doi:10.11959/j.issn.1000-436x.2016073

基于时空相关性的 HEVC 帧间模式决策快速算法

朱威[1]，张训华[1]，王财盛[1]，张桦[2]
（1. 浙江工业大学信息工程学院，浙江 杭州 310023；
2. 杭州电子科技大学计算机学院，浙江 杭州 310018）

摘 要：新一代的高效率视频编码标准 HEVC 采用编码树单元（CTU）四叉树划分技术和多达 10 种的帧间预测单元（PU）模式，有效地提高了编码压缩效率，但也极大地增加了编码计算复杂度。为了减少编码单元（CU）的划分次数和候选帧间 PU 模式个数，提出了一种基于时空相关性的帧间模式决策快速算法。首先，利用当前 CTU 与参考帧中相同位置 CTU、当前帧中相邻 CTU 的深度信息时空相关性，有效预测当前 CTU 的深度范围。然后，通过分析当前 CU 与其父 CU 之间的最佳 PU 模式空间相关性，以及利用当前 CU 已估计 PU 模式的率失真代价，跳过当前 CU 的冗余帧间 PU 模式。实验结果表明，提出的算法与 HEVC 测试模型（HM）相比，在不同编码配置下降低了 52%左右的编码时间，同时保持了良好的编码率失真性能；与打开快速算法选项的 HM 相比，所提算法进一步降低了 30%左右的编码时间。

关键词：高效率视频编码；编码树单元；预测单元；时空相关性
中图分类号：TN919.8 **文献标识码**：A

Spatio-temporal correlation based fast inter mode decision for HEVC

ZHU Wei[1], ZHANG Xun-hua[1], WANG Cai-sheng[1], ZHANG Hua[2]
(1. College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China;
2. School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China)

Abstract: The new generation high efficiency video coding (HEVC) standard adopts coding tree unit (CTU) quadtree partition technology and ten inter prediction unit (PU) modes to improve compression efficiency significantly, but it brings in an enormous encoding computational complexity. A spatio-temporal correlation based fast inter mode decision algorithm for HEVC was proposed to reduce partition times of coding unit (CU) and inter PU mode candidates. First, the spatio-temporal correlation of depth information among the current CTU, the same located CTU in the reference frame and the spatial neighbor CTU in the current frame are utilized to predict depth range of the current CTU. Then, the spatial correlation of optimal PU modes between the current CU and its parent-CU was analyzed. And the rate-distortion (RD) costs of the estimated PU modes of the current CU were also used. Based on these, the redundant inter PU modes of the current CU are skipped. The experiment results demonstrate that as compared to the HEVC test model (HM), the proposed algorithm saves about 52% encoding time under different encoding configurations, and it maintains a good RD performance. When compared to HM with fast mode decision algorithms enabled, the proposed algorithm further saves about 30% encoding time.

Key words: high efficiency video coding, coding tree unit, prediction unit, spatio-temporal correlation

收稿日期：2015-05-29；**修回日期**：2015-12-15
基金项目：国家自然科学基金资助项目（No.61401398, No.61471150）；浙江省重点科技创新团队子项基金资助项目（No.2011R50009-4）
Foundation Items: The National Natural Science Foundation of China (No.61401398, No.61471150), Sub-Foundation of Zhejiang Provincial Key Innovation Team (No.2011R50009-4)

1　引言

随着视频会议、视频监控和数字电视等视频多媒体应用的快速发展，高清视频（720P 和 1080P）逐渐普及，超高清视频（4K 和 8K）相继出现，人们对更高分辨率和更高帧率的视频需求不断提升。目前广泛应用的视频编码标准 H.264 已经难以满足高清和超高清视频存储及传输的发展需求，为进一步获得更高效的视频编码效率，国际组织 ITU-T 的视频编码专家组和 ISO/IEC 的运动图像专家组于 2013 年联合制定了新一代高效率视频编码标准 HEVC。与前一代的视频编码标准 H.264 相比，HEVC 在相同视频质量的条件下可以节省 50%左右的视频码流[1]，但其编码计算复杂度也成倍增加，这严重阻碍了它在工业领域和民用领域的快速应用。HEVC 采用许多改进编码技术，包括编码树单元（CTU, coding tree unit）四叉树划分[2]、非对称帧间预测单元（PU, prediction unit）模式和多方向帧内 PU 模式等。其中，CTU 四叉树划分技术使 HEVC 的编码单元（CU, coding unit）划分尺寸变得灵活多样，且每个 CU 还需要从多种帧间和帧内 PU 模式中选取最佳 PU 模式，因此整个模式决策过程是 HEVC 编码计算复杂度最为集中的部分。

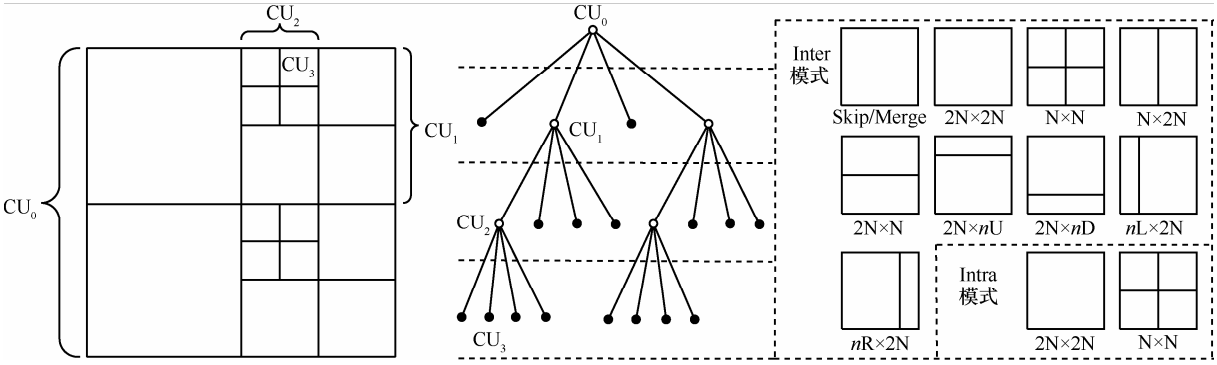
为了降低 HEVC 编码的计算复杂度，目前已经有一些研究人员对其模式决策快速算法展开研究。针对帧间编码的模式决策，文献[3]利用前一帧中相同位置 CU 与其左方 CU 的深度相关性计算当前 CU 的复杂度类型，以缩小深度搜索范围。该算法虽然可以较好地保持编码率失真性能，但降低的计算复杂度比较有限，还有进一步提升的空间。文献[4]利用相邻 CTU 的运动矢量提前终止当前 CU 的划分，但该方法在不同测试序列之间的计算复杂度降低差异较大，编码率失真性能损失也较多。文献[5]提出一种基于纹理和运动复杂度的 CU 划分决策算法。文献[6]利用时空相邻 CTU 的深度遍历区间类型预测 CU 深度遍历区间，并且采用贝叶斯决策原理对 CU 进行早期裁剪。文献[7]先利用时空相邻 CU 的深度加权计算当前 CU 的深度预测值，再通过对该值进行分类讨论得到 CU 的深度范围，并且基于运动一致性、率失真代价和 Skip 模式对 CU 划分进行提前终止判别。针对帧内编码的模式决策，文献[8]将图像特征和 CU 划分结合起来，首先用各个深度的边缘点阈值来缩小 CU 的深度范围，然后基于率失真代价提前终止 CU 的划分。文献[9]通过分析 CU 的纹理一致性，结合空间相邻 CU 的深度信息，有效减少了 CU 的划分次数。

