Content-Adaptive Feature-Based CU Size Prediction for Fast Low-Delay Video Encoding in HEVC

Thanuja Mallikarachchi, Student Member, IEEE, Dumidu S. Talagala, Member, IEEE, Hemantha Kodikara Arachchi, Member, IEEE, and Anil Fernando, Senior Member, IEEE

Abstract—Determining the best partitioning structure of a Coding Tree Unit (CTU) is one of the most time consuming operations in HEVC encoding. Specifically, it is the evaluation of the quadtree hierarchy using the Rate-Distortion (RD) optimization that has the most significant impact on the encoding time, especially in the cases of High Definition (HD) and Ultra High Definition (UHD) videos. In order to expedite the encoding for low delay applications, this paper proposes a Coding Unit (CU) size selection and encoding algorithm for inter-prediction in the HEVC. To this end, it describes (i) two CU classification models based on Inter $N \times N$ mode motion features and RD cost thresholds to predict the CU split decision, (ii) an online training scheme for dynamic content adaptation, (iii) a motion vector reuse mechanism to expedite the motion estimation process, and finally introduces (iv) a computational complexity to coding efficiency mechanism to expedite the motion estimation process, and finally introduces (iv) a computational complexity to coding efficiency trade-off process to enable flexible control of the algorithm. The experimental results reveal that the proposed algorithm achieves a consistent average encoding time performance ranging from 55% - 58% and 57%-61% with average Bjøntegaard Delta Bit Rate (BDBR) increases of 1.93% – 2.26% and 2.14% – 2.33% compared to the HEVC 16.0 reference software for the low delay $P$ and low delay $B$ configurations, respectively, across a wide range of content types and bit rates.

Index Terms—HEVC, CU size, low-delay, motion classification, RD optimization, video coding

I. INTRODUCTION

T TO recent developments in the Consumer Electronic (CE) technologies, and the content capturing capabilities of these devices have made multimedia the most frequently exchanged type of content over the modern communication networks. Moreover, the rapidly increasing mobile video data traffic, including the High Definition (HD) and Ultra High Definition (UHD) content captured by CE devices, (forecast to reach three-fourth of the mobile data traffic in 2019 [1]), present challenges such as the need for greater compression efficiency, energy efficiency and speed. In this context, the High Efficiency Video Coding (HEVC) [2] addresses the first of these requirements through the vastly superior performance exhibited over its predecessor H.264/AVC. However, the increased complexity of the features in the HEVC architecture significantly increase the demand for computational time and energy [5]; a non-trivial bottleneck for resource-constrained CE devices such as smart phones and camcorders. Efficient encoder designs that expedite the encoding process are therefore crucially important for the realization of high frame rate and real-time video communication applications in CE devices.

Although HEVC is essentially based on a hybrid coding architecture similar to that of H.264/AVC, it is accompanied by an assortment of novel coding features such as efficient prediction modes, filtering modes, parallelization tools, and flexible coding structures (e.g., Coding Units (CU), Prediction Units (PU), Transform Units (TU), etc.) [2]. The wide range of block sizes and combinations (i.e., $8 \times 8$ to $64 \times 64$) that this entails is one of the most important contributors towards the encoder’s improved efficiency, yet at the same time, is also a major source of the complexity within the HEVC architecture [4], [5]. This is mainly due to the brute force Rate-Distortion (RD) optimization required for the combinations of coding modes that determines the best coding configuration. For example, an average encoding time increase of 43% is reported in [4], due to a simple increase of the maximum CU size from $16 \times 16$ to $64 \times 64$. Therefore, the recent literature has predominantly proposed mechanisms to reduce the complexity of the RD optimization that selects the best coding structure. In this context, the state-of-the-art fast encoding solutions generally utilize the depth correlation of spatial and temporal blocks, RD cost statistics of the CUs and the Inter $2N \times 2N$ prediction mode, feature-based offline and online training approaches, etc. [8]- [23], to determine the optimum CU size. Hence, the selection of the CU size now becomes a prediction, whose effectiveness will determine the output quality and the bit rate of the encoded content. However, the vast differences in video characteristics, and the availability of additional information in the encoding chain itself, have not been fully investigated nor have they been exploited in these prediction approaches in order to realize a consistent encoding time saving across a wide range of content types and quality settings. Thus, the potential exists to develop implementation-friendly encoding algorithms that can effectively trade-off the coding efficiency in order to gain a reduction of the computational complexity.

