



研究与开发

基于随机森林的 HEVC 复杂度控制方法

文鹏, 彭宗举, 陈芬, 蒋刚毅, 郁梅

(宁波大学信息科学与工程学院, 浙江 宁波 315211)

摘要: 高效视频编码 (high efficiency video coding, HEVC) 具有较高的计算复杂度, 而快速编码算法不能在给定编码时间下完成视频编码, 因此提出了一种基于随机森林的 HEVC 复杂度控制方法。首先训练得到 3 种不同预测准确率的随机森林分类器, 实现编码树单元 (coding tree unit, CTU) 的多种编码配置; 然后, 通过建立平均深度—复杂度模型为 CTU 分配复杂度; 最后通过平坦度、平均深度、比特和 CTU 级累计编码误差确定的 CTU 编码配置完成复杂度控制。实验结果表明, 本文算法复杂度控制精度较高, 在率失真性能方面优于当前复杂度控制方法。

关键词: 复杂度控制; 高效视频编码; 随机森林

中图分类号: TN919.8

文献标识码: A

doi: 10.11959/j.issn.1000-0801.2019034

Complexity control method of random forest based HEVC

WEN Peng, PENG Zongju, CHEN Fen, JIANG Gangyi, YU Mei

Faculty of Information Science and Engineering, Ningbo University, Ningbo 315211, China

Abstract: High efficiency video coding (HEVC) has high computational complexity, and fast algorithm cannot perform video coding under restricted coding time. Therefore, a complexity control method of HEVC based on random forest was proposed. Firstly, three random forest classifiers with different prediction accuracy were trained to provide various coding configurations for coding tree unit (CTU). Then, an average depth-complexity model was built to allocate CTU complexity. Finally, the CTU coding configuration, determined by the smoothness, average depth, bit, and CTU-level accumulated coding error, was used to complete complexity control. The experimental results show that the proposed method has better complexity control precision, and outperforms the state-of-the-art method in terms of video quality.

Key words: complexity control, HEVC, random forest

收稿日期: 2018-05-25; 修回日期: 2019-01-20

基金项目: 国家自然科学基金资助项目 (No.61771269, No.61620106012, No.61671258); 浙江省自然科学基金资助项目 (No.LY17F010005); 宁波市自然科学基金资助项目 (No.2018A610052)

Foundation Items: The National Natural Science Foundation of China (No.61771269, No.61620106012, No.61671258), The Natural Science Foundation of Zhejiang Province of China (No.LY17F010005), The Natural Science Foundation of Ningbo of China (No.2018A610052)

1 引言

随着视频分辨率的不断提高,视频编码标准 H.264 已经难以满足超高清视频存储和传输的要求,因此 ITU-T VCEG (Video Coding Experts Group) 和 ISO/IEC MPEG (Moving Pictures Experts Group) 成立视频编码联合专家组 (Joint Video Exploration Team, JVET),制定了新一代高效视频编码 (high efficiency video coding, HEVC) 标准^[1]。

HEVC 仍沿用 H.264 的混合编码结构^[2],但为了取得更高的压缩效率,采用了许多改进的编码技术。在帧内编码方面,H.264 中的 9 种角度预测模式被增至 35 种^[3]。在帧间编码方面,相比 H.264 增加了非对称划分模式^[4],且支持 1/8 像素精度的预测补偿。在划分结构方面,HEVC 采用了灵活的四叉树结构。通过新的编码技术,HEVC 相比 H.264 在同等客观质量下压缩效率提升了 44%,但计算复杂度也成倍增加^[5]。

针对编码复杂度高的问题,许多学者通过快速编码单元 (coding unit, CU) 尺寸决策和预测单元 (prediction unit, PU) 模式快速选择等方法来减少编码复杂度^[6-9]。但对于实时应用而言,如何在给定编码复杂度情况下,使得编码器能实时编码任意视频序列至关重要。针对这一问题,Correa 等人^[10]利用编码树单元 (coding tree unit, CTU) 时空域相关性对最大分割深度进行限制,并通过控制受限 CTU 的数量完成复杂度控制。Correa 等人^[11]进一步通过控制受限帧的数量来实现复杂度控制。参考文献[10-11]所提算法虽利用 CTU 时空域相关性减少复杂度,但不能保证纹理丰富序列的率失真性能。Correa 等人^[12]利用时域同位 CTU 的率失真代价作为判断当前 CTU 受限或不受限的依据,使其具有更好的率失真性能。Correa 等人^[13]通过调整编码参数配置使可控制编码复杂度低至 20%。参考文献[12-13]所提算法对受限 CTU 进行了合理的初步分配,但仍不能保证

纹理丰富序列具有较好的率失真性能。参考文献[14]通过在线训练得到不同分割深度提前终止的阈值集,通过动态调整阈值提前终止 CU 分割的递归过程,使目标复杂度 T_c (即目标编码时间) 的归一化比例能控制在 60%。Zhang 等人^[15]利用统计模型估计 CTU 编码复杂度,并合理地限制 CU 深度遍历范围,但深度遍历范围内的复杂度冗余有待进一步节省。参考文献[16]利用视觉感知因子对最大分割深度进行限定,实现复杂度控制。参考文献[17]进一步研究了最大分割深度与复杂度的关系,并结合 CTU 时域相关性和视觉权重来限定 CTU 的最大分割深度,在既保证主观质量又保证客观质量的同时,实现了复杂度控制。参考文献[16-17]的算法对纹理不丰富的序列较为有效,对纹理丰富的序列存在较大的率失真性能损失。

针对以往复杂度控制算法难以保证纹理丰富序列的率失真性能,本文提出了一种基于随机森林的复杂度控制算法。首先在 CU 深度为 0、1 和 2 的情况下,基于随机森林建立了 3 种预测准确率的两类 (跳过和提前终止) 分类器为 CTU 提供多种编码配置。然后建立平均深度-复杂度 (mean depth-complexity, MDC) 模型为 CTU 分配编码复杂度。最后通过平坦度、平均深度、比特和 CTU 级累计编码误差选择编码配置以减少复杂度。实验结果表明,本文算法能较好地保证纹理丰富序列的率失真性能。

2 HEVC 编码结构

HEVC 将每一帧都分割成若干大小为 64×64 的 CTU,每个 CTU 通过四叉树结构进行递归遍历选择最优的 CU 分割尺寸和 PU 预测模式^[18]。CTU 为最大的编码单元 (largest coding unit, LCU),其划分深度 d 范围为 0~3,对应的 CU 尺寸为 $(64 \times 64) \sim (8 \times 8)$ 。一个 CU 能递归地划分成 4 个大小相同的子 CU,直到划分成 8×8 大小的 CU 为止。一个 $2N \times 2N$ 大小的 CU 通过遍历 11 种 PU 预