虽然以上快速算法已使用视频的时间和空间相关性来提高算法性能，但 CTU 深度信息的时空相关性、当前 CU 与其父 CU 的空间相关性以及当前 CU 已估计 PU 模式的率失真代价还未充分挖掘利用。为了在保持编码率失真性能的条件下进一步降低编码计算复杂度，本文提出了一种基于时空相关性的帧间模式决策快速算法。首先根据当前 CTU 与相邻 CTU 的深度信息时空相关性，利用相邻 CTU 的最小深度值和最大深度值，预测当前 CTU 的整体深度范围，以减少 CU 的划分次数；接着根据当前 CU 与其父 CU 的最佳 PU 模式空间相关性和当前 CU 已估计 PU 模式的率失真代价，跳过当前 CU 的部分冗余帧间 PU 模式。

2　CTU 四叉树划分和帧间 PU 模式的统计分析

在 HEVC 编码过程中，一帧图像先被划分成多个 CTU，每个 CTU 可采用四叉树划分技术再被逐层划分为一个或多个 CU。与 H.264 中的尺寸固定为 16×16 的编码宏块相比，CTU 的四叉树划分技术使 CU 的划分尺寸种类增加，也使帧间和帧内编码更具适应性和多样性，显著地提升了视频压缩效率。当 CTU 的大小为 64×64 时，CU 的大小可以为 64×64、32×32、16×16 和 8×8，对应未划分 CU 的深度分别为 0、1、2 和 3。如图 1(a)所示，CTU 的大小为 64×64，将尺寸为 64×64 的 CU 记作 CU_0，CU_0 被划分成 4 个尺寸相同的子 CU，其每个子 CU 还可以被继续划分。CU_3 是 CU_2 被划分后的一个子 CU，但它已处在最大深度，不能被继续划分。将 CU_0 称为 CU_1 的父 CU，将 CU_2 称为 CU_3 的父 CU。在 HEVC 测试模型（HM, HEVC test model）中，CTU 最终划分的深度信息以 4×4 块为单位进行存储，CU_3 覆盖的 4 个 4×4 块所处的深度值都为 3，则 CU_3 的深度为 3，同理，CU_2 的深度为 3，CU_1 和 CU_0 的深度范围分别为[2, 3]和[1, 3]，整个 CTU 的深度为[1, 3]。如果 CTU 不进行四叉树划分，只是单个 CU，则该 CTU 的深度为 0。图 1(b)是图 1(a)中相应 CTU 的四叉树划分结构，白色圆点表示 CU 继续划分，黑色圆点表示 CU 终止划分。

HEVC 在 CU 的基础上，以 PU 为单位依次进行帧间和帧内估计，组成每个 CU 的 PU 个数可以



(a) CTU 划分　　(b) CTU 四叉树划分结构　　(c) PU 模式

图 1　CTU 四叉树划分和 PU 模式

是 1、2 或 4 个，如图 1(c)所示。CU 的候选 PU 模式分两大类：Inter 模式和 Intra 模式。其中 Inter 模式包括 Skip、Merge、2N×2N、N×N、N×2N、2N×N、2N×*n*U、2N×*n*D、*n*L×2N 和 *n*R×2N 等 10 种模式，HEVC 按照此顺序依次进行帧间估计，最后的 4 种模式由于采用非对称运动划分（AMP, asymmetric motion partitioning）被统称为 Inter AMP 模式。Merge 模式是一种新的 HEVC 预测模式，它从相邻的 PU 获得运动信息，这些相邻 PU 组成分享共同运动信息的区域。Skip 模式是一种 Merge 模式的特殊情况[1]，其残差为 0，不需传输运动信息，只传输相关标识位和索引到解码器。为了减少帧间估计的计算复杂度，HEVC 限制亮度像素使用 4×4、8×2 和 2×8 等小尺寸 PU 划分，因此 Inter N×N 模式和 Inter AMP 模式在 CU 大小为 8×8 时都被禁止使用。为了避免与 CU 划分的功能冲突，只有在 CU 为最小尺寸且最小尺寸大于 8×8 的情况下，才允许使用 Inter N×N 模式。由于 CTU 采用四叉树划分技术，HM 采用全搜索的模式决策算法（简称 HM 参考算法），在每个 CTU 的编码过程中递归遍历所有深度，并且每次递归都需计算出各个候选 PU 模式的率失真代价，以获得最佳 CTU 划分和最佳 PU 模式。这个模式决策过程可以获得很好的编码率失真性能，但却消耗了 HEVC 编码大部分的处理时间。

为了降低 HEVC 模式决策的计算复杂度，需要统计分析 CTU 四叉树划分和帧间 PU 模式的特点，本节选取 5 个不同分辨率、不同运动量和纹理特征的测试序列（TS, test sequence）。PeopleOnStreet (2 560×1 600, TS0)序列是许多人在街上行走的场景；ParkScene(1920×1080, TS1)序列镜头移动，有一些物体运动；BlowingBubbles(416×240, TS2)序列的运动量较大且纹理较复杂；BasketballDrill(832×480, TS3)序列中物体运动剧烈；Johnny(1 280×720, TS4)序列的运动量小且纹理简单。软件实现平台为 HM-8.0，量化参数（QP, quantization parameter）取 32，编码配置为 LDP(low delay P)，每个序列编码 100 帧。