To this end, this paper proposes a CU size prediction mechanism for fast low-delay HEVC video encoding based on the following contributions. First, (i) two independent content adaptive decision making models are proposed to enable the prediction of the optimal CU size. These are, a dynamic motion feature-based CU classification

1From the implementation perspective of the encoder, the CU size prediction boils down to a decision of whether a particular CU should be split into smaller CUs or not, i.e., the CU split decision referred to in the literature.
model created by evaluating the Inter $N \times N$ mode for a CU classification using both motion and RD cost characteristics, and a heuristic RD cost threshold-based model of the CU split decision. Thereafter, (ii) a window-based feature selection process is introduced to each model for independent, content specific updating, which ensures the framework remains robust and dynamically adapts to varying conditions such as scene changes. Then, (iii) Inter $N \times N$ mode motion vector reuse for further encoding complexity reduction, and (iv) the inclusion of a complexity control parameter to enable the flexibility of trading-off the proposed algorithm’s complexity reduction for coding efficiency, are investigated. The simulation results reveal that the proposed algorithm achieves a significant and a consistent reduction of the encoding time compared to the HM 16.0 [26] implementation and state-of-the-art algorithms with minimal impact on the RD performance for a variety of content types, and quality levels.

The remainder of the paper is organized as follows. Section II provides an overview of the prior work in the literature. The motion feature-based CU classification model and the RD cost threshold-based CU classification model are described in Section III. The resulting CU size prediction algorithm and the encoding framework are presented in Sections IV and V, respectively, and are followed by the results and discussion in Section VI. Finally, Section VII concludes and discusses the potential for future improvements.

II. BACKGROUND AND RELATED WORK

A. Background

HEVC utilizes a block-based encoding architecture similar to H.264/AVC, albeit with a wider range of block sizes. In the main profile of HEVC, for example, a Coding Tree Unit (CTU) is partitioned into multiple CUs ranging from $8 \times 8$ to $64 \times 64$ in size (Fig. 1 illustrates an example partitioning structure for a typical video frame) [4], and contain multiple PUs and TUs that maintain prediction and transform information. Although this greatly enhances the flexibility of the architecture itself, the analysis of encoder implementations however reveal that the addition of these novel features produces a high coding gain to the detriment of computational complexity [5].

Rate-Distortion (RD) optimization, the process of determining the optimum coding modes and structure for a given content (which leads to the highest attainable coding efficiency in the encoder), uses a Lagrangian cost function to parameterize the encoding efficiency. This cost function is expressed as,

$$\min_p \left\{ D(p) + \lambda R(p) \right\}, \quad p \in P \tag{1}$$

where $\lambda \geq 0$ is the Lagrange multiplier, $p$ is a coding parameter combination from the set of all the possible coding options $P$, and $D(p)$, $R(p)$ are the distortion and rate associated with the selected set of coding parameters, respectively [6]. The process of selecting the optimum set of coding parameters, using the Lagrangian cost function in (1), is considered to be a major source of complexity in encoder implementations, due to the large number of possible combinations of the CU sizes, PU modes and other coding parameters [5]. The complexity increase in HEVC, with respect to both inter- and intra-prediction, therefore directly impacts the encoding performance; thus, the recent literature predominantly investigate mechanisms to minimize the number of RD evaluations and the encoding time, while retaining the coding performance. Moreover, minimizing the complexity associated with the coding structure determination and the motion estimation in inter-prediction is seen as a more pressing requirement as it encompasses a larger portion of the encoding time [5], [7].

