率失真优化技术找到该 QP 值对应的一个 (R, D) 点，其中 R 表示码率，D 表示失真。该点具有的特点是在码率不高于 R 的条件下，D 为最小失真。而码率控制算法的目标是在码率受限的条件下使视频整体失真最小，因此码率控制问题可以看做是一个率失真优化问题。即在给定的码率下，通过确定量化参数 QP 和 λ 值，然后利用拉格朗日率失真代价公式选择使失真最小的编码参数。

4.1.1　视频编码率失真曲线

离散信源率失真函数 $R(D)$ 的一般曲线如图 4-1 所示。由图可知，当率失真曲线已知时，对于给定的失真限制 D^*，在率失真曲线上可以直接得到最小速率 $R(D^*)$。同理，对于给定的速率限定 $R(D^*)$，则能够得到的最小失真为 D^*。

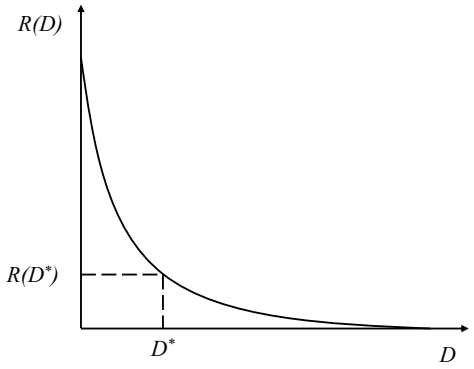


图 4-1　$R(D)$ 的典型曲线

实际的编码系统中，$R-D$ 平面中包含了许多离散点，每一个离散点都代表了码率和失真的一种对应关系。离散点的数量由编码标准中的参数数量决定，假设编码参数有 N 个，第 i 个参数有 x_i 种取值，则离散点的数量可以由式 4-2 计算。这种计算方式认为各个参数是相互独立的，但在编码过程中，前一个参数的选择有时候会影响到下一个参数的取值范围。在 HEVC 编码标准中，参数有很多个，如量化参数、帧内预测的预测模式、帧间预测的 PU 划分模式、TU 划分模式、运动矢量 MV 等等。其中量化参数有 52 个取值，帧内预测有 35 种预测模式，包括 33 种角度预测、DC 模式和 Planar 模式。

$$Sum=\prod_{i=1}^{i=N}x_i \tag{4-2}$$

使用一组特定的编码参数对视频进行编码，就可以获得该编码参数条件下的编码速率和失真，这组编码参数对应的离散点 (R,D) 称为视频编码实际率失真曲线的一个可操作点。遍历所有可行的编码参数组合就可以得到所有可操作点，由于实际编码系统的参数取值都是有限的，所以可操作点数量也是有限的。

对于给定的速率限制 R^*，可以找到一个失真最小的可操作点。遍历所有的

R^*，就可以得到一组可操作点。将这些可操作点连接起来，得到的曲线称为可操作率失真曲线，如图 4-2 所示。可操作率失真曲线是 HEVC 针对当前视频源能够达到的最佳率失真性能，在选择编码参数时，使其码率失真性能尽可能接近可操作率失真曲线。

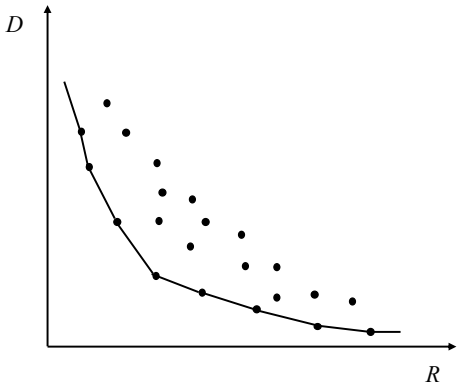


图 4-2　可操作率失真曲线

目前常用的视频编码率失真曲线模型有指数模型和双曲模型，指数模型的计算方式是对 D 和 R 进行指数建模，如式 4-3 所示。其中 α 和 β 是与视频源相关的参数。

$$D = \alpha e^{-\beta R} \tag{4-3}$$

双曲模型是对 D 和 R 进行双曲建模，如式 4-4，其中 C 和 K 是与视频源相关的参数。

$$D(R) = CR^{-K} \tag{4-4}$$

文献[53]已经证明双曲模型能够更好地描述编码视频的码率和失真之间的关系，如图 4-3 所示。尽管这两个视频的 C 和 K 差距较大，但是双曲模型对 D 和 R 的拟合效果非常好。

其中 bpp 和 MSE 的定义如式 4-5 和式 4-6。

$$bpp = \frac{R}{f \cdot w \cdot h} \tag{4-5}$$

$$MSE = \frac{1}{N}\sum_{i}(rec_i - org_i)^2 \tag{4-6}$$

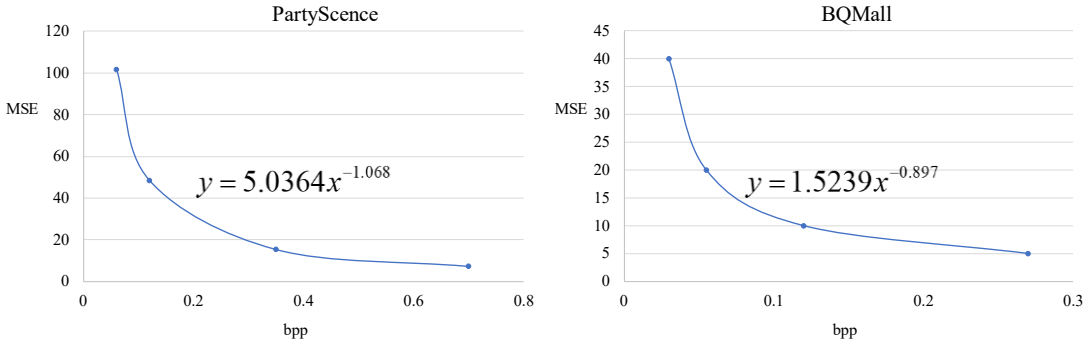


图 4-3　双曲模型 $R-D$ 拟合曲线

R 表示码率，f 表示帧率，单位为帧/秒，w 表示图像宽，h 为图像高度，bpp（bit per pixel）表示传输每个像素需要的比特数。MSE 表示均方误差，rec_i 表示第 i 个像素的重建值，org_i 表示第 i 个像素的原始值，N 表示像素总数。

4.1.2　λ 域码率控制算法

码率控制的基本流程如图 4-4 所示。

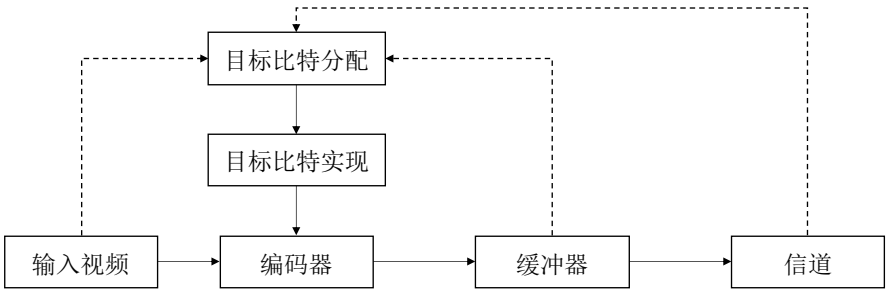


图 4-4　码率控制基本流程

缓冲器的主要作用是为了缓解编码器输出速率与信道带宽不匹配的矛盾，产生该矛盾的原因是编码器的实际输出有时并不等于分配的目标比特数，当实际输出码率大于信道带宽时，多余的数据量就会存储在缓冲器中。此外，缓冲器中数据状态也影响着接下来编码单元的目标比特分配。

码率控制算法可以分为两步，第一是目标比特的分配，不同的码率控制算法在目标比特的分配上是相似的，都在三个级别上进行实现，分别为 GOP 级别、

图片级别和 LCU 级别，具体的分配方式各有不同。第二个是目标比特实现，即在已知该编码单元的目标比特后，为编码单元选择合适的编码参数。评价码率控制算法性能的标准有两个，一是该编码单元实际编码比特数与目标比特数是否接近，二是选择的编码参数是否能使编码单元失真达到最小。

针对 HEVC 的码率控制算法有 $R-Q$ 模型和 $R-\lambda$ 模型。$R-Q$ 模型利用编码比特数和量化参数之间的关系，也称为 Q 域码率控制算法，该算法假定量化参数是决定比特率的关键因素，但是当编码参数比较灵活时，该关系已不能较好的拟合。文献[53]以拉格朗日因子 λ 为纽带建立了 $R-\lambda$ 与 $\lambda-Q$ 模型用于确定编码单元的量化参数，并于 JCT-VC 第 11 次会议中提出了 K0103 速率控制提案。与 $R-Q$ 模型相比，K0103 提案码率控制更为精确，比特波动更小，因此该提案加入了 HEVC 测试平台 HM 中，也是本章所研究的码率控制算法。