本节将 CTU 的最大深度值标记为 *Depth*，其值可能为 0、1、2 或 3。为了分析 CTU 最终划分的深度信息与测试序列特征之间的关系，表 1 给出了各个序列的 *Depth* 分布情况。从表中可以得出，不同的 *Depth* 值所占的比例不同。TS2 序列有 89.5% CTU 的 *Depth* 大于 0，而 TS4 序列有 75.9% CTU 的 *Depth* 为 0。这表明运动量较大且纹理较复杂序列的 CTU 划分深度相对较大，运动量较小且纹理较简单序列的 CTU 划分深度相对较小。对于 TS0 序列，有 63.9% CTU 的 *Depth* 为 3，而仅有 7.1% CTU 的 *Depth* 为 1。对于 TS1 和 TS3 序列，则分别有 33.5% 和 34.7% CTU 的 *Depth* 为 3。这说明 CTU 最终划分深度值的分布不固定，与测试序列自身特征相关。由于 *Depth* 分布不均匀，HEVC 模式决策算法没有必要对每个 CTU 遍历所有深度。因此，如果能在每个 CTU 进行划分前预测其深度范围，就可以有效减少四叉树划分的计算复杂度。

表 1　　CTU 的 *Depth* 分布

序列	CTU 的 *Depth* 比例/%			
	0	1	2	3
TS0	16.8	7.1	12.1	63.9
TS1	45.6	10.3	10.5	33.5
TS2	10.5	15.0	23.9	50.6
TS3	38.5	15.3	11.5	34.7
TS4	75.9	7.5	12.8	3.8
平均值	37.5	11.1	14.2	37.3

为了分析帧间 PU 模式与测试序列特征之间的关系，表 2 给出了各个序列中不同深度 CU 的最佳 PU 模式比例。从表中可以看出，测试序列的纹理信息和运动量不仅影响 CTU 的最终划分深度，而且还影响 CU 的最佳 PU 模式选择。各个序列的最佳 PU 模式分布很不均衡，平均有 73%以上的 CU 选择将 Skip、Merge 或 Inter 2N×2N 作为最佳 PU 模式，选择 Inter N×2N、Inter 2N×N、Inter AMP 和其他模式的平均比例均不足 10%。其中，纹理复杂且运动量中等的 TS1 序列有 55.8%~96.0%的 CU 选择 Skip、Merge 或 Inter 2N×2N 为最佳 PU 模式。运动量较小且纹理简单的 TS4 序列有 83.9%~98.7%的 CU 选择 Skip、Merge 或 Inter 2N×2N 为最佳 PU 模式，而选择 Inter N×2N、Inter 2N×N 或 Inter AMP 的比例仅为 0.3%~15.1%。这说明对运动越小且纹理越简单的序列，其 CU 选择 Skip、Merge 或 Inter 2N×2N 为最佳 PU 模式的可能性较大，而选择 Inter N×2N、Inter 2N×N 或 Inter AMP 为最佳 PU 模式的可能性较小。因此，模式决策算法可以根据最佳 PU 模式分布不均衡的特征，针对每个 CU 减少其候选帧间 PU 模式的个数，以降低 PU 模式选择的计算复杂度。

3　帧间模式决策快速算法

为了降低 HEVC 模式决策过程中 CTU 四叉树划分和 PU 模式选择的计算复杂度，本文根据相邻 CTU 的深度信息时空相关性、父 CU 与子 CU 的最佳 PU 模式空间相关性和当前 CU 的已估计 PU 模式的率失真代价，提出了一种基于时空相关性的帧间模式决策快速算法。该算法包括 CTU 深度范围预测（CDRP, CTU depth rang prediction）算法和帧间 PU 模式选择（IPMS, inter PU mode selection）算法。其中，CDRP 算法利用 3 个时空相邻 CTU 的最小和最大深度值，预测当前 CTU 的深度范围；IPMS 算法利用父 CU 的最佳 PU 模式和当前 CU 已估计 PU 模式的率失真代价，减少当前 CU 的候选帧间 PU 模式个数。

表 2　　**不同深度 CU 的最佳 PU 模式比例**

序列	深度	最佳 PU 模式比例/%						
		Skip	Merge	Inter 2N×2N	Inter N×2N	Inter 2N×N	Inter AMP	其他
TS0	0	14.1	4.5	7.2	27.6	28.0	13.4	5.2
	1	31.3	6.3	6.2	13.3	14.8	23.3	4.8
	2	56.6	7.1	8.6	5.7	6.4	9.3	6.3
	3	78.7	5.2	5.9	2.2	2.2	0.0	5.8
TS1	0	41.2	4.0	10.6	14.4	17.0	11.0	1.7
	1	64.4	5.6	5.9	5.8	5.7	10.8	1.9
	2	81.0	5.4	3.7	2.3	1.9	3.5	2.1
	3	90.4	3.9	1.7	1.0	0.8	0.0	2.1
TS2	0	9.2	3.3	11.0	33.2	27.1	13.5	2.7
	1	36.6	6.9	9.6	14.0	9.5	20.0	3.5
	2	62.9	8.8	7.2	5.5	3.5	7.8	4.4
	3	80.7	6.4	3.7	2.9	1.4	0.0	4.8
TS3	0	29.7	16.4	9.1	18.4	15.4	8.7	2.3
	1	48.0	16.9	5.7	8.0	6.4	11.3	3.7
	2	70.4	10.5	4.8	2.8	2.2	3.6	5.8
	3	85.8	5.4	2.4	1.0	0.6	0.0	4.7
TS4	0	77.8	3.5	2.6	6.2	5.4	3.5	1.0
	1	88.8	2.4	1.8	2.0	1.4	2.6	1.1
	2	95.1	1.1	1.1	0.6	0.3	0.6	1.1
	3	97.9	0.4	0.4	0.2	0.1	0.0	1.1
平均值		62.0	6.2	5.5	8.4	7.5	7.1	3.3

3.1 CTU 深度范围预测算法

为了分析 CTU 深度信息的时空相关性，本节先对不同时空位置 CTU 进行标记。在图 2 中，Cur-CTU 表示当前 CTU，L-CTU、LU-CTU、U-CTU 和 RU-CTU 分别表示 Cur-CTU 空间相邻的左方、左上方、上方和右上方的 CTU，Col-CTU 表示时间相邻的参考帧中与 Cur-CTU 相同位置的 CTU。

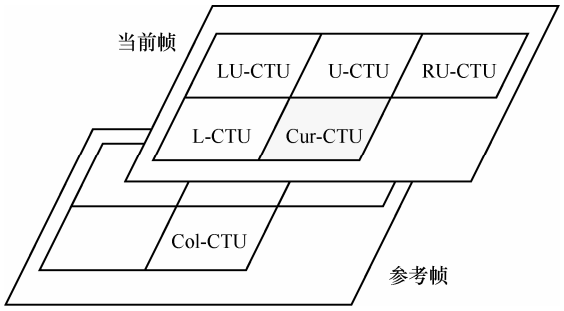


图 2 相邻 CTU 示意

表 3 给出了 Cur-CTU 分布与每个相邻 CTU 的最大深度值 *Depth* 的差值统计，测试条件同第 2 节。如果差值越小，则表明 Cur-CTU 与对应的相邻 CTU 的 *Depth* 相关性越强，反之则表明相关性越弱。从表中可以看出，Cur-CTU 与 5 个相邻 CTU 在不同测试序列下的平均差值都小于 1，这说明当前 CTU 与相邻 CTU 的都具有一定的时空相关性。从表中还可以看出，Cur-CTU 与 Col-CTU、L-CTU 和 U-CTU 的平均差值都比与其他 2 个相邻 CTU 的平均差值小 0.1 以上，并且比较接近，其中，Cur-CTU 与 Col-CTU 的平均差值最小。因此，在 5 个时空相邻 CTU 中，Cur-CTU 与 Col-CTU、L-CTU 和 U-CTU 的深度信息相关性较强，而且 Cur-CTU 与 Col-CTU 的深度信息相关性最强，可以利用 Col-CTU、L-CTU 和 U-CTU 的深度信息预测 Cur-CTU 的深度范围。