测模式, 包括 skip/merge 模式、8 种帧间模式 ($2N \times 2N$, $N \times N$, $N \times 2N$, $2N \times N$, $2N \times nU$, $2N \times nD$, $nL \times 2N$, $nR \times 2N$) 和两种帧内模式 ($2N \times 2N$, $N \times N$), 其中 N 表示 CU 大小的一半, 表明 PU 与 CU 的相对形状, 如图 1(c) 所示, 得到最优 PU 预测模式。而最优的 CU 分割尺寸是通过比较当前 CU 最优 PU 预测模式的率失真代价和 4 个子 CU 最优 PU 预测模式的率失真代价之和的大小来实现的, 当前 CU 的率失真代价更小, 则当前 CU 尺寸更优即不分割, 反之则分割。当递归完成后, CTU 的最优分割也随之确定, 如图 1 所示。

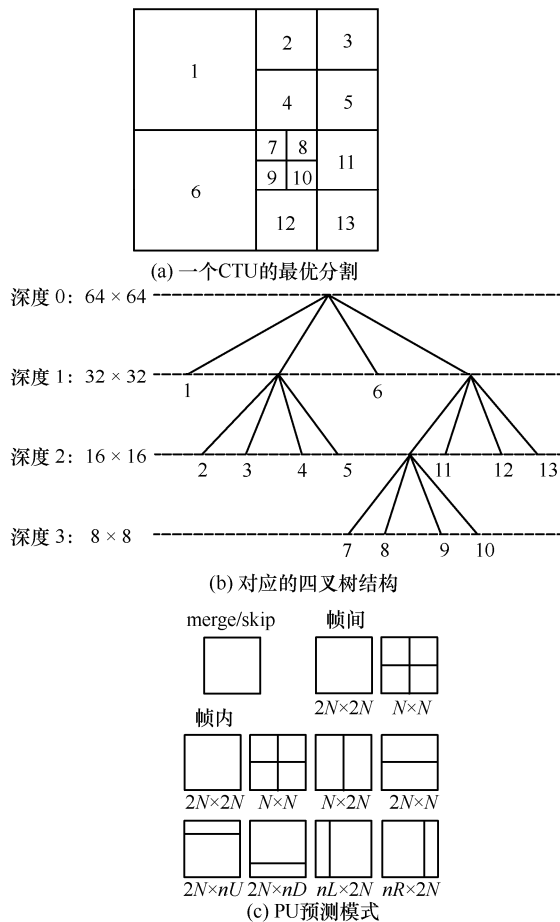


图 1 CTU 最优分割示例和 PU 预测模式

CTU 的最优分割是由深度不一的 CU 组成的集合 $S = \{s_0, s_1, s_2, s_3\}$, 例如 s_1 表示最优分割中深度为 1 的 CU 数量, 则称这些 CU 为最优 CU。最优 CU 的遍历为有效遍历, 而其父 CU 和子 CU 的

遍历过程皆为冗余的 CU 遍历, 即跳过父 CU 或提前终止子 CU 的遍历过程不会损失率失真性能。因此根据视频编码特征在满足控制条件下确定合理的 CU 尺寸是本文算法的研究重点。

3 基于随机森林的 HEVC 复杂度控制方法

3.1 随机森林模型的建立

3.1.1 CU 决策

减少非最优 CU 的遍历过程能减少冗余的编码复杂度, 因此两种用途的分类器被使用在深度 0、1 和 2 的情况下, 且各深度都将单独训练, CU 决策流程如图 2 所示, 图 2 中 $d \in \{0, 1, 2, 3\}$ 表示 CU 深度, $s \in \{0, 1, 2, 3\}$ 表示当前父 CU 的子 CU 序号。skip/merge 模式和帧间 $2N \times 2N$ 模式将先被执行。然后, 在深度不超过 3 时, 分类器#1 (即 skip 分类器) 做出判决。有两种判决结果: (1) 跳过余下 PU 模式; (2) 遍历余下 PU 模式。在 skip 分类器的判决结果为 (2) 的情况下, 分类器#2 (即提前终止 (early termination, ET) 分类器) 才被用来判决 CU 是否提前终止子 CU 的遍历过程。

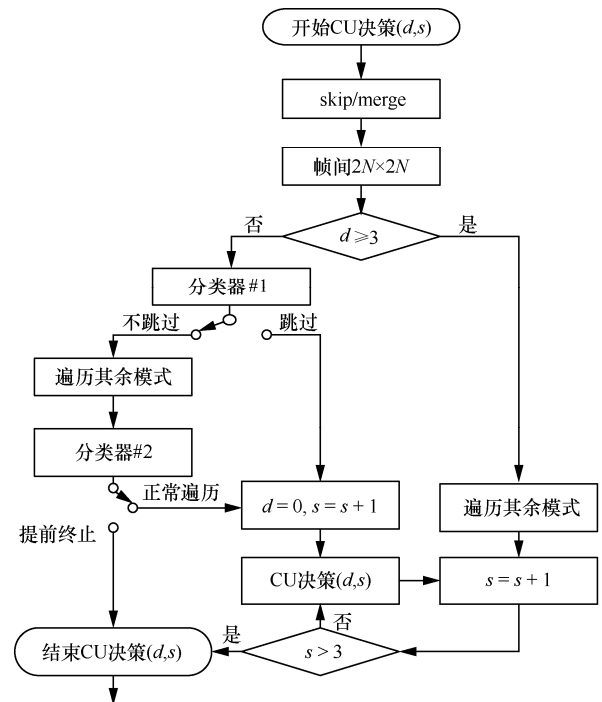


图 2 CU 决策流程

3.1.2 特征选取

好的特征能极大地提升分类器预测准确率和时间节省比例,因此选取合适的特征至关重要。在视频编码中,应选择计算复杂度低的特征,并根据分类器的特点选择合适的特征数量。随机森林分类器具有训练速度快和能较好地处理高维特征的特点,因此从图像本身的特征和编码过程中产生的编码信息中选择了以下 11 个特征用于分类器训练。

(1) 参考列表 0 中位置为 0 参考帧的同位 CU 在最优分割中编码的最大深度与当前 CU 深度的差值,记作 $x_{\Delta\text{Depth}}$ 。

(2) 当前帧与参考列表 0 中位置为 0 参考帧的量化参数 (quantization parameter, QP) 差值,记作 $x_{\Delta\text{QP}}$ 。

(3) CU 原始像素亮度分量的方差,记作 x_{Var} 。

(4) 4 个子 CU 方差的方差,记作 x_{SubCUVar} 。

(5) 编码 skip/merge 模式的率失真代价,记作 x_{RDcost} 。

(6) 编码 skip/merge 的比特,记作 x_{Bits} 。

(7) 编码帧间 $2N \times 2N$ 模式的运动矢量,记作 x_{Mv} 。

(8) skip 标志位,记作 x_{SkipFlag} 。

(9) 已编码块标志位,记作 x_{CBF} 。

(10) 将当前 CU 和同位 CU 亮度分量缩小至 8×8 ,并转为 64 级灰度,对比每个像素值与平均值的大小,大于或等于平均值记为 1,小于平均值记为 0,得到 64 位二进制数。当前 CU 和同位 CU 的二进制数不一致位数为散列值的大小,记为 $x_{\text{HashValue}}$ 。