B. Related Work

Reducing the computational complexity associated with inter-prediction in the HEVC architecture can be attempted for a range of operations, such as motion estimation, coding structure determination, filtering, etc. In fact, the complexity analysis presented in [7] suggests that optimizing the motion estimation alone only leads to a minimal reduction of the encoding time. However, the optimization of the different operations are not mutually exclusive, and could be utilized in conjunction with coding structure determination methods to achieve significant reductions of the encoding time. In this context, a summary of state-of-the-art is presented in the Table I to illustrate the operational basis of each algorithm.

It can be observed here that optimizing the PU level mode decision is one of the more popular complexity reduction methods. In addition to the approaches summarized in Table I, this can also encompass methods that use spatio-temporal correlations and inter-level mode information [10], RD cost prediction [12], and the dynamic encoder parameter selection [15]. Furthermore, in the recent past, the SKIP mode decision has been used extensively to terminate the PU mode evaluation (Table I). However, the common theme in these works is the fact that they focus on the PU mode decision and do not consider the CU size decision (it should be noted that they can still be used in conjunction with a CU size selection algorithm); thus, every CU depth level must be evaluated in order to determine the optimum CU size. In practice, experimental results [13] suggest that the potential exists for further reduction of the encoding complexity than what has already been achieved.

Crucial to this problem is the optimal selection of the CU size in the coding hierarchy; hence, Table I also summarizes the relevant state-of-the-art CU size selection algorithms and discusses their merits and demerits. Here, it is observed that these CU size selection algorithms predominantly rely on the statistics gathered from the evaluation of the Inter $2N \times 2N$, SKIP, or other PU modes, can be enhanced further. This is especially true in the case of textured sequences with complex
TABLE I
A SUMMARY OF THE STATE-OF-THE-ART QUADTREE STRUCTURE OPTIMIZATION SCHEMES USED FOR FAST HEVC ENCODING