4.1.2.1　目标比特分配

目标比特分配采用分级策略（GOP 级、图像级、LCU 级）依次为不同编码单元分配目标比特。

（1）GOP 级目标比特分配

视频序列在编码时通常被划分为多个连续的 GOP，GOP 级目标比特分配是指根据信道速率和缓冲区状态为每个 GOP 分配目标比特数，如式 4-8。

$$R_{PicAvg}=\frac{R_{tar}}{F_r} \tag{4-7}$$

$$T_{GOP}=\frac{R_{PicAvg}\times(N_{coded}+SW)-R_{coded}}{SW}\times N_{GOP} \tag{4-8}$$

其中，R_{tar} 表示目标比特率，F_r 表示帧率，R_{PicAvg} 表示每一帧分配到的比特数，T_{GOP} 表示当前 GOP 的目标比特数，N_{coded} 表示已经编码的帧数，R_{coded} 表示编码 N_{coded} 帧所实际产生的比特数，SW 表示滑动窗口的尺寸，目的是平滑比特波动，其值一般为 40。例如，当已编码的 GOP 产生的比特数大于其分配的比特数，之后的 GOP 的目标比特数就会更小。

（2）图像级目标比特分配

一个 GOP 中包含多个图像，图像级目标比特分配是指根据该 GOP 的总目

标比特数为每幅图像分配目标比特数，分配方式如式 4-9。

$$T_{Pic}=\frac{T_{GOP}-Coded_{GOP}}{\sum_{\{AllNotCodedPicture\}}\omega_{Pic}}\times\omega_{PicCurr} \tag{4-9}$$

其中，T_{Pic} 表示当前图像分配的目标比特数，T_{GOP} 表示当前图像所属于的 GOP 分配的目标比特数，$Coded_{GOP}$ 表示当前 GOP 中已经消耗掉的比特，w_{Pic} 表示分配目标比特时图像对应的权重。该权重有两种设置方式，一种是 GOP 中所有的图像具有相同的权重，即剩余比特数平均分配给剩余图像。另一种是根据图像在 GOP 中的不同位置赋予其不同的权重，如表 4-1。

表 4-1　低延迟结构的 w_{Pic}

编码顺序	播放顺序	分配权重			
		$bpp>0.2$	$0.2\geq bpp>0.1$	$0.1\geq bpp>0.05$	其他
1	1	2	2	2	2
2	2	3	3	3	3
3	3	2	2	2	2
4	4	6	10	12	14

（3）LCU 级别目标比特分配

LCU 级别目标比特分配是指根据当前图像的目标比特数为该图像内每个 LCU 分配目标比特数，分配方式如式 4-10。

$$T_{LCU}=\frac{T_{Pic}-Bit_{H}-Coded_{Pic}}{\sum_{\{AllNotCodedLCU\}}\omega_{LCU}}\times\omega_{LCUCurr} \tag{4-10}$$

其中 Bit_H 是该图像头信息比特数的预测值，其值为视频序列中与当前图像处于相同层的所有已编码图像头信息所用实际比特的平均值，T_{Pic} 是当前图像的目标比特数，$Coded_{Pic}$ 表示当前图像已经使用的比特数，ω_{LCU} 表示每个 LCU 的权重，计算公式如式 4-12。

$$MAD_{LCU} = \frac{1}{N_{pixels}} \sum_{\{AllPixelsInLCU\}} \left| P_{org} - P_{pred} \right| \tag{4-11}$$

$$\omega_{LCU} = MAD_{LCU}^2 \tag{4-12}$$

式 4-11 中用于计算的 LCU 为与当前图像处于相同层且距离最近的图像对应位置的 LCU，即通过时域预测得到当前 LCU 的权重。

4.1.2.2　量化参数确定与模型参数更新

当目标比特分配完成后，接下来就是要确定各个编码参数，以满足编码单元失真最小。该问题可以抽象为一个约束性优化问题，如式 4-13。

$$\{Para\}_{opt} = \arg \min_{\{Para\}} D \qquad s.t.\ R \le R_t \tag{4-13}$$

其中 $para$ 是编码参数集，包括 QP、模式、运动矢量等，R_t 表示分配的目标比特数。约束性问题可以通过拉格朗日乘子法转换为非约束问题，如式 4-14。

$$\{Para\}_{opt} = \arg \min_{\{Para\}} (D + \lambda R) \tag{4-14}$$

视频编码率失真曲线可以使用双曲建模，如式 4-4，即对于任意的 R_t，最小率失真为 $D(R_t)$，可操作点 $(R_t, D(R_t))$ 对应的编码参数集就是最优的编码参数集。当将问题转化为非约束性问题时，R_t 和拉格朗日因子 λ 是一一对应的关系，如式 4-15。其中 α 和 β 是和视频源相关的参数。此外，实验表明，量化参数 QP 和 $\ln\lambda$ 之间存在比较好的线性关系，如式 4-16。

$$\lambda = -\frac{\partial D}{\partial R} = CK \cdot R^{-K-1} \overset{\Delta}{=} \alpha R^{\beta} \tag{4-15}$$

$$QP = 4.2005 \ln \lambda + 13.7122 \tag{4-16}$$

当给定目标比特 R_t 时，通过式 4-15 和式 4-16 可以得到相应的 λ 和 QP, 然后利用拉格朗日率失真代价公式可以确定其余的编码参数。

式 4-15 中，α 和 β 是和视频源相关的参数，当处理某一视频数据时，两者

的值是未知的，文献[53]经过大量实验给出了两者的初始值，分别为 3.2003 和-1.367。在视频编码的过程中，α 和 β 的值随着视频内容的变化不断在更新，模型参数更新发生在两个层级上，分别为图像级和 LCU 级，两者更新方式基本相同。更新方式如下所示。

$$\lambda_{comp} = \alpha_{old} bpp_{real}^{\beta_{old}} \tag{4-17}$$

$$\alpha_{new} = \alpha_{old} + \delta_{\alpha} \times (\ln \lambda_{real} - \ln \lambda_{comp}) \times \alpha_{old} \tag{4-18}$$

$$\beta_{new} = \beta_{old} + \delta_{\beta} \times (\ln \lambda_{real} - \ln \lambda_{comp}) \times \ln bpp_{real} \tag{4-19}$$

其中 α_{old} 和 β_{old} 为当前 LCU 在计算 λ 值时使用的参数值，bpp_{real} 为当前 LCU 编码后产生的实际比特数，λ_{real} 表示当前 LCU 编码时使用的拉格朗日因子，α_{new} 和 β_{new} 为更新之后的参数值，δ_{α} 和 δ_{β} 的值分别设置为 0.1 和 0.05。在使用式 4-15 计算 λ 时，其使用的 α 和 β 分别为与当前图像处于相同层且距离最近的图像对应位置 LCU 的 α 和 β 的更新值。

此外，为了保持编码视频质量不会有剧烈波动，λ 和 QP 都不应该发生显著变化，因此在图片级和 LCU 级都对 λ 和 QP 的值进行了限制，如下式。

$$\lambda_{lastLCU} \cdot 2^{\frac{-1}{3}} \le \lambda_{curLCU} \le \lambda_{lastLCU} \cdot 2^{\frac{1}{3}} \tag{4-20}$$

$$\lambda_{curPic} \cdot 2^{\frac{-2}{3}} \le \lambda_{curLCU} \le \lambda_{curPic} \cdot 2^{\frac{2}{3}} \tag{4-21}$$

$$QP_{lastLCU} - 1 \le QP_{curLCU} \le QP_{lastLCU} + 1 \tag{4-22}$$

$$QP_{curPic} - 2 \le QP_{curLCU} \le QP_{curPic} + 2 \tag{4-23}$$

其中 λ_{curLCU} 表示当前 LCU 的拉格朗日因子，$\lambda_{LastLCU}$ 表示和当前 LCU 相邻的前一个 LCU 的拉格朗日因子，λ_{curPic} 表示当前 LCU 所属的图像对应的拉格朗日因子。此外，当前 LCU 的 QP 值与相邻 LCU 的 QP 值相差不能大于 1，与其所属的图像的 QP 值相差不能大于 2。

4.2　图像的视觉显著性检测算法

λ域码率控制算法的核心是为编码单元分配目标比特之后，根据$R-\lambda$模型和$\lambda-QP$模型进而确定使当前编码单元失真最小的其他编码参数，所以更加合理的目标比特分配可以提高码率控制算法的性能。