(11) 当前大小为 $2N \times 2N$ 的 CU 的方向复杂度,记为 x_{DC} ,其计算式为:

$$x_{\text{DC}} = \sum_{i=0}^{2N-1} \sum_{j=0}^{2N-1} G(i, j) = \sum_{i=0}^{2N-1} \sum_{j=0}^{2N-1} \sqrt{G_{\text{Hor}}(i, j)^2 + G_{\text{Ver}}(i, j)^2} \approx \sum_{i=0}^{2N-1} \sum_{j=0}^{2N-1} |G_{\text{Hor}}(i, j)| + |G_{\text{Ver}}(i, j)| \quad (1)$$

其中, $G(i, j)$ 表示坐标为 (i, j) 像素的梯度值, $G_{\text{Hor}}(i, j)$ 和 $G_{\text{Ver}}(i, j)$ 分别为水平和垂直 Sobel 矩阵对 CU 中坐标为 (i, j) 像素的亮度值做卷积得到的水平和垂直方向亮度差分近似值。

3.1.3 随机森林设计

在线学习能使分类器更适应当前视频序列的编码,所以本文采用在线学习的方式来训练分类器。而在线学习的方式必须考虑分类器的训练复杂度,且本文应用于低时延配置,需要选择训练速度快的机器学习方法。再考虑到集成学习得到的模型比单个模型的性能更优,因此本文使用随机森林^[19]来解决模式是否跳过和是否提前终止这两个二分类问题。

随机森林为包含 K 棵决策树的分类器, $T = \{T_k, k = 1, \dots, K\}$, 其分类结果为这 K 棵决策树输出结果的众数。每棵决策树的训练参数一致,但训练集都使用无权重抽样方法^[20]获取。决策树是一种树形结构,其内部节点表示就某一特征属性进行分叉,其叶节点表示分类结果。在决策树的生成过程中,选择最佳的分叉特征属性至关重要。本文采用 CART 决策树,其采用基尼 (Gini) 系数^[21]来比较不同特征属性分叉后结果的优劣,其计算式为:

$$\text{Gini}(D) = 1 - \sum_{i=1}^m P_i^2 = 1 - \sum_{i=1}^m \left(\frac{N_i}{N} \right)^2 \quad (2)$$

其中, D 表示当前内部节点的数据集, N 为 D 的样本数, P_i 表示类别 C_i 在 D 中出现的概率, N_i 表示类别 C_i 的样本数,在二分类问题中 $m=2$ 。

在通过某一特征属性 A 的阈值 θ 进行划分后,数据集被分为两个子集 D_1 和 D_2 , 对应的样本数为 L 和 R , 此次划分的 Gini 系数为:

$$\text{Gini}_{\text{Split}}(D, A^\theta) = \frac{L}{N} \text{Gini}(D_1) + \frac{R}{N} \text{Gini}(D_2) \quad (3)$$

将式 (2) 代入式 (3), 则 Gini 系数简化为:

$$\text{Gini}_{\text{Split}}(D, A^\theta) = 1 - \frac{1}{N} \left(\frac{\sum_{i=1}^m (L_i)^2}{L} + \frac{\sum_{i=1}^m (R_i)^2}{R} \right) \quad (4)$$



其中, L_i 和 R_i 分别表示在 D_1 和 D_2 中类别 C_i 的样本数。

采用在线训练的方式得不到均衡的样本数据集, 而引入类别权重可减少影响。 W_0 和 W_1 分别为 Class 0 和 Class 1 的类别权重, 而考虑到各类自身的样本率, 需进行归一化, 得到归一化后的先验概率 q_i , 即:

$$q_i = \frac{\frac{W_i}{N_i}}{\sum_{i=1}^m \frac{W_i}{N_i}} \quad (5)$$

将式 (5) 代入式 (4), 得:

$$\text{Gini}_{\text{Split}}(D, A^\theta, q) = 1 - \frac{1}{N} \left(\frac{\sum_{i=1}^m (q_i L_i)^2}{\sum_{i=1}^m q_i L_i} + \frac{\sum_{i=1}^m (q_i R_i)^2}{\sum_{i=1}^m q_i R_i} \right) \quad (6)$$

每个内部节点会随机选取固定数量的特征和相应的阈值, 通过式 (6) 进行比较, 最终选择 Gini 系数最小的特征和阈值进行分叉, 将数据集分成两个子集, 直至满足停止条件。本文内部节点的候选特征数量为 3, 停止条件为:

- (1) 当前内部节点数据集的样本个数小于某个给定的数值, 设定为总样本数的 1%;
- (2) 决策树深度大于设定的数值, 设定为 25;
- (3) 当前内部节点数据集的样本属于同一类。

3.1.4 权重选择

合理的权重能减少在线训练数据不平衡问题的影响, 使某一类的预测准确度较高。skip 和 ET 分类器错误判断的影响是一致的, 即标签为 Class 0 被预测为 Class 1, 会损失率失真性能, 标签为 Class 1 被预测为 Class 0, 不会减少编码复杂度。通过定义 Class 0 预测准确率 P_0 和 Class 1 预测准确率 P_1 来衡量率失真损失和编码复杂度节省情况, 计算式为:

$$P_0 = \frac{C_{00}}{C_{00} + C_{01}}, P_1 = \frac{C_{11}}{C_{10} + C_{11}} \quad (7)$$

其中, C_{00} 和 C_{01} 表示 Class 0 被分别预测为 Class 0 和 Class 1 的数量, C_{10} 和 C_{11} 表示 Class 1 被分别预测为 Class 0 和 Class 1 的数量。

调整 W_0 和 W_1 的比值会相应地改变 P_0 和 P_1 , 为找到最佳的权重比值, 本文通过 3 个视频序列 BQMall、Johnny 和 ParkScene 来量化权重比值 (W_0/W_1) 与两种分类器不同 CU 深度间分类准确度的关系。在确定的权重比值下, 本文对 4 个不同 QP (22、27、32 和 37) 分别编码 150 帧并收集相应的样本数据, 其中第 2~13 帧的样本数据用来训练分类器, 其余数据用来验证分类器预测准确率, 在复杂度控制算法中也将第 2~13 帧作为训练帧收集训练用的样本数据。在测试中, W_0/W_1 的范围为 1:10 到 10:1, 步长为 1, 即 {1:10, 1:9, ..., 1:1, 2:1, ..., 9:1, 10:1}。图 3 为 skip 分类器和 ET 分类器在 $d \in \{0, 1, 2\}$ 下 4 个 QP 的平均预测准确度。