表 3 Cur-CTU 与相邻 CTU 的 *Depth* 差值

序列	Cur-CTU 与相邻 CTU 的 *Depth* 差值				
	L-CTU	LU-CTU	U-CTU	RU-CTU	Col-CTU
TS0	0.56	0.72	0.61	0.67	0.38
TS1	0.73	0.96	0.79	0.99	0.87
TS2	0.73	0.77	0.72	0.88	0.80
TS3	0.93	1.19	0.80	1.06	0.56
TS4	0.32	0.48	0.41	0.51	0.32
平均值	0.65	0.82	0.67	0.82	0.59

根据以上分析，本文提出的 CDRP 算法在 Col-CTU 存在时，且 L-CTU 和 U-CTU 至少 1 个存在的情况下，按式(1)预测 Cur-CTU 的深度范围（记为 R）。式中 D_{MIN} 和 D_{MAX} 分别表示存在的相邻 CTU 中的最小深度值和最大深度值，当 D_{MIN} 与 D_{MAX} 不相等，则直接将$[D_{\mathrm{MIN}}, D_{\mathrm{MAX}}]$作为 R；当 D_{MIN}、D_{MAX} 都为 0 时，则将[0, 1]作为 R；当 D_{MIN}、D_{MAX} 都为 3 时，则将[2, 3]作为 R；其他情况下将$[D_{\mathrm{MIN}}-1, D_{\mathrm{MAX}}+1]$作为 R。为了验证 CDRP 算法的有效性，表 4 给出了不同 *QP* 下的准确率统计。当 Cur-CTU 中的最终划分深度（HM 中用多个 4×4 块深度来表示）在预测范围内，则认为 CDRP 算法命中。从表中可以看出，不同 *QP* 下 5 个序列的准确率为 94.5%~99.4%，且平均值都达到 95%以上。其中，运动量小且纹理简单的 TS4 序列获得的平均准确率高达 98.3%。这些数据表明本节提出的 CDRP 算法很好地预测了 CTU 的深度范围。

$$R=\begin{cases}[D_{\mathrm{MIN}}, D_{\mathrm{MAX}}], & D_{\mathrm{MIN}} \neq D_{\mathrm{MAX}} \\ [0, 1], & D_{\mathrm{MIN}} = D_{\mathrm{MAX}} = 0 \\ [2, 3], & D_{\mathrm{MIN}} = D_{\mathrm{MAX}} = 3 \\ [D_{\mathrm{MIN}}-1, D_{\mathrm{MAX}}+1], & \text{其他}\end{cases} \tag{1}$$

表 4 不同 *QP* 下 CDRP 算法的准确率

QP	CDRP 算法的准确率/%				
	TS0	TS1	TS2	TS3	TS4
22	97.7	96.4	98.5	94.5	96.9
27	97.2	94.9	97.2	95.8	97.9
32	97.2	95.3	94.7	96.4	98.9
37	96.7	96.7	95.1	96.8	99.4
平均值	97.2	95.8	96.4	95.9	98.3

3.2 帧间 PU 模式选择算法

由于父 CU 与当前 CU 之间存在包含和被包含的空间关系，它们之间的 PU 模式选择也应存在一定的空间相关性，因此本节先利用父 CU 的最佳 PU 模式对当前 CU 的最佳 PU 模式进行分类统计。表 5 给出了 TS0 和 TS1 序列在父 CU 最佳 PU 模式分别为 Skip、Merge、Inter 2N×2N、Inter N×2N 和 Inter 2N×N 的情况下，当前 CU 的最佳 PU 模式比例，测试条件同第 2 节。对于 TS0 和 TS1 序列，当父 CU 的最佳 PU 模式为 Skip 时，当前 CU 将 Skip 作为最佳 PU 模式的比例分别达到 98.8%和 98.9%，而选择 Inter N×2N、Inter 2N×N 或 Inter AMP 作为最佳

PU 模式的比例分别只占 0.4%和 0.5%。对于 TS0 序列，当父 CU 的最佳 PU 模式为 Merge 和 Inter 2N×2N 时，Inter AMP 的比例分别为只有 1.8%和 2.0%。以上数据是因为当父 CU 最佳 PU 模式为 Skip，那么当前 CU 所覆盖的区域大都运动量小且纹理简单，它选择 Inter N×2N、Inter 2N×N 或 Inter AMP 的可能性小。同理，当父 CU 最佳 PU 模式为 Merge 或 Inter 2N×2N，那么当前 CU 所覆盖的区域运动量较小且纹理较简单，它选择 Inter AMP 的可能性也小。这些数据及分析说明当前 CU 与其父 CU 的最佳 PU 模式具有较强的空间相关性，尤其是当父 CU 的最佳 PU 模式为 Skip、Merge 或 Inter 2N×2N 时，当前 CU 可以跳过一些成为最佳 PU 模式可能性较小的帧间 PU 模式。

在 HM 参考算法中，当前 CU 的所有 PU 模式依次进行估计，并将率失真代价最小的 PU 模式被选为最佳 PU 模式。表 6 给出了 TS2 和 TS3 序列在 PU 模式选择过程中不同阶段下，当前 CU 的最佳 PU 模式分布，测试条件同第 2 节。在 Inter 2N×2N 模式估计之后，如果 Skip 或者 Merge 模式的率失真代价最小，统计所有模式估计后当前 CU 最佳 PU 模式的分布；在 Inter N×2N 和 Inter 2N×N 模式估计之后，如果 Skip、Merge 或 Inter 2N×2N 模式的率失真代价最小，统计所有模式估计后当前 CU 最佳 PU 模式的分布，如表 6 所示。从表中可以看出，在上述第一种情况中，对于 TS3 序列，当前 CU 选择 Inter N×2N、Inter 2N×N 或 Inter AMP 作为最佳 PU 模式的比例为 2.5%；在第二种情况中，对于 TS2 和 TS3 序列，当前 CU 选择 Inter AMP 的作为最佳 PU 模式的比例分别为 1.2%和 0.7%。因此，在当前 CU 依次进行 PU 模式估计的过程中，如果已估计模式中 Skip、Merge 或 Inter 2N×2N 的率失真代价，那么当前 CU 可以跳过那些后续选为最佳 PU 模式可能性较小的帧间 PU 模式。