<table>
<thead>
<tr>
<th>Reference</th>
<th>Algorithm Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gweon et al. [8]</td>
<td>Uses the Coded Block Flag (CBF) to skip the evaluation of remaining PU modes.</td>
<td>The complexity reduction achieved is limited since only PUs are being considered and the lower likelihood of the CBF condition being satisfied in complex sequences.</td>
</tr>
<tr>
<td>Vanne et al. [9]</td>
<td>Optimizes the PU mode selection between Symmetric (SMP) and Asymmetric Partitions (AMP). The method selectively skips PU mode evaluations based on the merge and SKIP modes.</td>
<td>Complexity is reduced with respect to [8] due to the early determination of PU modes that can be skipped. However, the complexity reduction is less significant when the number of skipped PU modes are lower. The performance is seen to degrade for complex and textured content which generally results in fewer merge and SKIP mode selections. A similar behavior is observed at lower QPs.</td>
</tr>
<tr>
<td>Sampaio et al. [11]</td>
<td>A Motion Vector Merge (MVM) algorithm that evaluates the Inter N×N mode to determine which SMP PU modes to evaluate.</td>
<td>The computational complexity reduction is limited due to the smaller number of PU evaluations that are skipped.</td>
</tr>
<tr>
<td>Yang et al. [14]</td>
<td>Early Skip Mode Detection (ESD) assesses the SKIP mode selection of a CU and determines when to skip the remaining modes.</td>
<td>The approach is more suited for less complex sequences encoded at higher QPs where the SKIP mode is more likely to be the preferred coding mode.</td>
</tr>
<tr>
<td>Shen et al. [16]</td>
<td>Utilizes the neighboring and co-located CU information to determine and skip unnecessary CU depth levels.</td>
<td>The algorithm demonstrates good RD performance and computational complexity reductions. However, a large variance in the achievable complexity reduction is observed with respect to the QP and content types. Use of higher QPs and encoding of less complex contents results in the best performance due to a greater prevalence of SKIP and merge modes. Moreover, the evaluation of the current depth level becomes futile in the event the CU requires further splitting, such as in the case of low QP encoding of complex content.</td>
</tr>
<tr>
<td>Lee et al. [13]</td>
<td>Employs RD cost statistics of Inter 2N×2N mode evaluations, and the status of SKIP and merge modes, to determine the CU early termination depths. A SKIP Mode Decision (SMD) algorithm makes use of the SKIP mode selection statistics to skip both PU modes and CU depth levels.</td>
<td>The threshold comparison values (which are not adapted to the QP or to the content) and decision tree topology derived from offline training is less adaptable to changing content; thus, more dynamic video sequences and sequences with scene changes exhibit degraded RD performance. Furthermore, the early termination conditions evaluated at the end of each CU depth results in a reduction of the encoding time performance at high bit rates.</td>
</tr>
<tr>
<td>Shen et al. [17]</td>
<td>CU split decisions are calculated using a Bayesian decision rule based on a feature set collected from the Inter 2N×2N mode.</td>
<td>The optical flow calculation within the encoding loop, that is used to extract motion, can be both time consuming and resource intensive [21].</td>
</tr>
<tr>
<td>Shen et al. [18]</td>
<td>CU early termination decisions based on an offline Support Vector Machine (SVM) trained algorithm.</td>
<td>The optical flow calculation within the encoding loop, that is used to extract motion, can be both time consuming and resource intensive [21].</td>
</tr>
<tr>
<td>Correa et al. [19]</td>
<td>Early CU termination using the offline trained decision trees and threshold values.</td>
<td>The optical flow calculation within the encoding loop, that is used to extract motion, can be both time consuming and resource intensive [21].</td>
</tr>
<tr>
<td>Xiong et al. [20]</td>
<td>Uses Pyramid Motion Divergence (PMD) features calculated from the optical flow of downsampled frames for CU size selection.</td>
<td>The optical flow calculation within the encoding loop, that is used to extract motion, can be both time consuming and resource intensive [21].</td>
</tr>
</tbody>
</table>

motion where high quality is required; an area where existing implementations show some deficiency, and is a scenario that must be addressed to operate at a wide range of bit rates. Moreover, a content adaptive operation is crucially important to cater for the diversity of the content; thus, the capacity for dynamic training and content-specific feature extraction also becomes a necessity to improve the general performance of the algorithms.

III. CU CLASSIFICATION FOR SPLIT LIKELIHOOD MODELING

The CU sizes resulting from the splitting decisions attained through the RD optimization are highly dependent on the nature and the complexity of the content. Therefore, predicting the CU size beforehand, using a set of pre-determined features, becomes challenging due to this dynamic nature of the problem. In this context, it becomes evident that the early determination of the CU size requires a modelling of the CU split likelihoods of that particular content using a set of content-specific features. The selection of appropriate features that accurately model the CU split decision, that are also easy to extract from the encoding chain, is therefore crucially important. Two dynamic content-specific techniques that can be used for this purpose are described next.

A. Motion Feature-Based CU Classification

The first model, based on the motion feature-based CU classification approach [23], [24], attempts to represent the CU split likelihood as a function of three parameters given by

$$f(F) := f(\alpha, \beta, \omega),$$

where $\alpha$, $\beta$, and $\omega$ represent a motion classification, an Inter $N\times N$ RD cost category and the CU size, respectively, and $f$ denotes a probabilistic model that determines the outcome of the CU split decision and is described in Section IV.