由图 3 可知, 各深度下的预测模型均呈现相同的规律, P_0 随 W_0/W_1 的增加而上升, P_1 随 W_0/W_1 的增加而下降。在 CTU 级的复杂度控制中, 需根据编码信息对分类器的预测精度进行调整, 完成控制 CTU 编码复杂度的目的。因此每种模型在每一种深度下都根据 P_0 选择 3 种配置进行训练, 表 1 列出了各深度两种分类器下的 3 种 P_0 及相应的配置。低 P_0 会最大限度地节省编码时间, 高 P_0 则会最大限度地降低率失真损失, 而中 P_0 则是两者的折中。

3.2 复杂度控制方法

3.2.1 平均深度与复杂度的关系

在 CTU 最优分割中, s_2 和 s_3 的值越大代表该 CTU 分割越精细, 因此定义平均深度 MD 来表示 CTU 最优分割的精细程度, 计算式为:

$$\text{MD} = \frac{s_1}{4} + 2 \cdot \frac{s_2}{16} + 3 \cdot \frac{s_3}{64} \quad (8)$$

采用 $P(s|\text{MD})$ 进一步分析 MD 与 CTU 最优分割之间的关系, 如 $P(s_1|\text{MD} \leq 0.5)$ 表示平均深度不大于 0.5 时, 深度为 1 的最优 CU 尺寸之和占

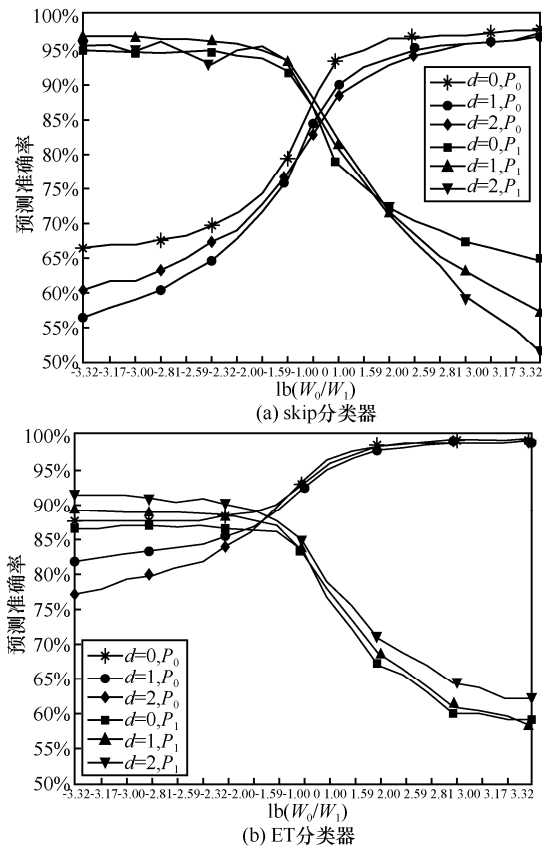


图3 两种分类器的预测准确率

表1 预测准确率及相应权重

深度	skip		ET	
	W_0/W_1	P_0	W_0/W_1	P_0
0	1:10	66.4%	1:10	84.9%
0	2:1	93.1%	1:1	90.7%
0	10:1	97.4%	10:1	98.9%
1	1:10	56.5%	1:10	78.2%
1	3:1	92.3%	1:1	90.3%
1	10:1	96.7%	10:1	98.6%
2	1:10	60.6%	1:10	72.7%
2	3:1	90.9%	1:1	91.5%
2	10:1	97.1%	10:1	99.1%

CTU 尺寸的比率。表 2 展示了 3 个不同序列 $P(s|MD)$ 的平均结果。由表 2 可知, MD 越小, 小深度最优 CU 的占比越大。特别是当 $MD \leq 0.5$ 时, 最优 CU 深度皆为 0。而 $MD > 2.5$ 时, 最优 CU 深度只为 2 和 3。由此可知, MD 值越大, 则 CTU 分割处就越精细, 且能反映当前 CTU 有效遍历的深层层。

表2 概率 $P(s|MD)$ 的平均值

	$P(s MD)$	s_0	s_1	s_2	s_3
BQMall	$MD \leq 0.5$	100	0	0	0
	$0.5 < MD \leq 1.5$	0	80.35	16.78	2.87
	$1.5 < MD \leq 2.5$	0	28.13	48.40	23.48
	$MD > 2.5$	0	0	35.52	64.48
Johnny	$MD \leq 0.5$	100	0	0	0
	$0.5 < MD \leq 1.5$	0	85.14	13.81	1.05
	$1.5 < MD \leq 2.5$	0	38.12	49.37	12.51
	$MD > 2.5$	0	0	37.74	62.26
ParkScene	$MD \leq 0.5$	100	0	0	0
	$0.5 < MD \leq 1.5$	0	81.58	16.51	1.91
	$1.5 < MD \leq 2.5$	0	27.54	51.78	20.67
	$MD > 2.5$	0	0	35.03	64.97

当前 CTU 的 MD 只能在编码完成后获得, 但帧间存在强时域相关性, 因此可用同位 CTU 的 MD_{Col} 来替代。考虑到不同时域层间编码参数的差异, 当前 CTU 的 MD 为:

$$MD = MD_{Col} + \Delta MD \quad (9)$$

其中, ΔMD 表示与当前帧 QP 值相同的训练帧与其前一帧的帧级平均深度差值。通过分析 MD 与 CTU 最优分割之间的关系能为当前 CTU 分配合理的编码复杂度权重 w , 建立 MDC 模型, 即:

$$w = \begin{cases} DP_0, & MD \leq 0.5 \\ DP_1 + DP_2, & 0.5 < MD \leq 1.5 \\ DP_1 + DP_2 + DP_3, & 1.5 < MD \leq 2.5 \\ DP_2 + DP_3, & MD > 2.5 \end{cases} \quad (10)$$

其中, DP_0 、 DP_1 、 DP_2 和 DP_3 分别表示训练帧中所有深度为 0、1、2 和 3 的 CU 编码复杂度之和占训练帧编码复杂度的比例。

3.2.2 段级和图像组级的复杂度控制

图像组 (group of pictures, GOP) 间由于内容的差异, 编码时间可能存在较大差异, 因此设定段作为原始编码时间的预测单位, 段中含 Z 组

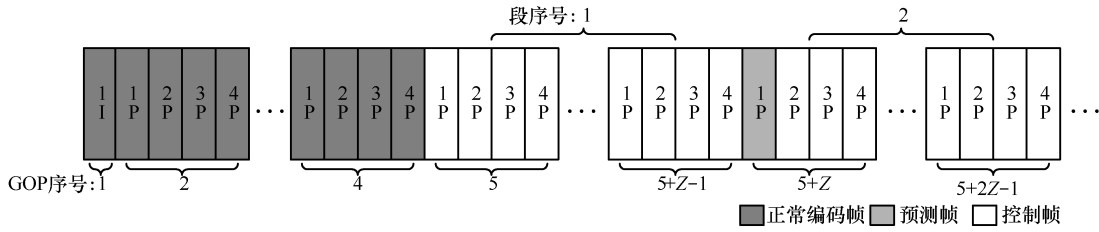


图4 控制结构及帧类型

GOP, 如图4所示。而 T_c 越大, 这种差异对控制精度的影响程度也越大, 为消除这种影响, Z 的取值应越小。

将视频帧分为正常编码帧、预测帧和控制帧, 正常编码帧和预测帧采用原始平台编码, 控制帧用于减少编码复杂度。前13帧均为正常编码帧, 每段中的第1帧为预测帧(除第1段外), 其余均为控制帧。预测帧预测当前段原始编码时间及序列原始编码时间, 即采用原始平台编码所耗时间。第 h 段的原始编码时间 T_h^{Seg} 的计算式为:

$$T_h^{\text{Seg}} = \begin{cases} Z \cdot \frac{t_3^{\text{GOP}} + t_4^{\text{GOP}}}{2}, & h=1 \\ Z \cdot \frac{t_{14+4Z(h-1)}}{\frac{1}{2} \cdot \left(\frac{t_6}{t_3^{\text{GOP}}} + \frac{t_{10}}{t_4^{\text{GOP}}} \right)}, & h>1 \end{cases} \quad (11)$$

其中, t_6 和 t_{10} 分别为第6帧和第10帧的实际编码时间, t_3^{GOP} 和 t_4^{GOP} 分别为第3组和第4组GOP的实际编码时间, Z 的大小由 T_c 决定, 即:

$$Z = \begin{cases} 1, & T_c \geq 0.9 \\ 2, & 0.9 > T_c \geq 0.8 \\ 3, & 0.8 > T_c \geq 0.7 \\ 8, & \text{其他} \end{cases} \quad (12)$$

得到 T_h^{Seg} 后, 可得序列的原始编码时间 T_0 , 即:

$$T_0 = \sum_{f=1}^{13} t_f + \sum_{x=1}^h T_x^{\text{Seg}} + Z \cdot \frac{t_3^{\text{GOP}} + t_4^{\text{GOP}}}{2} \cdot (H-h) \quad (13)$$

其中, t_f 为第 f 帧的实际编码时间, H 为视频序列所含段的数量。根据 T_0 和 T_c 可得第 h 段的目标编码时间 $\bar{T}_h^{\text{Interval}}$, 即:

$$\bar{T}_h^{\text{Seg}} = \frac{T_0 \cdot T_c - \sum_{f=1}^{13} t_f - \sum_{x=1}^{h-1} t_x^{\text{Seg}}}{H-h+1} \quad (14)$$

其中, t_x^{Interval} 为第 x 段的实际编码时间。段内GOP的编码时间差异较小, 因此当前段中第 z 组GOP目标编码时间的计算式为:

$$\bar{T}_z^{\text{GOP}} = \frac{\bar{T}_h^{\text{Seg}} - t_{\text{coded}}^{\text{Seg}}}{Z-z+1} \quad (15)$$

其中, $t_{\text{coded}}^{\text{Seg}}$ 为当前段已编码帧的实际编码时间之和。

3.2.3 帧级和CTU级的复杂度控制

GOP内不同时域帧由于编码参数差异导致编码时间存在差异, 但不同时域帧编码时间与GOP编码时间之比相对稳定, 因此当前GOP中第 r 帧目标编码时间 $\bar{T}_r^{\text{F}}$ 的计算式为:

$$\bar{T}_r^{\text{F}} = \left(\bar{T}_z^{\text{GOP}} - t_{\text{coded}}^{\text{GOP}} \right) \cdot \frac{P_r}{\sum_{r'=r}^4 P_{r'}} \quad (16)$$

其中, $t_{\text{coded}}^{\text{GOP}}$ 为当前GOP已编码帧的实际编码时间之和, P_r 近似地表示第 r 帧编码时间与GOP编码时间之比, 其计算方式为:

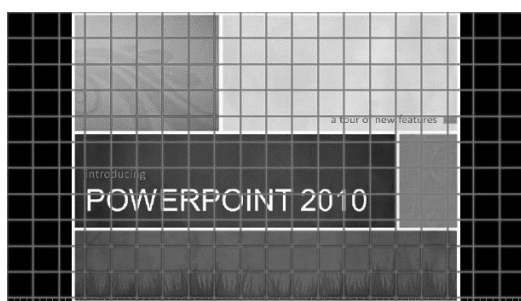
$$P_r = \frac{1}{2} \cdot \frac{t_{5+r}}{\sum_{r'=1}^4 t_{5+r'}} + \frac{1}{2} \cdot \frac{t_{9+r}}{\sum_{r'=1}^4 t_{9+r'}} \quad (17)$$

而第 n 个CTU的目标编码时间 $\bar{T}_n^{\text{CTU}}$ 可根据MDC得到的复杂度权重进行分配:

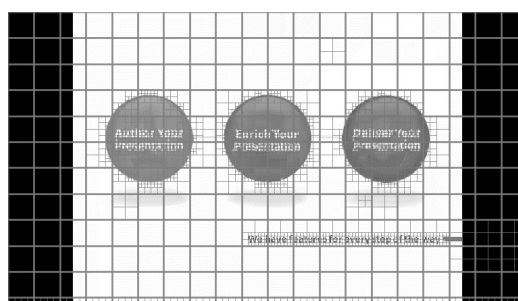
$$\bar{T}_n^{\text{CTU}} = \frac{w_n \cdot \bar{T}_r^{\text{F}}}{\sum_{n'=1}^N w_{n'}} \quad (18)$$

其中, N 表示一帧中 CTU 的数量。MD 能较好地体现帧间变化不强 CTU 的分割情况, 对具有强帧间变化的平坦序列效果较差。最优分割如图 5 所示, 第 27 帧非边界 CTU 中最优 CU 深度均为 0, 而编码该帧仅需 120 bit 数据量。第 28 帧非边界 CTU 中有 57 个 CTU 存在深度为 2 或 3 的最优 CU, 编码这些 CTU 所需的比特数据量占编码该帧所需比特数据量的 95.9%, 而编码该帧需 37 592 bit。由此可见, 平坦序列中编码比特比 MD 更能反映 CTU 的分割情况。因此本文将训练帧中只存在深度为 0 或 1 最优 CU 的 CTU 占比定为平坦度 FS, 并得到训练帧中存在深度为 2 或 3 最优 CU 的 CTU 平均编码比特 MB。将大于 $2/3$ 的 FS 视为平坦序列, 通过深度为 0 的 CU 在最优

模式下编码所需的数据量 Bit、训练帧中 CTU 的平均编码时间 MT 和 CTU 级累计编码误差 e_n 来确定当前 CTU 各深度下预测模型的精度以实现 CTU 级的复杂度控制, 见表 3。 e_n 为前 n 个 CTU 的累计目标编码时间与累计实际编码时间之差。而 FS 不大于 $2/3$ 时, 通过 MT、MD 和 e_n 将确定当前 CTU 各深度下预测模型的精度, 见表 4。表 4 中 skip00、skip01 和 skip02 表示深度为 0 时, 分别采用低 P_0 、中 P_0 和高 P_0 的 skip 分类器进行判决, 同样 ET00、ET01 和 ET02 表示深度为 0 时, 分别采用低 P_0 、中 P_0 和高 P_0 的 ET 分类器进行判决。而 Only0 表示只执行深度为 0 的 CU 遍历, 相应的 Only12 表示只执行深度为 1 和 2 的 CU 遍历。