根据以上分析，本文提出的 IPMS 算法利用当前 CU 与其父 CU 的最佳 PU 模式的空间相关性和当前 CU 已估计 PU 模式的率失真代价，分 5 种情况跳过当前 CU 的部分候选帧间 PU 模式，如表 7 所示。1) 当父 CU 的最佳 PU 模式为 Skip，则当前 CU 将跳过对 Inter N×2N、Inter 2N×N 和 Inter AMP 模式的估计。2) 当父 CU 最佳模式为 Merge 或 Inter 2N×2N，则当前 CU 将跳过对 Inter AMP 模式的估计。3) 在 Inter 2N×2N 模式估计之后，如果当前 CU 符合 Skip 模式条件且 RD_{Skip} 小于 $RD_{\text{Inter2N×2N}}$（即

表 5　　当前 CU 与其父 CU 的最佳 PU 模式关系

序列	父 CU 最佳 PU 模式	当前 CU 的最佳 PU 模式比例/%						
		Skip	Merge	Inter 2N×2N	Inter N×2N	Inter 2N×N	Inter AMP	其他
TS0	Skip	98.8	0.4	0.4	0.2	0.2	0.0	0.1
	Merge	53.2	21.9	5.7	3.9	3.7	1.8	9.7
	Inter 2N×2N	60.0	9.1	16.1	3.5	3.6	2.0	5.7
	Inter N×2N	45.8	8.4	15.1	7.7	8.2	8.8	6.1
	Inter 2N×N	45.3	9.0	12.2	9.5	10.1	7.4	6.6
TS1	Skip	98.9	0.4	0.2	0.3	0.1	0.1	0.0
	Merge	52.4	29.1	5.0	3.7	3.1	1.9	4.8
	Inter 2N×2N	62.2	13.1	12.6	3.5	3.0	2.8	2.9
	Inter N×2N	57.9	12.1	10.9	5.4	4.7	6.3	2.7
	Inter 2N×N	52.6	13.2	12.0	6.1	6.1	7.7	2.4

表 6　　不同估计阶段下最佳 PU 模式分布

序列	不同估计阶段	率失真代价最小的 PU 模式	当前 CU 最佳 PU 模式比例/%			
			Inter N×2N	Inter 2N×N	AMP	其他
TS2	Inter 2N×2N 模式估计后	Skip 或 Merge	2.8	1.5	1.2	94.5
	Inter N×2N 和 Inter 2N×N 模式估计后	Skip、Merge 或 Inter 2N×2N	0	0	1.2	98.8
TS3	Inter 2N×2N 模式估计后	Skip 或 Merge	1.2	0.7	0.6	97.5
	Inter N×2N 和 Inter 2N×N 模式估计后	Skip、Merge 或 Inter 2N×2N	0	0	0.7	99.3

*Flag*1 等于 1），那么当前 CU 跳过 Inter N×2N、Inter 2N×N 和 Inter AMP 模式的估计。其中，*Flag*1 是候选帧间 PU 模式跳过判别标识 1，按式(2)计算，式中 $RD_{\text{Inter2N×2N}}$ 表示 Inter 2N×2N 模式的率失真代价，RD_{Skip} 表示 Skip 模式的率失真代价。4) 在估计 Inter 2N×2N 模式之后，如果当前 CU 不符合 Skip 模式条件且 RD_{Merge} 小于 $RD_{\text{Inter2N×2N}}$ 以及 RD_{Parent} 的 $\frac{1}{4}$（即 *Flag*2 等于 1），那么当前 CU 跳过 Inter N×2N、Inter 2N×N 和 Inter AMP 模式的估计。其中 *Flag*2 是候选帧间 PU 模式跳过判别标识 2，按式(3)计算得到。由于当前 CU 与父 CU 具有空间相关性，同时也为了提高跳过准确率，式(3)中加入了父 CU 最佳模式的率失真代价 RD_{Parent}。5) 在估计 Inter N×2N 和 Inter 2N×N 模式之后，如果 RD_{Square} 小于 RD_{Symmetry}（即 *Flag*3 等于 1），则跳过 Inter AMP 模式的估计。其中，*Flag*3 是候选帧间 PU 模式跳过判别标识 3，按式(4)计算得到，式中 RD_{Square} 表示 RD_{Skip}、RD_{Merge} 和 $RD_{\text{Inter2N×2N}}$ 的较小值，RD_{Symmetry} 表示 Inter N×2N 和 Inter 2N×N 模式的率失真代价较小值。

表 7 当前 CU 可跳过的帧间 PU 模式

判别条件	当前 CU 跳过的帧间 PU 模式
父 CU 最佳 PU 为 Skip	Inter N×2N、Inter 2N×N 和 Inter AMP
父 CU 最佳 PU 为 Merge 或 Inter 2N×2N	Inter AMP
Inter 2N×2N 估计后当前 CU 符合 Skip 条件且 *Flag*1 等于 1	Inter N×2N、Inter 2N×N 和 Inter AMP
Inter 2N×2N 估计后当前 CU 不符合 Skip 条件且 *Flag*2 等于 1	Inter N×2N、Inter 2N×N 和 Inter AMP
Inter N×2N 和 Inter 2N×N 估计后 *Flag*3 等于 1	Inter AMP

$$Flag1=\begin{cases}1, & RD_{\text{Skip}}<RD_{\text{Inter 2 N×2 N}}\\ 0, & \text{其他}\end{cases} \tag{2}$$

$$Flag2=\begin{cases}1, & RD_{\text{Merge}}<\min\left(RD_{\text{Inter2N×2N}},\ \frac{1}{4}RD_{\text{Parent}}\right)\\ 0, & \text{其他}\end{cases} \tag{3}$$

$$Flag3=\begin{cases}1, & RD_{\text{Square}}<RD_{\text{Symmetry}}\\ 0, & \text{其他}\end{cases} \tag{4}$$

为了验证 IPMS 算法的有效性，表 8 给出了不同 *QP* 下的准确率统计。当 CU 的最佳 PU 模式不是表 7 中跳过的帧间 PU 模式，则认为 IPMS 算法命中。5 个序列在不同 *QP* 下对应的平均准确率分别为 97.5%、98.1%、96.0%、98.7%和 99.4%，其中，当 *QP* 为 37 时，TS4 序列的准确率高达 99.9%。这些统计数据说明本节提出的 IPMS 算法具有很好的跳过准确率。