First, observing the partitioning behavior of CUs during inter-prediction, it can be observed that blocks with similar motion tend to utilize larger CUs, whereas blocks with complex motion tend to utilize smaller CUs [11], [20], [25]. Attempting to classify these characteristics, from the information available within the encoding chain itself, therefore becomes attractive due to both its simplicity and its minimal impact on the complexity. To this end, this paper proposes an initial Inter $N\times N$ mode evaluation (skipping the traditional PU evaluation
This results in a RD cost category $\beta$ given by,

$$
\beta = \begin{cases} 
\sqrt{\frac{\gamma}{\Delta}} + \frac{1}{2} & \gamma \leq 200\Delta \\
200 & \text{otherwise.}
\end{cases}
$$

Finally, the CU size $\omega$ can be used as a third parameter that describes the CU split likelihood. The relationship between $\alpha_1$, $\beta$ and $\omega$, and the motion-based feature $F$ in (2), however remains complex and content dependent as illustrated in Fig. 3. For example, it can be observed that each $\alpha_i$ exhibits diverse CU split likelihoods for different $\beta$ in Fig. 3(a) and $\omega$ in Fig. 3(b), while larger $\beta$ favours CUs with larger $\omega$ to be split in Fig. 3(c). Thus, although the motion feature-based approach possesses the necessary flexibility to model the dynamic nature of the content, a separate decision making process (described in Section IV-A) is still required.

### B. RD Cost Threshold-Based CU Classification

In contrast to the motion feature-based approach, the CU split likelihood can also be modelled in a partly heuristic fashion. This second model investigates its relationship with respect to a general distribution of the $Inter\ N \times N$ mode RD cost $\gamma$, the CU size $\omega$ and the Quantization Parameter $QP$. First, observing multiple video sequences reveals that the CU splitting behaviour can be modelled by two Gaussian distributions; a CU split and non-split likelihood distribution. The motion feature-based approach (described in Section IV-A) is still required.

To this end, the distribution of $HTh_{spt}$ and $HTh_{nspt}$ is analyzed for sequences encoded using the low delay $P$ configuration and $QP$ in $\{22, 27, 32, 37\}$ in HM16.0. Despite some variations, from the results observed in Fig. 5, it is evident that an exponential curve can parameterize the behaviour of these two RD cost thresholds. Thus, generalized RD cost thresholds can be obtained, which are given by

$$
HTh_{spt} = \begin{cases} 2347\times e^{0.1248\times QP}, & \omega = 64 \\
851.2\times e^{0.1228\times QP}, & \omega = 32 \\
279.9\times e^{0.1227\times QP}, & \omega = 16
\end{cases} 
$$

$$
HTh_{nspt} = \begin{cases} 736.2\times e^{0.1378\times QP}, & \omega = 64 \\
225.4\times e^{0.1468\times QP}, & \omega = 32 \\
57.72\times e^{0.1607\times QP}, & \omega = 16
\end{cases} 
$$

Table III summarizes the R-squared measure of the goodness-of-fit obtained for $HTh_{spt}$ and $HTh_{nspt}$ in (5) and (6). The modelled curves and the data illustrated in Fig. 5 suggest that the proposed models for $HTh_{spt}$ and $HTh_{nspt}$
Proportion of CUs that are split in each motion category \((\alpha_i)\) for each \(\beta\). (Averaged over CU sizes 64, 32 and 16).

Fig. 3. Distribution of normalized proportion (%) of CUs that are split across \(\beta\), motion category \((\alpha)\) and CU size \((\omega)\) for 200 frames in “City (720p)” video sequence when encoded with QP=27 using low delay P main configuration in HM 16.0. Results depict the CU split likelihoods for the different combinations of the parameters \(\alpha\), \(\beta\), and \(\omega\) in the feature vector \(F\).

(a) Normalized proportion of CUs that are split in each CU size \((\omega)\) for each motion category \((\alpha_i)\). (Averaged over entire range of \(\beta\)).