λ域码率控制算法的 LCU 层目标比特分配公式为式 4-10，待编码的 LCU 的目标比特分配权重取决于与当前图像处于相同层且距离最近的图像对应位置的 LCU 的平均预测残差绝对值之和的平方，即通过时域预测得到当前 LCU 的权重。若预测残差的绝对值之和越大，则经过二维离散余弦变换、量化以及熵编码之后得到的比特数就趋向于越大，因此该目标比特分配权重的计算具有合理性。但是该权重的计算仅仅利用了时域上的相关性，并没有考虑当前编码单元的内容特征，此外，该权重的计算仅利用了预测残差绝对值之和与需要的编码比特数之间的简单的正相关关系，可能会导致目标比特分配的不合理。

当人眼观察到一幅图像时，人类视觉系统（Human Visual System，HVS）会将人的目光迅速聚焦在图像中特定的对象上，若图像中含有多个对象，则目光聚焦也会有一定的先后顺序。这些对象是人类较为感兴趣的区域，HVS 的这种快速搜索图片或者视频中感兴趣区域的能力被叫做视觉显著性检测。这种能力可以帮助人类在海量的信息中迅速找到自己需要的信息，当人眼聚焦在图像中感兴趣的区域时，会对其余部分进行类似低通滤波的处理，使其模糊化，尤其是观看动态视频时，人的注意力会更加集中在显著性较高的对象上，而模糊化处理其余部分。若可以判断图像中每个 LCU 的视觉显著性高低，为视觉显著性较高的 LCU 分配更多的目标比特，而显著性较低的 LCU 分配更少的目标比特，那么得到的解码视频将会有更好的主观质量。

4.2.1　典型的视觉显著性检测算法结果与分析

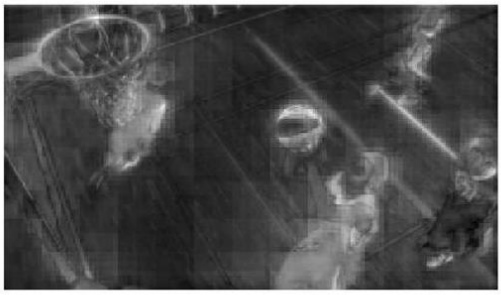
视觉显著性检测算法模拟 HVS 的工作机制，将一帧图像中各个区域的显著性高低以具体的数值描述，得到一张与原图像相同大小的显著性图。视觉显著性检测算法可以分为两个分支，一个是基于图像底层特征的自底向上的计算方式，另一个是通过特定样本训练，目标驱动的自上而下的计算方式。由于要处理的视频序列内容是随机的，所以在本章中仅仅选取了几种利用图像底层特征而设计的显著性检测算法，通过对一些图片的显著性图进行分析，从而确定不同图片中视觉显著性较高的区域具有的共同特征。

图像底层特征包括颜色、亮度、方向特征以及对比度等几个方面，基于这些

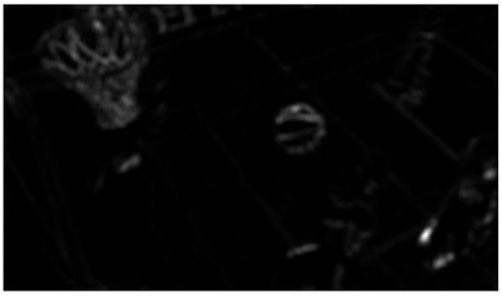
特征中的若干个而设计的典型的视觉显著性算法有 Itti 算法[68]、SR（Spectral Residual）算法[69]、FT（Frequency Tuned）算法[70]、HC（Histogram-Based Contrast）算法[71]、CA（Context Aware）算法[72]以及 GR（Graph Regularized）算法[73]。

图 4-5　BasketBall_Drill_55

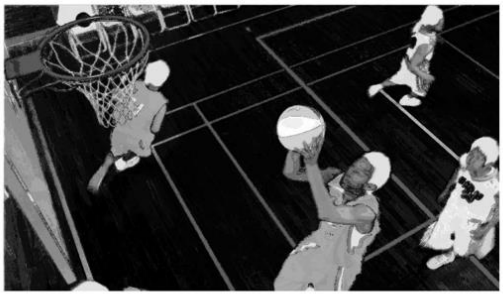
图 4-5 是 HEVC 标准测试序列 BasketballDrill_832x480_50 的第 55 帧，在图 4-6 中显示了其在不同的视觉显著性检测算法中的显著性图。



（a）Itti算法

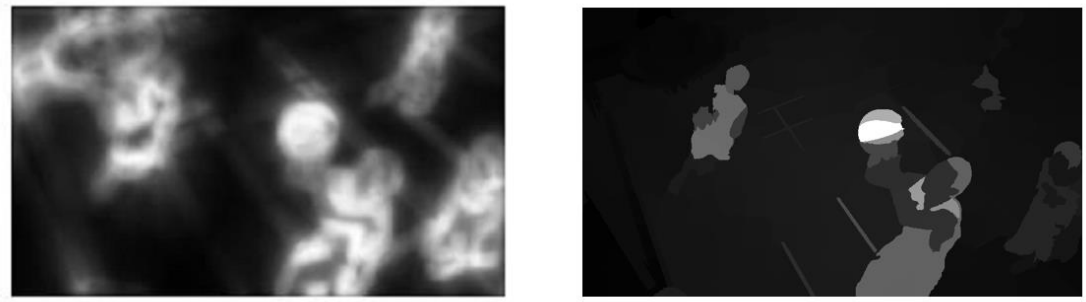


（b）SR算法



（c）HC算法

（d）FT算法



（e）CA算法　　（f）GR算法

图 4-6　典型的视觉显著性检测算法实验结果

在图 4-6 中，显著性图是和原图像大小相同的灰度图像，灰度值越高的部分显著性越高。对比不同算法的实验结果，可以发现视觉显著性较高的部分主要集中在和背景对比较为突出的各个对象上。

在不使用码率控制算法压缩视频时，不同帧以及相同帧中不同的 LCU 需要的编码比特数是不同的，当帧中含有较多运动速度较快的对象时，当前帧需要的编码比特数就更多。主要原因是帧中含有较多运动对象时，各个对象相对于背景的位置发生变化、各个对象之间可能会发生遮挡，这些都导致帧中的 LCU 无法找到较好的匹配块，残差信息就会变大，导致最后的编码比特数和语法元素增加。这表示编码处于运动对象中的 LCU 相对于背景区域的 LCU 需要更多的比特数，而这些运动对象的显著性由图 4-6 可知都是较高的，所以为视觉显著性较高的 LCU 分配更多的目标比特数不仅可以提高人类观察解码图像的主观感受，而且也符合无码率控制下不同 LCU 需要的编码比特数的相对关系。

4.2.2　低复杂度视觉显著性检测算法设计

上一小节分析了在为 LCU 分配目标比特时考虑视觉显著性的优势，而在较多的视觉显著性检测算法中选择一种合适的算法也是非常重要的。

表 4-2　实验参数

参数	
CPU	Intel(R) Core(TM) i5-6200U CPU@2.30GHz
测试平台	MATLAB R2021b

考虑到 HEVC 本身计算复杂度就非常高，所以对应的显著性检测算法应该

具有较低的计算复杂度。表 4-3 显示了以上六种算法计算图 4-5 的显著性图花费的时间，表 4-2 为表 4-3 的实验参数。

表 4-3　显著性检测算法计算时间

显著性算法	时间/s
Itti 算法	3.154
SR 算法	0.411
HC 算法	3.918
FT 算法	0.370
CA 算法	42.667
GR 算法	2.370

由表 4-3 可知，相较于其他算法，FT 算法在计算视觉显著性图像时花费的时间最少。考虑到 HEVC 本身就具有极高的计算复杂度，因此本章选择在 FT 算法的基础上设计计算复杂度更低的显著性检测算法，算法具体流程如下。

（1）输入视频序列为 YUV 格式，对 Y 通道图像进行高斯滤波，核大小为 3×3。

（2）对滤波后的 Y 通道图像求均值 $YAve$。

（3）求滤波后的 Y 通道图像中的每一个像素点与 $YAve$ 的差的平方 $DS(x,y)$，并记录下最大值 Dma

（4）利用（3）中的 Dma 归一化 $DS(x,y)$，得到显著性图。

上述流程中每一个值的具体计算公式如下。设输入的图像大小为 $M \times N$，$YO(x,y)$ 表示图像中坐标为 (x,y) 的像素的亮度值，对图像沿自身的边界进行镜像映射扩展，由于选择的高斯核大小为 3，这里扩展宽度为 1。高斯核如图 4-7 所示，对应的 σ 为 3。