表 8 不同 *QP* 下 IPMS 算法的准确率

QP	IPMS 算法的准确率/%				
	TS0	TS1	TS2	TS3	TS4
22	95.2	95.6	91.3	97.4	98.6
27	97.5	98.0	95.7	98.6	99.5
32	98.4	99.1	98.0	99.2	99.8
37	98.8	99.6	99.1	99.5	99.9
平均值	97.5	98.1	96.0	98.7	99.4

3.3 总体算法

本文提出的总体算法包括 CTU 深度范围预测算法 CDRP 和帧间 PU 模式选择算法 IPMS，总体算法流程如图 3 所示。

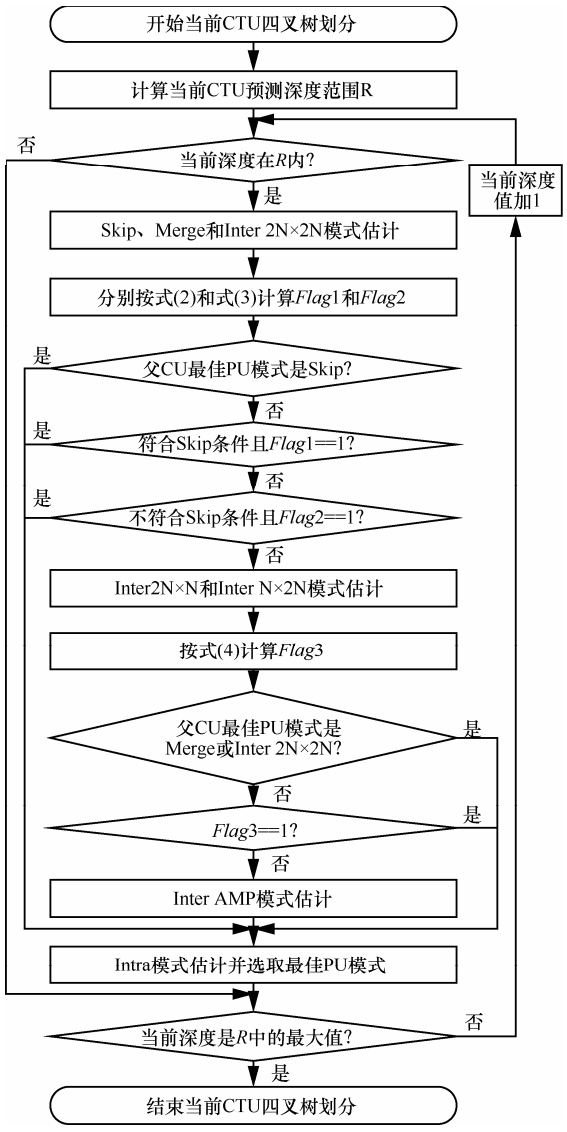


图 3 总体算法流程

4　实验结果及分析

本文算法的软件实现平台为 HM-8.0，其基本参数包括：CTU 大小为 64×64，最大深度为 3，编码配置为 LDP(low delay P)和 RA(random access)，*QP* 为 22、27、32 和 37，选取的 18 个典型测试序列包括 A、B、C、D 和 E 5 种类别，每个测试序列的像素深度为 8 bit，其他配置和参数见文献[10]。硬件实现平台的处理器为 Intel Xeon E5-2440，主频为 2.40 GHz，内存为 48.0 GB。为了统计 4 个 *QP* 下的编码率失真性能变化，实验计算了每个序列的 BDPSNR(bjontegaard delta peak signal to noise ratio)和 BDBR(bjontegaard delta bit rate)[11]。为了同时考虑亮度和色度的编码率失真性能，在计算 BDPSNR 和 BDBR 时所使用的峰值信噪比 PSNR 按式(5)[12]计算得到。

$$PSNR=\frac{(PSNR_{\mathrm{Y}}\times 6+PSNR_{\mathrm{U}}+PSNR_{\mathrm{V}})}{8} \tag{5}$$

其中，$PSNR_{\mathrm{Y}}$ 是亮度峰值信噪比，$PSNR_{\mathrm{U}}$ 和 $PSNR_{\mathrm{V}}$ 是色度峰值信噪比。为了统计编码时间变化，实验利用 HM 的原始编码时间 T_{org} 和算法优化后的编码时间 T_{fast} 计算编码时间变化率 ΔT，如式(6)所示。

$$\Delta T=\frac{(T_{\mathrm{org}}-T_{\mathrm{fast}})}{T_{\mathrm{org}}}\times 100\% \tag{6}$$

表 9 给出了在关闭 HM-8.0 已有快速算法 CFM(coding_block_flag fast mode)[13]、ECU(early coding unit)[14]和 ESD(early skip detection)[15]的情况下，本文 CDRP 算法和 IPMS 算法的性能。从表中可以看出，与 HM 参考算法相比，CDRP 算法在 LDP 和 RA 配置下分别平均减少了 24.8%和 22.6%的编码时间，IPMS 算法平均分别减少 40.5%和 41.1%的编码时间，同时它们的 BDPSNR 和 BDBR 变化非常小，基本不影响视频质量。对于 LDP 配置下的 FourPeople、Johnny 和 KristenAndSara 序列，CDRP 算法和 IPMS 算法编码效果好于其他序列，这是因为在运动量小且纹理简单的场景中 CTU 时空相关性和 PU 空间相关性都较强，本文算法获得了更好的计算性能。

为了评估算法的整体性能，表 10 给出了本文总体算法与文献[7]提出算法的性能。从表中可以看出，在 LDP 和 RA 配置下，文献[7]算法分别平均降低 37.6%和 38.8%的编码时间。在 LDP 配置下，对于运动量大且纹理复杂的测试序列，如 RaceHorses、BlowingBubbles 和 BasketballPass 序列，文献[7]算法只降低了 23.0%到 28.2%的编码时间，