(a) SlideShow 第 27 帧



(b) SlideShow 第 28 帧

图 5 最优分割

表 3 $FS > \frac{2}{3}$ 时 CTU 的编码配置

编码配置	$e_n < -MT$	$-MT \leq e_n < -\frac{MT}{2}$	$-\frac{MT}{2} \leq e_n \leq \frac{MT}{2}$	$-\frac{MT}{2} < e_n \leq MT$	$e_n > MT$
$Bit \geq 10 \cdot MB$	ET12 ET22	正常编码	正常编码	正常编码	正常编码
$MB \leq Bit < 10 \cdot MB$	ET02 skip12,ET12 skip22,ET22	ET02 ET12 ET22	正常编码	正常编码	正常编码
$0.1 \cdot MB \leq Bit < MB$	ET02 skip12,ET11 skip22,ET21	ET02 ET11 ET21	ET02 ET12 ET21	ET02 ET12 ET22	正常编码
$Bit \leq 0.1 \cdot MB$	Only0	ET00 ET10 ET20	ET02 ET10 ET20	ET02 ET12 ET20	正常编码

表 4 $FS \leq \frac{2}{3}$ 时 CTU 的编码配置

编码配置	$e_n < -MT$	$-MT \leq e_n < -\frac{MT}{2}$	$-\frac{MT}{2} \leq e_n \leq \frac{MT}{2}$	$\frac{MT}{2} < e_n \leq MT$	$e_n > MT$
$MD \leq 0.5$	Only0	skip02,ET01	skip2,ET02	skip02,ET02	正常编码
		skip12,ET10	skip2,ET10	skip12,ET11	
		skip22,ET20	skip2,ET20	skip22,ET20	
$0.5 < MD \leq 1.5$	Only12	skip00,ET02	skip00,ET02	skip01,ET02	正常编码
		skip11,ET11	skip12,ET12	skip12,ET12	
		skip22,ET20	skip22,ET20	skip22,ET22	
$1.5 < MD \leq 2.5$	Only123	skip00,ET02	skip00,ET02	skip00,ET02	正常编码
		skip10,ET11	skip12,ET11	skip12,ET12	
		skip21,ET20	skip22,ET21	skip22,ET22	
$MD > 2.5$	Only23	skip00,ET02	skip00,ET02	skip00,ET02	正常编码
		skip21,ET20	skip11,ET12	skip11,ET12	
		skip22,ET20	skip22,ET21	skip22,ET22	

3.3 总体算法

基于随机森林的 HEVC 复杂度控制方法具体步骤如下。

步骤 1 前 13 帧采用原始平台编码, 其中第 2~13 帧收集各深度样本数据, 按深度 (0、1、2) 和预测准确率 (低、中、高) 共训练 9 个 skip 分类器和 9 个 ET 分类器。确定 T_c 、FS、MB、H、Z 和总帧数 F , 并令 $h=0$, $z=0$, $r=0$, $f=0$ 。

步骤 2 令 $h=h+1$, 并根据式 (14) 更新 $\bar{T}_h^{\text{Seg}}$ 。

步骤 3 令 $z=z+1$, 若 $z>Z$, 则令 $z=0$, 并转步骤 2。根据式 (15) 更新 $\bar{T}_z^{\text{GOP}}$ 。

步骤 4 令 $r=r+1$, $f=f+1$, 若 $f>F$, 结束编码。若 $r>4$, 则令 $r=0$, 并转步骤 3。根据式 (16) 更新 $\bar{T}_r^{\text{F}}$ 。

步骤 5 令 $n=n+1$, 若 $n>N$, 则令 $n=0$, 并转步骤 4。更新 e_n 和 MD, 根据式 (18) 更新 $\bar{T}_n^{\text{CTU}}$, 平坦度大于 $2/3$ 按表 3 相应编码配置进行编码, 否则按表 4 相应编码配置进行编码, 转步骤 5。

4 实验结果及分析

本文算法在 HEVC 软件参考平台 HM16.0 上实

现并验证, 且与顶尖的 HEVC 复杂度控制算法^[16]进行了对比。本文采用的测试条件遵循公共测试条件^[22]所给建议, 使用 Low-delay P 配置编码, 共测试了 B 类、E 类、C 类和 F 类共 4 类序列, 实验结果由均分 4 个 QP (22, 27, 32 和 37) 时的数据得到。本文使用 R_c 表示实际编码时间与原始平台实际编码时间之比, 以衡量与 T_c 的偏离程度。并采用码率变化率 ΔBR 和峰值信噪比变化值 $\Delta PSNR$ 衡量复杂度控制算法的率失真性能。本文算法对 T_c 分别为 80%、60%和 40%的情况进行了测试分析。

表 5 展示了本文算法在 T_c 分别为 80%、60%和 40%时的编码性能。通过平均 R_c 能反映本文算法总体的控制精度, 而本文算法在 T_c 分别为 80%、60%和 40%时的平均 R_c 分别为 80.58%、60.73%和 40.68%, 偏差均在 1%以下, 表明本文算法总体控制精度较高。表 6 和表 7 展示了 T_c 分别为 80%、60%和 40%时两种算法的性能对比。从表 6 和表 7 可知, 本文算法仅在 T_c 为 80%的情况下, 平均控制精度稍逊于参考文献^[16]算法。

表5 本文算法在不同目标复杂度下的编码性能

序列 (类)	$T_c = 80\%$			$T_c = 60\%$			$T_c = 40\%$		
	R_c	$\Delta PSNR/dB$	ΔBR	R_c	$\Delta PSNR/dB$	ΔBR	R_c	$\Delta PSNR/dB$	ΔBR
BasketballDrive(B)	81.01%	-0.01	0.09%	61.71%	-0.04%	0.25	38.77%	-0.10	2.82%
BQTerrace (B)	80.09%	-0.01	-0.56%	60.37%	-0.03%	-0.94	40.95%	-0.06	-0.33%
ParkScene (B)	81.45%	-0.01	-0.02%	62.17%	-0.04%	0.07	40.66%	-0.08	1.16%
Johnny (E)	79.97%	-0.01	-0.50%	60.54%	-0.02%	-0.63	40.65%	-0.05	0.25%
FourPeople (E)	79.46%	-0.02	-0.40%	58.66%	-0.03%	-0.62	39.57%	-0.07	1.45%
Vidyo1 (E)	79.67%	-0.01	-0.23%	59.56%	-0.02%	-0.40	40.33%	-0.06	0.71%
Vidyo3 (E)	80.59%	-0.01	0.09%	60.72%	-0.02%	0.19	41.64%	-0.07	1.66%
Vidyo4 (E)	80.08%	-0.01	0.04%	60.36%	-0.02%	0.07	41.69%	-0.06	1.32%
PartyScene (C)	82.20%	-0.01	0.42%	61.31%	-0.04%	0.83	40.72%	-0.08	3.79%
RaceHorsesC (C)	82.62%	0.00	0.31%	62.16%	-0.03%	0.71	41.54%	-0.09	4.51%
BasketballDrill (C)	81.62%	-0.01	0.18%	60.77%	-0.03%	0.54	41.27%	-0.10	4.15%
SlideEditing (F)	78.69%	-0.02	0.20%	60.38%	-0.02%	-0.06	40.20%	-0.04	0.32%
SlideShow (F)	80.05%	-0.03	-0.04%	60.81%	-0.05%	0.33	40.90%	-0.22	2.65%
平均值	80.58%	-0.01	-0.03%	60.73%	-0.03%	0.03	40.68%	-0.08	1.88%