0.1070	0.1131	0.1070
0.1131	0.1196	0.1131
0.1070	0.1131	0.1070

图 4-7　高斯核

设$YF(x,y)$表示滤波后坐标为(x,y)的像素的亮度值，其计算公式如式 4-24，其中$G(x,y)$表示高斯核中坐标为(x,y)的权值，设高斯核中心坐标为$(0,0)$。

$$YF(x,y)=\sum_{i=-1}^{1}\sum_{j=-1}^{1}G(i,j)\cdot YO(x+i,y+j) \tag{4-24}$$

均值$YAve$的计算公式如式 4-25。

$$YAve=\frac{1}{M\cdot N}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}YF(x,y) \tag{4-25}$$

$DS(x,y)$计算公式如式 4-26。

$$DS(x,y)=(YF(x,y)-YAve)^2 \tag{4-26}$$

最终得到的显著性图中坐标为(x,y)的像素的显著性值为$Sali(x,y)$，计算公式如式 4-27。

$$Sali(x,y)=\frac{DS(x,y)}{Dma} \tag{4-27}$$

为了验证简化后的视觉显著性检测算法的效果，图 4-8 中左边图片为 FT 算法生成的显著性图，右边图片为简化后的算法生成的显著性图。通过对比可以发现，简化算法生成的显著性图中显著性较高的对象与 FT 算法是相同的，这表明简化算法可以有效的找到图片中显著性较高的对象。两幅图片分别为 HEVC 官方测试序列 BasketballDrill_832x480_50 的第 55 帧和 BQMall_832x480_60 的第 1 帧。

FT算法

简化算法

FT算法　　简化算法

图 4-8　显著性对比图

4.3　基于视觉显著性检测的 LCU 层码率控制算法设计

LCU 层的码率控制算法由三个部分组成，目标比特分配、目标比特实现和 LCU 层模型参数更新。其中合理的目标比特分配不仅可以提高视频的主观质量，客观评价标准中的失真也会减小。

LCU 层目标比特分配公式为式 4-10，其中决定每个 LCU 分配目标比特数的参数为 LCU 对应的权重 ω_{LCU}，该权重仅利用了与当前帧处于同一级别的距离最近的帧中相同位置的 LCU 的预测残差的绝对值之和与实际编码比特数之间的正相关关系来预测当前 LCU 应当分配的目标比特数。这种分配方式有两个问题，一是没有考虑人眼对图像中不同部位具有不同的关注度，二是以这种方式进行目标比特分配时，不同 LCU 分配的目标比特数的比值与其对应的预测残差绝对值之和的平方的比值相同，而不同的 LCU 实际编码比特数的比值并不符合这种简单的对应关系。

针对以上的问题，4.3.1 节设计了一种新的 LCU 层目标比特分配权重计算方式。该权重的计算综合利用了 LCU 的视觉显著性以及预测残差经离散余弦变换（Discrete Cosine Transform, DCT）后得到的变换块绝对值之和。

4.3.1　LCU 层目标比特分配权重计算

由式 4-27 可以求得一帧图像的显著性图，对该显著性图求均值 $SAve$，表达式如式 4-28。

$$SAve = \frac{1}{M \cdot N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} Sali(x, y) \tag{4-28}$$

其中 M 和 N 分别为图像的高度和宽度，对每一个 LCU 求显著性均值，如式 4-29。

$$SAve_{LCU}=\frac{1}{m\cdot n}\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}Sali(x,y) \tag{4-29}$$

其中 m 和 n 分别为 LCU 的高度和宽度，最后将求得的 LCU 的显著性均值与一帧图像的显著性均值进行比较，其结果 ω_1 越大，表明当前 LCU 的显著性越高，如式 4-30。

$$\omega_1=\frac{SAve_{LCU}}{SAve} \tag{4-30}$$

HEVC 标准中对预测残差进行二维 DCT 变换，对变换系数进行量化，较小的变换系数将会被去除，最后进行熵编码，达到图像压缩的目的。因此，相对于预测残差，DCT 变换系数的大小能更好地反映编码当前 LCU 所需要的比特数的多少。

HEVC 标准使用的二维 DCT 变换公式如式 4-31。其中，N 表示残差矩阵的宽度和高度，$P(x,y)$ 表示残差矩阵中坐标为 (x,y) 的残差值，$X(k,l)$ 表示变换系数矩阵中坐标为 (k,l) 的系数值。由于随着二维 DCT 变换尺寸的增加，其计算复杂度大幅增加，因此本文将 LCU 划分为 8×8 的子单元。对每一个子单元单独进行二维 DCT 变换，经一系列处理后转换为目标比特分配权重的一部分，其具体的流程如下。

$$\begin{aligned}&X(k,l)=C(k)C(l)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}P(x,y)\cos[\frac{(2x+1)k\pi}{2N}]\cos[\frac{(2y+1)l\pi}{2N}]\\&k,l=0,1...,N-1\end{aligned} \tag{4-31}$$

$$C(k)=C(l)=\begin{cases}\sqrt{\frac{1}{N}}, & k,l=0\\ \sqrt{\frac{2}{N}}, & \text{其他}\end{cases}$$

对式 4-31 进行转化推导，可以利用推导结果式 4-32 来求 8×8 子单元的二维 DCT 变换。其中 Y 为 8×8 系数矩阵，$C_{8\times8}$ 为 DCT 变换矩阵，$C_{8\times8}^{T}$ 为 $C_{8\times8}$ 的转置

矩阵，X 为预测残差矩阵。

$$\boldsymbol{Y}=\boldsymbol{C}_{8\times8}\boldsymbol{X}\boldsymbol{C}_{8\times8}^{T} \tag{4-32}$$

$$\boldsymbol{C}_{8\times8}=\begin{bmatrix}64 & 64 & 64 & 64 & 64 & 64 & 64 & 64\\ 89 & 75 & 50 & 18 & -18 & -50 & -75 & -89\\ 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83\\ 75 & -18 & -89 & -50 & 50 & 89 & 18 & -75\\ 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64\\ 50 & -89 & 18 & 75 & -75 & -18 & 89 & -50\\ 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36\\ 18 & -50 & 75 & -89 & 89 & -75 & 50 & -18\end{bmatrix} \tag{4-33}$$

假设一帧图像可以划分为 N 个 8×8 子单元，第 i 子单元对应的二维 DCT 变换系数的绝对值之和为 D_i，该帧图像中子单元对应的变换系数绝对值之和的平均值为 D_{ave}，计算公式如式 4-34。

$$D_{ave}=\frac{1}{N}\sum_{i=1}^{N}D_i \tag{4-34}$$

假设一个 LCU 中含有 M 个 8×8 子单元，第 j 个子单元对应的二维 DCT 变换系数的绝对值之和为 D_j，则该 LCU 中子单元对应的变换系数绝对值之和的平均值为 DL_{ave}，计算公式如式 4-35。

$$DL_{ave}=\frac{1}{M}\sum_{j=1}^{M}D_j \tag{4-35}$$

最后，将求得的 LCU 的变换系数均值与其所在图像的变换系数均值进行比较，其结果 ω_2 越大，表明当前 LCU 编码时需要的比特数倾向于更多，如式 4-36。

$$\omega_2=\frac{DL_{ave}}{D_{ave}} \tag{4-36}$$

$$\omega_{new\_LCU}=a\omega_1+(1-a)\omega_2 \tag{4-37}$$

$$T_{LCU}=\frac{T_{Pic}-Bit_{H}-Coded_{Pic}}{\sum_{\{AllNotCodedLCU\}}\omega_{new\_LCU}}\times\omega_{new\_LCUCurr} \tag{4-38}$$

本文综合利用 ω_1 和 ω_2 设计了一种新的 LCU 目标比特分配权重计算方式，如式 4-37，其中 a 和 $1-a$ 分别表示 ω_1 和 ω_2 在权重中所占的比重，该值取 0.7，新的 LCU 层目标比特分配公式如式 4-38。其中 Bit_H 是该图像头信息比特数的预测值，其值为视频序列中与当前图像处于相同层的所有已编码图像头信息所用实际比特的平均值，T_{Pic} 是当前图像的目标比特数，$Coded_{Pic}$ 表示当前图像已经使用的比特数，ω_{new_LCU} 表示式 4-37 设计的 LCU 权重值。然后通过缓冲区状态对初步分配的目标比特 T_{LCU} 进行调整，得到最终的 LCU 分配目标比特数。

4.3.2　LCU 层模型参数更新计算

当前 LCU 在编码完成之后，所需的编码比特数和失真是可以确定的，利用式 4-39 可以求得该 LCU 对应的率失真曲线的参数 C 和 K，其中 MSE 和 bpp 的定义式为 4-6 和式 4-5。

$$\begin{cases}MSE=C\cdot bpp^{-K}\\ \lambda=C\cdot K\cdot bpp^{-K-1}\end{cases} \tag{4-39}$$