表 9　本文 CDRP 和 IPMS 算法的性能

序列	LDP 配置		RA 配置	
	本文 CDRP 算法 BDBR/BDPSNR/ΔT (%/dB/%)	本文 IPMS 算法 BDBR/BDPSNR/ΔT (%/dB/%)	本文 CDRP 算法 BDBR/BDPSNR/ΔT (%/dB/%)	本文 IPMS 算法 BDBR/BDPSNR/ΔT (%/dB/%)
Traffic	—	—	0.32/−0.01/26.6	0.75/−0.02/49.4
PeopleOnStreet	—	—	0.21/−0.01/21.2	1.01/−0.04/32.2
Kimono	0.37/−0.01/29.7	0.82/−0.02/40.9	0.56/−0.01/30.8	0.72/−0.02/44.9
ParkScene	0.29/−0.01/23.3	1.46/−0.04/43.7	0.29/−0.01/24.1	0.84/−0.02/47.6
Cactus	0.32/−0.01/26.8	1.39/−0.03/40.7	0.25/0.00/26.2	0.75/−0.01/44.9
BQTerrace	0.40/−0.01/28.1	1.75/−0.02/44.6	0.25/0.00/27.6	0.78/−0.01/49.1
BasketballDrive	0.47/−0.01/26.7	0.94/−0.02/40.3	0.44/−0.01/26.7	0.83/−0.02/43.7
RaceHorsesC	0.30/−0.01/18.0	1.28/−0.04/31.7	0.46/−0.02/19.3	1.27/−0.04/33.4
BQMall	0.46/−0.02/20.8	1.63/−0.06/39.0	0.21/−0.01/20.9	1.10/−0.04/42.8
PartyScence	0.11/0.00/19.8	1.41/−0.05/34.5	0.18/−0.01/20.3	0.78/−0.03/38.6
BasketballDrill	0.43/−0.02/19.9	1.06/−0.04/36.2	0.32/−0.01/19.9	0.73/−0.03/39.1
RaceHorses	0.11/0.00/16.9	1.70/−0.07/29.4	0.25/−0.01/18.9	1.35/−0.06/30.2
BQSquare	0.11/0.00/18.8	1.95/−0.06/38.0	0.63/−0.02/20.8	0.68/−0.02/45.3
BlowingBubbles	0.18/−0.01/17.4	2.01/−0.07/36.6	0.10/0.00/19.5	0.87/−0.03/40.2
BasketballPass	0.11/−0.01/14.9	1.21/−0.05/32.3	0.25/−0.01/16.4	1.08/−0.05/34.6
FourPeople	0.85/−0.03/35.1	1.43/−0.04/51.1	—	—
Johnny	1.02/−0.02/42.2	2.23/−0.05/55.6	—	—
KristenAndSara	0.94/−0.03/38.5	1.58/−0.04/53.1	—	—
平均值	0.40/−0.01/24.8	1.49/−0.04/40.5	0.31/−0.01/22.6	0.90/−0.03/41.1

表 10　　本文总体算法与文献[7]算法的性能

序列	LDP 配置		RA 配置	
	文献[7]算法	本文总体算法	文献[7]算法	本文总体算法
	BDBR/BDPSNR/Δ*T* (%/dB/%)	BDBR/BDPSNR/Δ*T* (%/dB/%)	BDBR/BDPSNR/Δ*T* (%/dB/%)	BDBR/BDPSNR/Δ*T* (%/dB/%)
Traffic	—	—	0.65/−0.02/49.1	1.12/−0.03/60.1
PeopleOnStreet	—	—	0.89/−0.03/31.4	1.25/−0.05/46.9
Kimono	0.36/−0.01/37.3	1.28/−0.04/55.2	0.56/−0.02/42.6	1.26/−0.03/59.0
ParkScene	1.05/−0.03/39.8	1.79/−0.05/53.8	0.71/−0.02/46.3	1.19/−0.03/58.3
Cactus	0.97/−0.02/40.5	1.69/−0.03/53.2	0.66/−0.01/44.8	0.96/−0.02/56.8
BQTerrace	1.73/−0.02/43.7	2.02/−0.03/55.8	1.13/−0.02/48.9	1.04/−0.01/60.2
BasketballDrive	0.69/−0.02/39.4	1.39/−0.03/52.8	0.75/−0.02/43.1	1.30/−0.03/55.9
RaceHorsesC	0.87/−0.03/29.2	1.51/−0.05/43.7	0.84/−0.03/30.0	1.88/−0.06/46.7
BQMall	1.14/−0.04/34.7	2.00/−0.07/49.1	0.77/−0.03/39.9	1.88/−0.05/53.7
PartyScence	0.96/−0.03/28.5	1.59/−0.06/46.0	0.77/−0.03/35.7	0.91/−0.03/51.0
BasketballDrill	0.89/−0.03/34.9	1.32/−0.05/46.6	0.67/−0.03/37.5	1.03/−0.04/49.6
RaceHorses	0.81/−0.03/23.0	1.81/−0.07/41.6	0.79/−0.03/24.4	1.70/−0.07/44.7
BQSquare	1.61/−0.04/28.9	2.05/−0.06/47.6	0.69/−0.02/40.8	0.80/−0.03/55.7
BlowingBubbles	1.15/−0.04/26.3	1.99/−0.07/45.3	0.78/−0.03/35.4	1.01/−0.04/51.1
BasketballPass	0.75/−0.03/28.2	1.49/−0.07/42.7	0.74/−0.03/31.2	1.31/−0.06/45.1
FourPeople	1.92/−0.06/54.7	2.50/−0.07/63.4	—	—
Johnny	1.42/−0.03/58.3	2.84/−0.06/68.7	—	—
KristenAndSara	1.02/−0.03/54.3	2.59/−0.07/65.8	—	—
平均值	1.08/−0.03/37.6	1.87/−0.05/52.0	0.76/−0.02/38.8	1.21/−0.04/53.0

这表明对运动量大且纹理复杂的序列其计算性能还有进一步提高的空间。从表中可以看出，在 LDP 和 RA 配置下，本文总体算法分别平均降低了 52.0%和 53.0%的编码时间，而 BDBR 和 BDPSNR 损失较小，与文献[7]算法相近。对于在 LDP 配置下的 Johnny 序列，本文总体算法减少的编码时间高达 68.7%，是所有序列中最多的，这与表 9 中表现的 CDRP 算法和 IPMS 算法的计算性能一致，说明总体算法对运动量小且纹理简单的序列具有更好的效果。对于在 RA 配置下的 RaceHorses 序列，本文总体算法降低的编码时间比文献[7]算法降低的编码时间多了 20.3%。对于在 LDP 配置下 RaceHorses 序时编码率失真性能保持较好。从表中还可以看出，高分辨率的 A 类、B 类和 E 类序列的编码时间减少程度普遍高于低分辨率的 C 类和 D 类序列，这说明对于高分辨率序列，CTU 空间相关性更强，本文总体算法的计算性能更好。对于 RaceHorses 序列，本文总体算法减少了 41.6%的编码时间，是所有序列中最少的，这是因为该序列运动剧烈，CTU 时空相关性和 PU 空间相关性较弱，虽然对于该序列，本文总体算法降低的编码时间比其他序列少，但是依然比文献[7]算法多降低了 18.6%的编码时间。