表6 $T_c=80\%$ 和 $T_c=60\%$ 时本文算法与参考文献[16]算法的性能对比

序列 (类)	$T_c = 80\%$						$T_c = 60\%$					
	本文算法			参考文献[16]算法			本文算法			参考文献[16]算法		
	R_c	$\Delta PSNR/dB$	ΔBR	R_c	$\Delta PSNR/dB$	ΔBR	R_c	$\Delta PSNR/dB$	ΔBR	R_c	$\Delta PSNR/dB$	ΔBR
BasketballDrive (B)	81.01%	-0.01	0.09%	78.58%	-0.03	0.33%	61.71%	-0.04	0.25%	60.24%	-0.15	2.02%
BQTerrace (B)	80.09%	-0.01	-0.56%	79.75%	-0.02	0.33%	60.37%	-0.03	-0.94%	61.90%	-0.08	3.20%
ParkScene (B)	81.45%	-0.01	-0.02%	80.00%	-0.03	0.60%	62.17%	-0.04	0.07%	61.27%	-0.14	5.87%
Johnny (E)	79.97%	-0.01	-0.50%	80.84%	0.00	-0.05%	60.54%	-0.02	-0.63%	59.10%	0.00	-0.02%
FourPeople (E)	79.46%	-0.02	-0.40%	80.93%	0.00	0.13%	58.66%	-0.03	-0.62%	60.23%	-0.01	0.44%
Vidyo1 (E)	79.67%	-0.01	-0.23%	81.98%	0.00	0.03%	59.56%	-0.02	-0.40%	62.19%	-0.02	0.62%
Vidyo3 (E)	80.59%	-0.01	0.09%	81.11%	0.00	0.04%	60.72%	-0.02	0.19%	63.36%	-0.05	1.55%
Vidyo4 (E)	80.08%	-0.01	0.04%	82.28%	0.00	0.15%	60.36%	-0.02	0.07%	61.01%	-0.01	0.21%
PartyScene (C)	82.20%	-0.01	0.42%	78.73%	-0.10	2.53%	61.31%	-0.04	0.83%	60.50%	-0.25	8.29%
RaceHorsesC (C)	82.62%	0.00	0.31%	78.31%	-0.08	1.56%	62.16%	-0.03	0.71%	58.13%	-0.45	7.37%
BasketballDrill (C)	81.62%	-0.01	0.18%	82.75%	-0.02	0.46%	60.77%	-0.03	0.54%	62.71%	-0.18	4.85%
SlideEditing (F)	78.69%	-0.02	0.20%	81.53%	-0.03	0.42%	60.38%	-0.02	-0.06%	61.06%	-0.30	3.32%
SlideShow (F)	80.05%	-0.03	-0.04%	80.18%	-0.03	0.40%	60.81%	-0.05	0.33%	62.00%	-0.06	0.58%
平均值	80.58%	-0.01	-0.03%	80.54%	-0.02	0.53%	60.73%	-0.03	0.03%	61.05%	-0.13	2.95%

表 7 $T_c=40\%$ 时本文算法与参考文献[16]算法的性能对比

序列 (类)	本文算法			参考文献[16]算法		
	R_c	$\Delta\text{PSNR}/\text{dB}$	ΔBR	R_c	$\Delta\text{PSNR}/\text{dB}$	ΔBR
BasketballDrive (B)	38.77%	-0.10	2.82%	40.37%	-0.28	6.03%
BQTerrace (B)	40.95%	-0.06	-0.33%	41.62%	-0.31	6.59%
ParkScene (B)	40.66%	-0.08	1.16%	40.02%	-0.34	8.24%
Johnny (E)	40.65%	-0.05	0.25%	41.74%	-0.06	2.21%
FourPeople (E)	39.57%	-0.07	1.45%	41.10%	-0.24	6.50%
Vidyo1 (E)	40.33%	-0.06	0.71%	41.20%	-0.05	1.89%
Vidyo3 (E)	41.64%	-0.07	1.66%	38.37%	-0.11	3.67%
Vidyo4 (E)	41.69%	-0.06	1.32%	40.02%	-0.10	3.77%
PartyScene (C)	40.72%	-0.08	3.79%	41.42%	-1.03	13.24%
RaceHorsesC (C)	41.54%	-0.09	4.51%	37.55%	-1.34	11.12%
BasketballDrill (C)	41.27%	-0.10	4.15%	42.72%	-0.27	6.36%
SlideEditing (F)	40.20%	-0.04	0.32%	41.65%	-0.87	6.54%
SlideShow (F)	40.90%	-0.22	2.65%	41.12%	-0.12	1.83%
平均值	40.68%	-0.08	1.88%	40.68%	-0.39	6.00%

ΔPSNR 和 ΔBR 能反映本文算法的率失真性能, 在 T_c 分别为 80%、60% 和 40% 时的平均 ΔPSNR 分别为 -0.01 dB、-0.03 dB 和 -0.08 dB, 平均 ΔBR 分别为 -0.03%、0.03% 和 1.88%。这意味着对 T_c 为 80% 和 60% 的情况, 本文算法的率失真性能几乎没有损失。而纹理丰富或运动剧烈的视频序列, 如 RaceHorsesC, 与平坦视频序列的率失真性能相比没有较大差异。而 T_c 为 40% 时的 ΔBR 与 T_c 为 80% 和 60% 时的 ΔBR 相差较大, 是因为节省的复杂度已经接近本文算法所能控制的极限, 此时率失真性能损失呈现随 T_c 下降而指数上升趋势。在 ΔPSNR 和 ΔBR 方面, 本文算法均优于参考文献[16]。这是因为本文算法考虑了 CTU 的最优分割和整个视频序列的编码情况, 为 CTU 分配了合理的编码配置。而参考文献[16]仅依据帧级中所有 CTU 的相对重要性, 为 CTU 分配最大分割深度, 而忽视 CTU 本身的分割情况。参考文献[16]对于平坦序