当前 LCU 的率失真特性是作为下一帧同层图像相同位置处的 LCU 的参考，但即使处于同层的两帧图像，它们的参考帧的图像质量也并不是完全相同的，这就导致了不同的率失真特性。如在默认情况下，码率控制算法中 GOP 中含有 4 帧图像，这 4 帧图像按照播放顺序其层级依次为 3、2、3、1，分配目标比特时权重如表 4-1 所示。

表 4-4　编码过程中部分参数

帧序号	层级	Y-PSNR	参考帧
16	1	39.5438	15、12、8、4
17	3	36.8813	16、12、8、4
18	2	37.2003	17、16、12、8
19	3	36.4932	18、16、12、8

表 4-4 中显示了在开启码率控制的情况下，标准测试序列 BasketballDrill_832x480_50 在编码过程中的一些信息，其中 Y-PSNR（Peak Signal to Noise Ratio，峰值信噪比）表示 Y 通道的峰值信噪比，用于表征重建帧 Y 通道的失真情况。

由表 4-4 可知，第 17 帧和第 19 帧虽然同为层级 3，但第 17 帧的参考帧的质量是要高于第 19 帧的，即表示第 17 帧的参考帧更加接近于原始视频，这意味着第 17 帧在编码过程中产生的残差信息更小，编码需要的比特数更少，其率失真特性要好于第 19 帧。而在编码过程中第 17 帧的率失真特性会做为第 19 帧的参考，因此本文对第 17 帧的率失真特性做了修正，如式 4-40。

$$\begin{cases} K = K \times 1.18 \\ C = C \times 1.05 \end{cases} \tag{4-40}$$

将该修正扩展到所有层级为 3 的帧中，如式 4-41 和式 4-42。

$$\begin{cases} K = K \times 1.18 \\ C = C \times 1.05 \end{cases} \quad frame\_order = 1,5,9,13,17... \tag{4-41}$$

$$\begin{cases} K = K \times 0.85 \\ C = C \times 0.95 \end{cases} \quad frame\_order = 3,7,11,15,19... \tag{4-42}$$

当率失真曲线的参数 C 和 K 计算完成后，可以利用式 4-15 中的关系得出 $R-\lambda$ 模型中的参数 α 和 β，如式 4-43，该参数将用于计算和当前 LCU 所在帧相同级别的下一帧中与当前 LCU 同位置的 LCU 的 λ 值。

$$\begin{cases} \alpha = C \cdot K \\ \beta = -K - 1 \end{cases} \tag{4-43}$$

4.3.3　整体算法流程

码率控制算法的整体流程图如图 4-9 所示。4.3.1 节和 4.3.2 节分别介绍了本文对 LCU 层码率控制算法的目标比特分配模块和模型参数更新模块的改进，分别对应着流程图 4-9 中的绿色方框和蓝色方框。

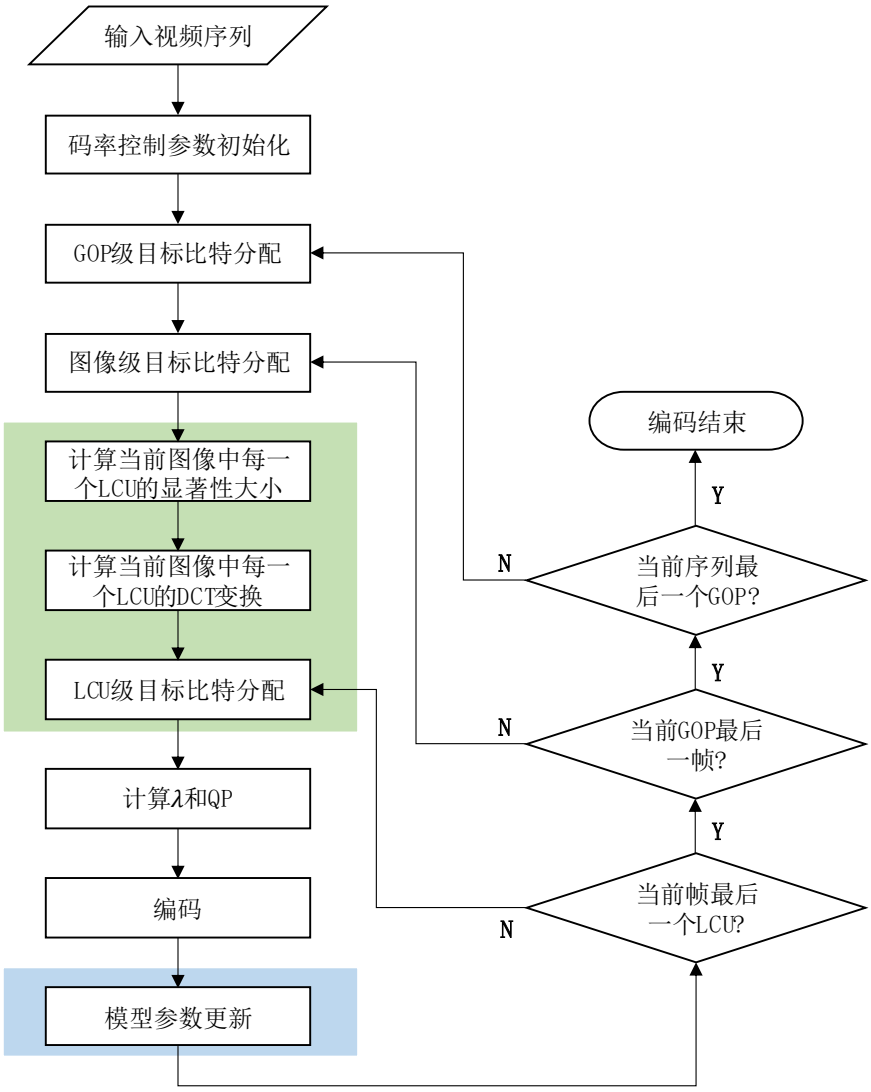


图 4-9　码率控制算法流程图

具体的处理流程如下：

改进的码率控制算法

1：　输入待编码视频序列，初始化码率控制参数；

2：　利用式 4-7、式 4-8 完成 GOP 级目标比特分配；

3：　利用式 4-9 完成图像级目标比特分配；

4：　利用式 4-24 至式 4-30 计算一帧图像的显著性图，并计算每一个 LCU 的显著性相对大小；

5：　利用式 4-32 至式 4-36 计算一帧图像中 8×8 子单元的 DCT 变换系

数绝对值之和的平均值，并计算当前 LCU 中子单元的 DCT 变换系数绝对值之和的平均值；

6：　利用式 4-37 和式 4-38 完成 LCU 级别目标比特分配；

7：　利用式 4-15 和式 4-16 计算当前 LCU 的 λ 和 QP 值；

8：　对当前 LCU 进行编码；

9：　利用式 4-39 至式 4-43 完成 LCU 层中 $R-\lambda$ 模型中参数的更新；

10：　判断当前 LCU 是否为帧中最后一个 LCU，若是，则进入 11；若不是，则进入 6，继续下一个 LCU 的编码过程；

11：　判断当前帧是否为 GOP 中最后一个帧，若是，则进入 12；若不是，则进入 3，开始为 GOP 中的下一帧分配目标比特；

12：　判断当前 GOP 是否为视频序列的最后一个 GOP，若是，则进入 13；若不是，则进入 2，开始为下一个 GOP 分配目标比特；

13：　编码结束。

4.4　实验结果与分析

为了检验 4.3 节所提算法的性能，本节从 3 个方面进行分析，分别为率失真性能分析、码率控制精度分析和视频主观质量分析。实验参数如表 4-5 和表 4-6 所示，其余参数为 encoder_lowdelay_P_main 配置文件默认参数。

表 4-5　开启码率控制时实验参数

参数	
测试平台	HM10.1
LCULevelRateControl	1
RCLCUSeparateModel	1
InitialQP	0
RCForceIntraQP	0

表 4-6　测试序列编码帧数

视频分辨率	编码帧数
416×240	240
832×480	120

1280×720	120
1920×1080	60

4.4.1　率失真性能分析

分析算法的率失真性能时通常使用 BD-BR 和 BD-PSNR 这两个量，其中 BD-BR 的含义是与原本的算法相比，提出算法在相同的视频质量下码率的变化；BD-PSNR 的含义是与原本的算法相比，提出算法在相同码率下视频质量的变化。为了求出这两个值，至少需要 4 组码率和失真，这里选择的 QP 值为 22、27、32 和 37，实验结果如表 4-7 所示。