为了进一步评估算法性能，表 11 给出了打开

表 11　打开 CFM、ECU 和 ESD 的本文总体算法性能

序列	LDP 配置	RA 配置
	BDBR/BDPSNR/Δ*T* (%/dB/%)	BDBR/BDPSNR/Δ*T* (%/dB/%)
Traffic	—	1.72/−0.05/29.5
PeopleOnStreet	—	1.80/−0.07/35.7
Kimono	1.53/−0.04/37.5	1.58/−0.04/36.1
ParkScene	1.99/−0.05/29.4	0.71/−0.02/30.1
Cactus	2.29/−0.05/32.6	2.01/−0.04/31.5
BQTerrace	2.01/−0.03/28.2	1.17/−0.02/27.4
BasketballDrive	1.53/−0.03/33.4	1.65/−0.03/32.2
RaceHorsesC	1.64/−0.05/33.3	2.90/−0.10/34.2
BQMall	2.00/−0.07/29.4	1.89/−0.07/29.6
PartyScence	1.61/−0.06/30.4	1.27/−0.05/32.2
BasketballDrill	1.64/−0.06/30.8	1.44/−0.05/30.2
RaceHorses	1.99/−0.08/31.7	3.14/−0.13/34.0
BQSquare	2.03/−0.06/26.3	1.00/−0.03/27.4
BlowingBubbles	1.83/−0.06/28.8	1.50/−0.06/30.2
BasketballPass	1.60/−0.07/29.1	1.91/−0.09/29.3
FourPeople	2.65/−0.08/25.4	—
Johnny	3.21/−0.06/23.8	—
KristenAndSara	3.15/−0.08/27.8	—
平均值	2.04/−0.06/29.9	1.71/−0.06/31.3

HM-8.0 中的 CFM、ECU 和 ESD 快速算法选项下的总体算法性能。本文总体算法在 LDP 和 RA 配置下分别平均减少 29.9%和 31.3%的编码时间，而 BDBR 分别平均增加 2.04%和 1.71%，BDPSNR 都降低了 0.06 dB，编码率失真性能损失较小。对于 LDP 配置下的 FourPeople 和 Johnny 序列，降低的编码时间只有 23.8%和 25.4%。这是因为对于运动量小且纹理简单的测试序列， CFM、ECU 和 ESD 优化效果较好，本文总体算法进一步提高计算性能的余地相对较少。从表中还可以看出，对于运动量较大且纹理较复杂的 Kimono 序列，在 LDP 和 RA 配置下分别减少 37.5%和 36.1%的编码时间，这说明在打开 CFM、ECU 和 ESD 的情况下，本文总体算法在这种场景下优化效果显著。

5　结束语

为了降低 HEVC 编码模式决策的计算复杂度，本文提出了一种基于时空相关性的帧间模式决策快速算法，包括利用 3 个相邻 CTU 的深度信息预测当前 CTU 的深度范围，以及利用父 CU 的最佳 PU 模式和当前 CU 已估计 PU 模式的率失真代价来减少候选帧间 PU 模式个数。实验结果表明，本文算法平均降低了 52%以上的编码时间，而 BDBR 和 BDPSNR 变化较小。与文献[7]提出的快速算法相比，本文算法进一步降低了约 14%的编码时间，同时编码率失真性能与之相当。此外，在文献[13~15]的模式决策快速算法的基础上，本文算法进一步降低了约 30%的编码时间，同时编码率失真性能损失较小。

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 and Systems for Video Technology, 2012, 22(12): 1649-1668.

[2] KIM I K, MIN J, LEE T, et al. Block partition structure in the HEVC standard[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2012, 22(12): 1697-1706.

[3] 周承涛, 田翔, 陈耀武. HEVC 编码单元尺寸快速选择算法[J]. 浙江大学学报（工学版）, 2014, 48(8): 1451-1460.
ZHOU C T, TIAN X, CHEN Y W. Fast coding unit size decision for HEVC[J]. Journal of Zhejiang University (Engineering Science), 2014, 48(8): 1451-1460.

[4] GARCIA R, KALVA H. HEVC inter-frame skip enhancement at low bit rates[C]//2014 IEEE International Conference on Consumer Electronics (ICCE). Las Vegas, c2014: 59-60.

[5] 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 and Systems for Video Technology, 2015, 25(3):422-435.

[6] 蒋刚毅, 杨小祥, 彭宗举, 等. 高效率视频编码的快速编码单元深度遍历选择和早期编码单元裁剪[J]. 光学　精密工程, 2014, 22(5): 1322-1330.
JIANG G Y, YANG X X, PENG Z J, et al. Fast CU depth rang selection and early CU pruning for HEVC[J]. Optics and Precision Engineering, 2014, 22(5): 1322-1330.

[7] SHEN L Q, LIU Z, ZHANG X P, et al. An effective CU size decision method for HEVC encoders[J]. IEEE Transactions on Multimedia, 2013, 15(2): 465-470.

[8] 齐美彬, 陈秀丽, 杨艳芳, 等. 高效率视频编码帧内预测编码单元划分快速算法[J]. 电子与信息学报, 2014, 36(7): 1699-1705.
QI M B, CHEN X L, YANG Y F, et al. Fast coding unit splitting algorithm for high efficiency video coding intra prediction[J]. Journal of Electronics & Information Technology, 2014, 36(7): 1699-1705.

[9] SHEN L Q, ZHANG Z Y, LIU Z. Effective CU size decision for HEVC intracoding[J]. IEEE Transactions on Image Processing, 2014, 23(10): 4232-4241.

[10] BOSSEN F. Common test conditions and software reference configurations[C]//Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC, JCTVC-J1100. Stockholm, c2012: 1-3.

[11] BJONTEGAARD G. Calculation of average PSNR differences between RD curves[C]//ITU-T SG16/Q6, VCEG-M33. Austin, c2001: 1-4.

[12] OHM J R, SULLIVAN G J, SCHWARZ H, et al. Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC)[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2012, 22(12): 1669-1684.

[13] GWEON R H, LEE Y L, LIM J. Early termination of CU encoding to reduce HEVC complexity[C]//Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC, JCTVC-F045. Torino, c2011: 1-4.

[14] CHOI K, PARK S H, JANG E S. Coding tree pruning based CU early termination[C]//Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC, JCTVC-F092. Torino, c2011: 1-4.

[15] YANG J, KIM J, WON K, et al. Early SKIP detection for HEVC[C]//Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC, JCTVC-G543. Geneva, c2011: 1-6.

作者简介：

朱威（1982-），男，浙江杭州人，浙江工业大学讲师，主要研究方向为视频编解码和机器视觉。

张训华（1989-），男，河南商丘人，浙江工业大学硕士生，主要研究方向为 HEVC 编码。

王财盛（1991-），男，浙江宁波人，浙江工业大学硕士生，主要研究方向为图像处理。

张桦（1980-），女，浙江杭州人，杭州电子科技大学讲师，主要研究方向为视频编解码和图像处理。