(b) Normalized proportion of CUs that are split in each CU split and non-split likelihood distributions of the “ParkScene” sequence when encoded with QP=32 using low delay P main configuration in HM 16.0. (b) A representation of the \(I^H_{th}\) and \(I^H_{thact}\) RD cost thresholds that identifies the CU split, CU non-split and ambiguous regions.

Fig. 4. (a) CU split and non-split likelihood distributions of the “ParkScene HD” video sequence for QP=32 using the low delay P main configuration in HM 16.0. (b) A representation of the \(I^H_{thact}\) and \(I^H_{thact}\) RD cost thresholds that identifies the CU split, CU non-split and ambiguous regions.

TABLE III

<table>
<thead>
<tr>
<th>CU Size</th>
<th>(HT_{thact})</th>
<th>(HT_{thact})</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>0.827</td>
<td>0.782</td>
</tr>
<tr>
<td>32</td>
<td>0.703</td>
<td>0.906</td>
</tr>
<tr>
<td>16</td>
<td>0.780</td>
<td>0.882</td>
</tr>
</tbody>
</table>

are good representations in general, yet, are not particularly accurate, especially at higher QP values given the content specificities. Obtaining these content specific thresholds is however crucially important for an accurate optimal CU size prediction. In essence, threshold values in (5) and (6) could be used as initialization parameters during the preliminary stages of the encoding, and can later be adapted with content specific data as described in Section V.

IV. FAST CU SIZE SELECTION

This section describes how the two independent CU split likelihood models in Section III can be used in a complementary fashion, to determine if a CU should be split or not, all the while adapting to the specific content being encoded.

A. Motion Feature-Based CU Size Selection

Applying the feature-based CU classification model, the split probability of a CU in the \(n^\text{th}\) frame can be defined as

\[
P_{s,n}(F) = \frac{D^1_{act}(F)|_n}{D^1_{act}(F)|_n + D^0_{act}(F)|_n},
\]

where \(D^1_{act}(F)|_n\) is the number of CUs with a feature vector \(F\), within the given frame that are either split \((\eta = 1)\) or not-split \((\eta = 0)\), based on the actual split decision obtained for the CUs by the RD optimization. Thus, \(P_{s,n}(F)\) can be considered a frame-wise statistic (calculated at the end of each encoded frame) of the optimal split decision computed from the statistics accumulated in \(D^1_{act}(F)|_n\) that is obtained for each feature vector during the training phases described in Section V-B.

However, since this statistic can vary over time due to changes in the underlying content, a snapshot of the actual split probability is obtained through a windowed averaging process across the split probabilities calculated for the individual frames (i.e., \(P_{s,n}(F)\)). Mathematically, this can be expressed as

\[
P_{s}(F)|_n = \frac{1}{\tilde{W}} \sum_{t=0}^{n-1} P_{s,t}(F) H(t - n),
\]

where the window function \(H(t)\) is given by

\[
H(t) = \begin{cases} 
1 + \frac{t}{\tilde{W}} & 0 \leq t \leq -\tilde{W} \\
0 & \text{otherwise},
\end{cases}
\]

and \(\tilde{W}\) represents the area under the curve of \(H(t)\). The window function and its effective length \(W\) are crucial in terms of the content adaptability, as it enables the predicted CU split decision to be biased to the most recent \(W\) frames. Therefore, by appropriately selecting \(W\), the prediction can be made content adaptive and less susceptible to scene dynamics.

The outcome of the decision of whether to split or not-split a CU is now obtained by comparing (8) with an empirically determined threshold \(T\), such that the decision for the \(n^\text{th}\) frame is given by

\[
D_{fs}|_n = \begin{cases} 
1 & P_{s}(F)|_n \geq T \\
0 & \text{otherwise}.
\end{cases}
\]

The threshold \(T\) therefore acts as a switch that either splits or does not split the CUs. Empirical observations reveal that

\[
T = \max_{\beta} \min_{\omega} \left( P_{s}(F) \right).
\]

It should be noted that, in order to gather the statistics of the actual splitting behaviour of the content being encoded, the RD optimization in (1) can not be completely bypassed. Thus, the outcome of the split decision must be evaluated using an appropriate balance of either the two models presented in this paper or the traditional RD optimization approach. Precisely how this is implemented to achieve the fast coding objective, is described in Section V-B.
the value of $T$ impacts both the bit rate and quality, where a smaller value of $T$ generally results in more CUs being split, while a larger $T$ results in less splitting. In this context, $T$ and the window length $W$ can be considered as design parameters that need to be empirically determined and preset for a desired trade-off of the quality and the bit rate.