表 4-7 中，Ref.[53] method 表示文献[53]中的码率控制算法，Proposed method 表示本文 4.3 节提出的改进的码率控制算法，比较对象均为无码率控制算法时的编码结果。

表 4-7　Y 通道 BDBR(%)和 BDPSNR(dB)实验结果

分辨率	序列名	Ref.[53] method		Proposed method	
		BDBR	BDPSNR	BDBR	BDPSNR
1920×1080	Kimono1	10.41	-0.3491	11.97	-0.3977
	Cactus	1.08	-0.0140	-0.78	0.0282
1280×720	FourPeople	27.02	-0.8000	24.92	-0.7489
	Johnny	58.40	-1.0990	52.14	-1.0033
	SlideEditing	62.83	-6.8843	54.77	-6.2474
	SlideShow	28.78	-1.9130	21.82	-1.5004
	vidyo1	48.12	-1.2411	40.54	-1.0717
	vidyo3	30.17	-0.8591	23.39	-0.6846
832×480	BasketballDrill	-7.19	0.2839	-8.64	0.3444
	BQMall	-0.58	0.0250	-0.76	0.0326
	BasketballDrillText	-8.38	0.3523	-8.97	0.3777
	PartyScene	2.32	-0.0971	1.17	-0.0487
416×240	BasketballPass	-4.03	0.1952	-6.14	0.3010
	RaceHorses	4.33	-0.2050	3.77	-0.1823
	BQSquare	0.67	-0.0359	1.42	-0.0651
	均值	16.93	-0.8427	14.04	-0.7244

由表 4-7 可知，与文献[53]中的码率控制算法相比，4.3 节提出的改进的码率控制算法可以平均降低 2.89%的码率。除测试序列 BQSquare、Kimono1 之外，编码其余 15 个测试序列时，在一定的码率范围内，使用 4.3 节提出的码率控制算法的率失真性能均优于使用文献[53]中的码率控制算法，通过图 4-10 和图 4-11 中的率失真曲线可以更加直观地观察到该结果。

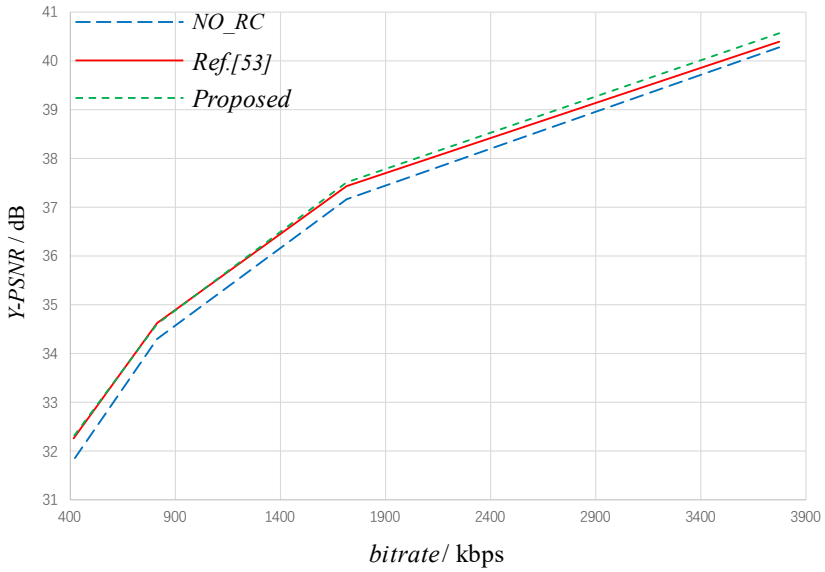


图 4-10　BasketballDrill 率失真曲线

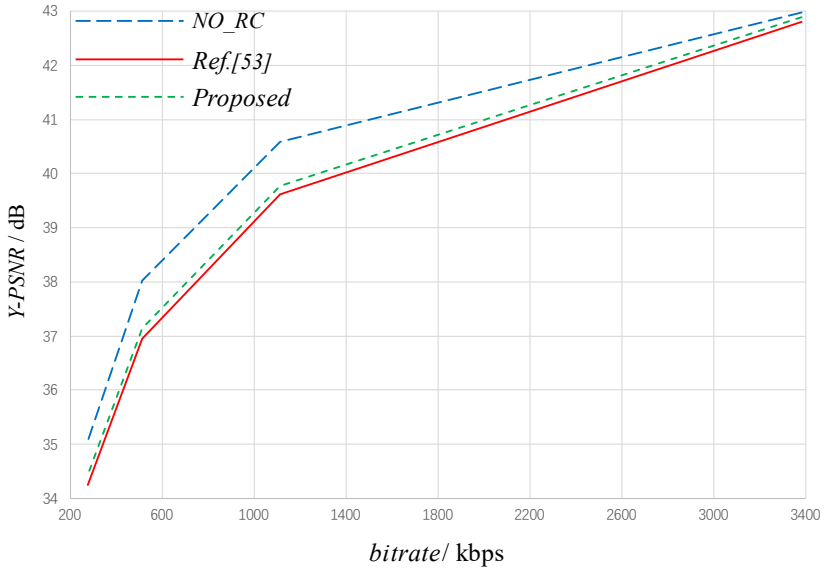


图 4-11　vidyo3 率失真曲线

图 4-10 和图 4-11 中显示了标准测试序列 BasketballDrill 和 vidyo3 的率失真曲线，其中 NO_RC 表示无码率控制时的率失真曲线、Ref.[53]表示使用文献[53]中的码率控制算法对应的率失真曲线、Proposed 表示使用 4.3 节提出的改进的码率控制算法对应的率失真曲线。两张图中，Proposed 对应的率失真性能均优于 Ref.[53]。

4.4.2　码率控制精度分析

码率控制算法设计的目的就是解决视频在不同带宽信道下的传输问题，因此设定的目标码率和真实码率之间的接近程度是评价一个码率控制算法性能的重要标准之一。本小节选择了量化参数为 22、27、32 和 37 时的视频码率作为参考，实验结果如表 4-8。其中一个测试序列的 MER（Match Error Ratio, 匹配错误率）的定义如式 4-44。

$$MER=\frac{1}{4}\sum_{QP=22,27,32,37}\frac{\left|R_{target}^{QP}-R_{actual}^{QP}\right|}{R_{target}^{QP}}\times 100\% \qquad (4\text{-}44)$$

其中 R_{target}^{QP} 表示分配的目标码率，其值为无码率控制时量化参数选择 QP 时需要的编码速率，R_{actual}^{QP} 表示在 R_{target}^{QP} 目标码率下使用码率控制算法编码视频序列实际的速率。MER 的大小表征了目标码率和真实码率之间的接近程度。

表 4-8　码率控制精度（%）

分辨率	序列名	Ref.[53] MER	Proposed MER
1920×1080	Kimono1	0.291	0.179
	Cactus	0.108	0.023
1280×720	FourPeople	0.440	0.323
	Johnny	0.820	0.459
	SlideEditing	7.156	5.524
	SlideShow	4.059	1.475
	vidyo1	0.703	0.714
	vidyo3	0.454	0.173
832×480	BasketballDrill	0.343	0.216
	BQMall	0.154	0.104

	BasketballDrillText	0.381	0.194
	PartyScene	0.106	0.124
416×240	BasketballPass	0.435	0.425
	RaceHorses	0.309	0.257
	BQSquare	0.486	0.441
均值		1.083	0.709

由表 4-8 可得，4.3 节提出的码率控制算法相比于文献[53]中的码率控制算法具有更高的码率控制精度，MER 均值降低了 0.374%，即使用本文所提算法压缩视频得到的码率更加接近目标码率，提高了视频实时传输的稳定性。

4.4.3　视频主观质量分析

本章提出的算法为基于视觉显著性检测的码率控制算法，由 4.3.1 节中目标比特分配权重计算公式可知，若一帧图像中某 LCU 的视觉显著性较高，则其分配的目标比特数将越多，其对应的 QP 就会越小，那么该 LCU 中细节显示就会越清晰。图 4-12 中（a）、（b）、（c）是测试序列 BQMall 在 665.920kbps 码率下，没有使用码率控制算法、使用文献[53]中的码率控制算法、使用本文提出的码率控制算法的解码帧的对比图像，且选取的图像块显著性较高。从图中可以清晰地看出，（c）图中白色手提袋上的字母、白色手提袋的轮廓以及黑色包的细节都更为清晰。由此可得，若使用本文提出的基于视觉显著性检测的 LCU 层码率控制算法，一帧图像中显著性较高的区域将会给人更好的观看感受。

(a)　　(b)　　(c)