列, 如 Johnny, 在 T_c 较大的情况下会有较好的率失真性能, 而对运动剧烈的视频序列, 如 RaceHorsesC, 会带来较大的率失真性能损失。这是因为平坦序列的 CTU 大多只采用深度为 0 的 CU 进行编码, 对这些 CTU 而言限制最大分割深度不会带来率失真性能的损失。而运动剧烈视频序列的 CTU 较多采用了深度为 2 和 3 的 CU 进行编码, 对这些 CTU 限制最大分割深度就会带来率失真性能的损失。正是因为本文较好地保证了视频训练的率失真性能, 仅能将编码复杂度控制至 40%, 而参考文献[16]则能控制至 20%。

5 结束语

本文提出了一种基于随机森林的编码复杂度控制方法。通过段、GOP、帧和 CTU 逐级地完成编码复杂度的分配和控制。其中, 段级通过预测原始编码时间以确定段级的目标复杂度, GOP 级和

帧级完成编码误差的重新分配, CTU 级选择编码配置来减少编码复杂度。为了实现 CTU 级编码复杂度的有效控制, 通过随机森林建立不同预测准确率的分类器以提供多种编码配置, MDC 模型为 CTU 分配合适的目标编码时间, 而 FS、MD、Bit 和 e_n 则确定了 CTU 最终的编码配置。实现结果表明, 本文算法具有较高的控制精度, 平均误差不超过 1%。而相比当前的编码复杂度控制方法, 本文算法在 Δ PSNR 和 Δ BR 两方面均具有明显的优势。

参考文献:

- [1] 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.
- [2] WIEGAND T, SULLIVAN G J, BJØNTEGAARD 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.
- [3] SHAN Y, YANG E H. Fast HEVC intra coding algorithm based on machine learning and Laplacian transparent composite model[C]//IEEE International Conference on Acoustics, Speech and Signal Processing, March 5-9, 2017, New Orleans, LA, USA. Piscataway: IEEE Press, 2017: 2642-2646.
- [4] ZHU L, ZHANG Y, PAN Z, et al. Binary and multi-class learning based low complexity optimization for HEVC encoding[J]. IEEE Transactions on Broadcasting, 2017, 63(3): 547-561.
- [5] TAN T K, WEERAKKODY R, MRAK M, et al. Video quality evaluation methodology and verification testing of HEVC compression performance[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2016, 26(1): 76-90.
- [6] ZHANG T, SUN M T, ZHAO D, et al. Fast intra mode and CU size decision for HEVC[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2017, 27(8): 1714-1726.
- [7] MALLIKARACHCHI T, TALAGALA D S, ARACHCHI H K, et al. Content-adaptive feature-based CU size prediction for fast low-delay video encoding in HEVC[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2018, 28(3): 693-705.
- [8] TSAI T H, SU S S, LEE T Y. Fast mode decision method based on edge feature for HEVC inter prediction[J]. IET Image Processing, 2018, 12(5): 644-651.
- [9] 姚英彪, 李晓娟. 基于图像空间相关性与纹理的 HEVC 块划分快速算法[J]. 电信科学, 2015, 31(1): 32-40.
- YAO Y B, LI X J. Fast algorithm for HEVC block partitioning based on image spatial correlation and texture[J]. Telecommunications Science, 2015, 31(1): 32-40.
- [10] CORREA G, ASSUNCAO P, CRUZ L A D S, et al. Computational complexity control for HEVC based on coding tree spatio-temporal correlation[C]//IEEE 20th International Conference on Electronics, Circuits, and Systems, Dec 8-11, 2014, Abu Dhabi, United Arab Emirates. Piscataway: IEEE Press, 2014: 937-940.
- [11] CORREA G, ASSUNCAO P, AGOSTINI L, et al. Coding tree depth estimation for complexity reduction of HEVC[C]//Data Compression Conference, March 20-22, 2013, Snowbird, UT, USA. Piscataway: IEEE Press, 2013: 43-52.
- [12] CORREA G, ASSUNCAO P, AGOSTINI L, et al. Complexity scalability for real-time HEVC encoders[J]. Journal of Real-Time Image Processing, 2016, 12(1): 107-122.
- [13] CORREA G, ASSUNCAO P, CRUZ L A D S, et al. Encoding time control system for HEVC based on rate-distortion-complexity analysis[C]//IEEE International Symposium on Circuits and Systems, May 24-27, 2015, Lisbon, Portugal. Piscataway: IEEE Press, 2015: 1114-1117.
- [14] JIMÉNEZ-MORENO A, MARTÍNEZ-ENRÍQUEZ E, DÍAZ-DE-MARÍA F. Complexity control based on a fast coding unit decision method in the HEVC video coding standard[J]. IEEE Transactions on Multimedia, 2016, 18(4): 563-575.
- [15] ZHANG J, KWONG S, ZHAO T, et al. CTU-level complexity control for high efficiency video coding[J]. IEEE Transactions on Multimedia, 2018, 20(1): 29-44.
- [16] DENG X, XU M, JIANG L, et al. Subjective-driven complexity control approach for HEVC[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2016, 26(1): 91-106.
- [17] DENG X, XU M, LI C. Hierarchical complexity control of HEVC for live video encoding[J]. IEEE Access, 2016, 4(99): 7014-7027.
- [18] 徐博文, 王晓东, 郭磊, 等. 基于幻方变换的 HEVC 熵编码加密方案[J]. 电信科学, 2018, 34(3): 95-104.
- XU B W, WANG X D, GUO L, et al. A video encryption in HEVC entropy coding based on magic square [J]. Telecommunications Science, 2018, 34(3): 95-104.
- [19] BREIMAN L. Random forests[J]. Machine Learning, 2001, 45(1): 5-32.
- [20] BREIMAN L. Bagging predictors [J]. Machine Learning, 1996, 24(2): 123-140.
- [21] BREIMAN L I, FRIEDMAN J H, OLSHEN R A, et al. Classification and regression trees (CART)[J]. Encyclopedia of Ecology, 1984, 40(3): 582-588.



- [22] BOSSEN F. Common test conditions and software reference configurations[C]//Document JCTVC-F900, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11 6th Meeting, July 14-22, 2011, Torino, Italy. [S.l.: s.n.], 2011.



陈芬（1973- ），女，博士，宁波大学信息科学与工程学院教授，主要研究方向为三维视频信号处理与编码。

[作者简介]



文鹏（1994- ），男，宁波大学信息科学与工程学院硕士生，主要研究方向为三维视频信号编码。



蒋刚毅（1964- ），男，博士，宁波大学信息科学与工程学院教授，主要研究方向为三维视频信号处理与编码。



彭宗举（1973- ），男，博士，宁波大学信息科学与工程学院教授，主要研究方向为三维视频信号处理与编码。



郁梅（1968- ），女，博士，宁波大学信息科学与工程学院教授，主要研究方向为三维视频信号处理与编码。