B. RD Cost Threshold-Based CU Size Selection

In the RD cost threshold-based CU classification approach, the Inter $N \times N$ RD cost $\gamma$ is compared to the $HTh_{spt}$ and $HTh_{nspt}$ thresholds for the $\omega$ and $QP$ relevant to each CU. The resulting split decision $D_{hs}$ of the $n^{th}$ frame can therefore be expressed as

$$\begin{align*}
D_{hs}|_{n} = \begin{cases} 
1, & \gamma \geq HTh_{spt} \\
0, & \gamma \leq HTh_{nspt}.
\end{cases}
\end{align*}$$

However, as described in the previous section, $HTh_{spt}$ and $HTh_{nspt}$ need to be made adaptive in order to become content-aware, and a heuristic process can be used to adapt the generic $HTh_{spt}$ and $HTh_{nspt}$ thresholds in (5) and (6).

First, in order to adapt the split and not-split thresholds to the content, the RD cost statistics of the actual split decision (similar to the feature-based model in (7)) are analyzed in terms of the mean and standard deviation of these thresholds for a particular CU size $\omega$ and Quantization Parameter $QP$. Adopting a window-based approach to maintain the content adaptability as before, the mean and standard deviation statistics during the $n^{th}$ frame for the split RD cost threshold can be expressed as

$$\begin{align*}
\mu^{spt}|_{n,\omega,QP} &= \frac{1}{W} \sum_{t=0}^{W-1} E \left[ \gamma^{spt}(n-t)|_{\omega,QP} \right],
\sigma^{spt}|_{n,\omega,QP} &= \sqrt{\frac{1}{W} \sum_{t=0}^{W-1} E \left[ \left( \gamma^{spt}(n-t)|_{\omega,QP} - \mu^{spt} \right)^2 \right]},
\end{align*}$$

respectively. $E(\cdot)$ represents the expectation operating on the applicable CUs in that frame, and $\gamma^{spt}(\cdot)|_{\omega,QP}$ represents the RD cost of the CUs that are split with a CU size of $\omega$ and a Quantization Parameter $QP$. The RD cost threshold statistics for the not-split scenario, $\mu^{nspt}|_{n,\omega,QP}$ and $\sigma^{nspt}|_{n,\omega,QP}$, can be obtained in a similar fashion.

Thus, the two thresholds themselves are made content adaptive by applying the following:

$$\begin{align*}
HTh_{spt}|_{n,\omega,QP} &= \left\{ \mu^{spt} + \tau \times \sigma^{spt} \right\}|_{n,\omega,QP},
\end{align*}$$

and

$$\begin{align*}
HTh_{nspt}|_{n,\omega,QP} &= \left\{ \mu^{nspt} - \tau \times \sigma^{nspt} \right\}|_{n,\omega,QP}.
\end{align*}$$

The parameter $\tau$ acts as a governor that controls the adaptation of the model via the adaptation and training process described in Section V-B, and is an empirical design parameter that can be used to trade-off the computational complexity for the coding efficiency in the proposed encoding algorithm.