图 4-12　测试序列 BQMall 解码帧观看感受对比

4.5　本章小结

本章首先详细地介绍λ域码率控制算法，主要包括 GOP 级、图像级和 LCU 级别的目标比特分配，以及模型参数更新模块。然后介绍了几种典型的视觉显著性检测算法，并选择 FT 算法作为接下来设计 LCU 层目标比特权重计算的视觉显著性算法。最后设计了一种新的 LCU 层目标比特分配权重计算方法以及模型参数更新方法且对算法性能进行了实验验证。

实验结果表明，本文提出的算法在率失真性能、码率精度控制性能以及显著性较高区域的观看感受均优于文献[53]中的码率控制算法。

第 5 章　全文总结与展望

5.1　全文总结

针对 HEVC 复杂度较高以及视频编码标准实际应用场景中信道带宽波动问题，本文提出了帧间预测编码单元快速划分算法和基于视觉显著性检测的 LCU 层码率控制算法。论文主体为第三、四章，分别详细介绍了本文所提的两种算法，论文其余部分介绍了研究背景、国内外研究现状以及与本文研究课题关系密切的关键技术。

论文主体的具体内容如下：

（1）为了降低 HEVC 的计算复杂度，本文首先通过实验观察了编码单元划分深度的分布特征，然后通过实验寻找 BSAD 与编码单元划分深度之间的相关性，即 BSAD 值越小，当前 CTU 的划分深度为零的概率就越大。为了进一步提高编码单元划分深度的预测准确性，本文也通过实验发现时空相邻的 CTU 的编码单元划分深度具有很大的相似性。综合考虑以上两个相关性，本文设计了一种修剪编码单元深度遍历范围的算法。实验结果表明，本文提出的算法可以平均提高 29.40%的编码速度，而码率仅仅增加 1.30%。

（2）为了提高码率控制算法的码率控制精度和率失真性能，本文对 HM10.1 中的λ域码率控制算法的 LCU 层的目标比特分配权重计算和模型参数更新进行了改进。实验结果表明，本文提出的算法在率失真性能、码率控制精度、视频的主观质量方面均优于测试平台 HM10.1 自带的码率控制算法。

5.2　后续工作展望

本文基于 HEVC 编码标准研究了低复杂度算法与码率控制算法，但由于读研时间有限，很多想法不能去验证。若将来有机会，则会继续探索以下内容：

（1）HEVC 的低复杂度算法设计通常计算某些特征值并判断是否满足预设的标准，若满足，则终止或者简化对应的处理过程，以此来提高编码速度，如本文所提算法利用 BSAD 值来简化编码单元划分过程。因此，如果可以找到特征值与处理过程间更加精确的相关性，或者针对某一处理过程，寻找关系更加紧密

的特征值。若可以做到，则对处理过程的终止和简化将更加准确，做到编码效率不变的情况下极大地提高编码速度。

（2）码率控制算法的核心有两个，一是目标比特分配，二是为编码单元选择合适的编码参数。在视频率失真曲线上假设有某点(R_T, D_T)，即表示当给定目标码率为R_T时，D_T是在 HEVC 标准框架下能够达到的最优的失真性能。该点上每个编码单元需要的目标比特和编码参数是固定的，若码率控制算法的目标比特分配和编码参数选择与其越接近，则码率控制算法的精度和率失真性能就会越好。因此，如何分配目标比特，如何为编码单元选择编码参数仍是一个值得继续深入研究的课题。

参考文献

[1] Wiegand T， Sullivan G J， Bjontegaard G , et al. Overview of the H.264/AVC video coding standard[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2003, 13(7):560-576.

[2] Sullivan G J， Ohm J R， Han W J , et al. Overview of the high efficiency video coding (HEVC) standard[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2012, 22(12):1649-1668.

[3] 周巍，周欣，张冠文，等. HEVC 视频编码优化与实现[M]. 北京：电子工业出版社，2019. 1~2.

[4] 王建富. H.265/HEVC 编码加速算法研究[D]. 合肥：中国科学技术大学，2015.

[5] Hosseini E， Pakdaman F， Hashemi M R, et al. A computationally scalable fast intra coding scheme for HEVC video encoder [J]. Multimedia tools applications, 2019, 78(9): 11607-11630.

[6] 石敏，席诗华，易清明. 基于预测单元尺寸的高效视频编码帧内预测模式快速选择的改进算法[J]. 激光与光电子学进展，2019，56(20)：234-242.

[7] Yan Z， Cho S-Y， Welsen S. Fast intra prediction mode decision for HEVC using random forest[C]. Proceedings of the 2019 International Conference on Image, Video and Signal Processing, 2019:45-49.

[8] Mengmeng Z， Xiaojun Z， Zhi L . Fast and Adaptive Mode Decision and CU Partition Early Termination Algorithm for Intra-prediction in HEVC[J]. EURASIP Journal on Image and Video Processing, 2017:86.

[9] Zhao L， Fan X， Ma S , et al. Fast intra-encoding algorithm for high efficiency video coding [J]. Signal Processing: Image Communication, 2014, 29(9):935-944.

[10] Shang X， Wang G， Fan T , et al. Fast CU size decision and PU mode decision algorithm in HEVC intra coding[C]. 2015 IEEE International Conference on Image Processing (ICIP), 2015:1593-1597.

[11] Sun X， Chen X， Xu Y , et al. Fast CU size and prediction mode decision algorithm for HEVC based on direction variance[J]. Journal of Real-Time Image Processing, 2019, 16(5): 1731-1744.

[12] Zhou C， Zhou F， Chen Y. Spatio-temporal correlation-based fast coding unit depth decision for high efficiency video coding[J]. Journal of electronic imaging, 2013, 22(4):043001.1-043001.13.

[13] Kun D， Pengyu L， Kebin J， Zeqi F. An Adaptive Quad-Tree Depth Range Prediction Mechanism for HEVC [J]. IEEE Access, 2018, 6:54195-54206.

[14] Shen L， Zhang Z， An P. Fast CU size decision and mode decision algorithm for HEVC intra coding[J]. IEEE Transactions on Consumer Electronics, 2013, 59(1):207-213.

[15] Zhao W， Onoye T， Song T. Hierarchical Structure-Based Fast Mode Decision for H.265/HEVC[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2015, 25(10):1651-1664.

[16] Shen L， Zhang Z， Liu Z. Adaptive Inter-Mode Decision for HEVC Jointly Utilizing Inter-Level and Spatiotemporal Correlations[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2014, 24(10):1709-1722.

[17] Kuo Y T， Chen P Y， Lin H C. A Spatiotemporal Content-Based CU Size Decision Algorithm for HEVC[J]. IEEE Transactions on Broadcasting, 2020, PP(99):1-13.

[18] Zhuoming L， Yu Z , et al. A Fast CU Partition Method Based on CU Depth Spatial Correlation and RD Cost Characteristics for HEVC Intra Coding[J]. Signal Processing: Image Communication, 2019, 75:141-146.

[19] 仲伟波，陈东，姚旭洋，冯友兵. 利用时空相关性的 HEVC 帧内编码块快速划分[J]. 中国图象图形学报，2018，23(02)：155-162.

[20] Cen Y F， Wang W L， Yao X W. A fast CU depth decision mechanism for HEVC[J]. Information Processing Letters, 2015, 115(9):719-724.

[21] Nalluri P， Alves L， Navarro A. Complexity Reduction Methods for Fast Motion Estimation in HEVC[J]. Signal Processing Image Communication, 2015, 39:280-292.

[22] Gao Y， Liu P， Wu Y , et al. Quadtree Degeneration for HEVC[J]. IEEE Transactions on Multimedia, 2016, 18(12):2321-2330.

[23] Cho S， Kim M. Fast CU splitting and pruning for suboptimal CU partitioning in HEVC intra coding[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2013, 23(9):1555-1564.

[24] Kim J， Choe Y， Kim Y G. Fast coding unit size decision algo rithm for intra coding in HEVC[C]. Proceedings of IEEE Inter national Conference on Consumer Electronics, Las Vegas:IEEE, 2013:637-638.

[25] Xiong J， Li H， Meng F , et al. Fast HEVC inter CU decision based on latent SAD estimation[J]. IEEE Transactions on Multimedia, 2015, 17(12):2147-2159.

[26] Lee J， Kim S， Lim K, et al. A fast CU size decision algorithm for HEVC[J]. IEEE Transactions on Circuits Systems for Video Technology, 2014, 25(3):411-421.

[27] Ahn S， Lee B， Kim M. A novel fast CU encoding scheme based on spatiotemporal encoding parameters for HEVC inter coding[J]. IEEE Transactions on Circuits Systems for Video Technology, 2014, 25(3):422-435.

[28] Caixia B， Chun Y. Fast Coding Tree Unit Decision for HEVC Intra Coding[C]. IEEE International Conference on Consumer Electronics, 2013:28-31.

[29] Maazouz M， Batel N， Bahri N , et al. Homogeneity-based fast CU partitioning algorithm for HEVC intra coding[J]. Engineering Science Technology, an International Journal, 2019, 22(3):706-714.

[30] Min B， Cheung R C. A fast CU size decision algorithm for the HEVC intra encoder[J]. IEEE Transactions on Circuits Systems for Video Technology, 2014, 25(5):892-896.

[31] Zhiyao Y, Qing L, et al. Fast coding algorithm for HEVC based on video contents[J]. IET Image Processing, 2017, 11(6):343-351.

[32] Zhang M， Liu Y， Liu Z. A new fast algorithm based on SATD for HEVC intra prediction[C]. 2016 Visual Communications and Image Processing (VCIP), 2016:1-4.

[33] 周帅燃，杨静. 低复杂度 HEVC 帧内编码快速划分算法[J]. 小型微型计算机系统，2021，42(7)：4.

[34] Kim H S， Park R H. Fast CU Partitioning Algorithm for HEVC Using an Online-Learning-Based Bayesian Decision Rule[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2016, 26(1):130-138.

[35] Xiong J， Li H， Meng F , et al. MRF-Based Fast HEVC Inter CU Decision With the Variance of Absolute Differences[J]. IEEE Transactions on Multimedia, 2014, 16(8):2141-2153.

[36] Zhang Y， Kwong S， Wang X , et al. Machine Learning-Based Coding Unit Depth Decisions for Flexible Complexity Allocation in High Efficiency Video Coding[J]. IEEE Transactions on Image Processing, 2015, 24(7):2225-2238.

[37] Ruiz D， Fernandez-Escribano G， Adzic V , et al. Fast CU partitioning algorithm for HEVC intra coding using data mining[J]. Multimedia Tools & Applications, 2017, 76(1):861-894.

[38] Zhang Y， Zhang C， Fan R , et al. Recent Advances on HEVC Inter-frame Coding From Optimization to Implementation and Beyond[J]. IEEE Transactions on Circuits Systems for Video Technology, 2019, 30(11):4321-4339.

[39] Luo J， Yang X， Liu L. A fast motion estimation algorithm based on adaptive pattern and search priority[J]. Multimedia Tools Applications, 2015, 74(24):11821-11836.

[40] Medhat A， Shalaby A， Sayed M S , et al. Adaptive low-complexity motion estimation algorithm for high efficiency video coding encoder[J]. IET Image Processing, 2016, 10(6): 438-447.

[41] Ko Y-H， Kang H-S， Lee S-W. Adaptive search range motion estimation using neighboring motion vector differences[J]. IEEE Transactions on Consumer Electronics, 2011, 57(2):726-730.

[42] Dai W , Au O C , Li S , et al. Adaptive search range algorithm based on Cauchy distribution[C]. 2012 Visual Communications and Image Processing, 2012:1-5.

[43] Jeong J H , Parmar N , Sunwoo M H. Enhanced test zone search algorithm with rotating pentagon search[C]. 2015 International SoC Design Conference (ISOCC), 2015:275-276.

[44] Yang S-H , Jiang J-Z , Yang H-J. Fast motion estimation for HEVC with directional search[J]. Electronics Letters, 2014, 50(9):673-675.

[45] Z. Z. Chen , K. N. Ngan. Recent advances in rate control for video coding[J]. Signal Processing-Image Communication, 2007, 22(2007):19-38.

[46] 罗敏珂，周益民，钟敏. AVS2 视频编码码率控制算法[J]. 系统工程与电子技术，2016，38(9)：2192-2200.

[47] T. H. Chiang , Y. Q. Zhang. A new rate control scheme using quadratic rate distortion model [J]. IEEE Transactions on Circuits and Systems for Video Technology, 1997, 7(1):246-250.

[48] H. J. Lee , T. Chiang , Y. Q. Zhang. Scalable rate control for MPEG-4 video[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2000, 10(6):878-894.

[49] K. N. Ngan , T. Meier , Z. Z. Chen. Improved single-video-object rate control for MPEG-4[J]. IEEE Transactions on Circuits And Systems for Video Technology, 2003, 13(5):385-393.

[50] Z. Li , F. Pan , K. Pang. Adaptive basic unit layer rate control for JVT[R]. Pattaya,Thailand:Document JVT-G012, March, 2003.

[51] H. Choi , J. Nam , J. Yoo , et al. Rate control based on unified RQ model for HEVC[R]. San José, CA, USA:Document JCTVC-H0213, February, 2012.

[52] B. Li , H. Li , L. Li , et al. Rate control by R-lambda model for HEVC[R]. Shanghai, China:Document JCTVC-K0103, October, 2012.

[53] B. Li , H. Q. Li , L. Li , et al. λ domain rate control algorithm for high efficiency video coding[J]. IEEE Transactions on Image Processing, 2014, 23(9):3841-3854.

[54] M. Karczewicz , X. Wang. Intra frame rate control based on SATD[R]. Incheon, Korea: Document JCTVC-M0257, April, 2013.

[55] B. Li , H. Li , L. Li. Adaptive bit allocation for R-lambda model rate control in HM[R]. Incheon, Korea:Document JCTVC-M0036, April, 2013.

[56] J. T. Wen , M. Y. Fang , M. H. Tang, et al. R-lambda model based improved rate control for HEVC with pre-encoding[C]. Data Compression Conference(DCC), Snowbird, UT, USA, 2015, 53-62.

[57] I. Zupancic , E. Izquierdo , M. Naccari , et al. Two-pass rate control for UHDTV delivery with HEVC[C]. Picture Coding Symposium(PCS), Nuremberg, Germany, 2016, 1-5.

[58] I. Zupancic , M. Naccari , M. Mrak , et al. Two-pass rate control for improved quality of experience in UHDTV delivery[J]. IEEE Journal of Selected Topics in Signal Processing, 2017, 11(1):167-179.

[59] S. Li , M. Xu , Z. Wang , et al. Optimal bit allocation for CTU level rate control in HEVC [J]. IEEE Transactions on Circuits and Systems for Video Technology, 2017, 27(11):2409-2424.

[60] Y. Hou , Y. Ye , J. Lei , et al. Rate control for HEVC based on spatio-temporal context and motion complexity[J]. Multimedia Tools and Applications, 2016, 76(12):14035-14053.

[61] M. H. Wang , K. N. Ngan. Optimal bit allocation in HEVC for real-time video communications[C]. International Conference on Image Processing (ICIP), Quebec City, QC, Canada, 2015, 2665-2669.

[62] H. Sun , S. S. Gao , C. Zhang. Adaptive bit allocation scheme for rate control in high efficiency video coding with initial quantization parameter determination[J]. Signal Processing-Image Communication, 2014, 29(2014):1029-1045.

[63] M. H. Liu , P. Ren , Z. Xiang. Frame-level bit allocation for hierarchical coding of H.265/HEVC considering dependent rate-distortion characteristics[J]. Signal Image and Video Processing, 2016, 10(8):1457-1463.

[64] A. Fiengo , G. Chierchia , M. Cagnazzo , et al. Rate allocation in predictive video coding using a convex optimization framework[J]. IEEE Transactions on Image Processing, 2017, 26(1):479-489.

[65] F. Song , C. Zhu , Y. Liu , et al. A new GOP level bit allocation method for HEVC rate control[C]. International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), Cagliari, Italy, 2017, 1-4.

[66] S. Li , M. Xu , X. Deng , et al. Weight-based R-λ rate control for perceptual HEVC coding on conversational videos[J]. Signal Processing-Image Communication, 2015, 38(2015):127-140.

[67] H. Q. Zeng , A. S. Yang , K. N. Ngan , et al. Perceptual sensitivity-based rate control method for high efficiency video coding[J]. Multimedia Tools and Applications, 2016, 75(17):10383-10396.

[68] Itti L. A model of saliency-based visual attention for rapid scene analysis[J]. IEEE Trans, 1998, 20.

[69] Hou X , Zhang L. Saliency Detection: A Spectral Residual Approach[C]. IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 2007.

[70] R Achanta† , S Hemami‡ , F Estrada† , et al. Frequency-tuned salient region detection[C]. IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 2009.

[71] Cheng , Mitra N J , Huang X , et al. Salient Object Detection and Segmentation[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2013, 37(3):1.

[72] Ahmadi M , Karimi N , Samavi S. Context-Aware Saliency Detection for Image Retargeting Using Convolutional Neural Networks[J]. 2019.

[73] Yang C , Zhang L , Lu H. Graph-Regularized Saliency Detection With Convex-Hull-Based Center Prior[J]. IEEE Signal Processing Letters, 2013, 20(7):637-640.