V. PROPOSED FAST ENCODING FRAMEWORK

In this section, computing the ultimate CU split decision, using the two independent decisions in the previous section, is described. This is followed by a mechanism to exploit the Inter $N \times N$ mode motion characteristics obtained during the CU split likelihood modelling, to supplement the CU size selection algorithm and further expedite the encoding process.
A. Joint CU Split Decision Prediction

The approach to using the two independent decisions described in Section IV forms the basis of the proposed fast encoding algorithm. Thus, when obtaining the joint decision, two distinct categories exist; one where both independent split decisions concur, and another where they differ. Hence, for the former category, the joint split decision during the \( n \)-th frame predicted by the encoding framework can be expressed as

\[
|CU_n| = \begin{cases} 
1 & D_{fs}|_n = 1 \land D_{hs}|_n = 1 \\
0 & D_{fs}|_n = 0 \land D_{hs}|_n = 0,
\end{cases}
\]

where \( D_{fs}|_n \) and \( D_{hs}|_n \) are the two independent decisions obtained for the applicable \( F \), \( \gamma \), \( \omega \) and \( QP \) of each CU. The second category of decisions, where the models differ, can now be used to initiate the adaptation of the framework to enhance its robustness to different contents.

B. Model Adaptation and Training

Following from the discussion of the joint split decision prediction, it is crucial that the models are able to adapt; thus, some RD evaluation becomes essential to calculate the actual CU split statistics in Section III. The proposed algorithm therefore consists of multiple training phases that facilitate the gathering of these statistics, such that the expected performance gains are not compromised. These are described in detail below.

1) Initial Training: Both split likelihood models described in Section III require some initial training\(^4\) to gather content specific data at the beginning of a video sequence. In this work, the first four frames, i.e., \( n = 1, \ldots, 4 \), are used for this statistical information gathering. During this phase, the CU split decisions are obtained via the traditional RD optimization; thus, the two models are initialized with sufficient content specific data. However, these models may diverge from the actual content due to changes in the scene, making the accumulated statistics less relevant with time. Hence, the models must be continually refreshed, via intermediate training, as described next.

2) Intermediate Training: Intermediate training of the proposed framework can be split into three categories; training where no data exists for the features associated with a CU, training where the two models’ decisions differ, and training for modelling efficiency improvements.

The first type of intermediate training is triggered when the actual information required to compute \( P_s(F)|_n \) does not exist (e.g., a situation where the feature \( F \) has not been encountered within the window length \( W \)). Similarly, the RD optimization is followed in the event that \( D_{fs}|_n \) in (10) and \( D_{hs}|_n \) in (11) contradict each other. In both cases, the additional information obtained regarding the actual splitting behaviour will result in the refinement of both split likelihood models, thereby improving the accuracy of the subsequent decisions of similar CUs.

\[^4\]The RD cost threshold-based model can still utilize the generic values of the \( HT_{h_{spt}} \) and \( HT_{h_{nspt}} \) thresholds in (5) and (6), respectively, until such content specific statistics are accumulated.

The third and most important intermediate training phase is triggered by RD costs where \( HT_{h_{nspt}} < \gamma < HT_{h_{spt}} \) (see Fig. 4(b)), and the CU split decision of the RD cost threshold-based model \( D_{hs}|_n \) is undefined as per (11). Controlling the size of this region will result in a trade-off of the output quality for the reduction of the computational complexity. Hence, in addition to facilitating the statistics gathering, the complexity control parameter \( \tau \) (introduced for this purpose in (14) and (15)) affords a degree of flexibility when implementing the overall algorithm. Here, a larger \( \tau \) will expand this region, allowing more decisions to be taken by the RD optimization, resulting in better quality. A smaller \( \tau \) on the other hand will reduce the number of the RD optimizations and will result in improvements to the encoding time performance.

C. Motion Vector Reuse in Motion Estimation

This subsection describes how the motion vectors computed during the \( Inter \times \times \times \) mode evaluation can be reused to further expedite the encoding process during the subsequent PU level evaluations that repeatedly occur for each CU.

Consider the four constituent motion vectors and reference frames returned by the initial \( Inter N \times \times \times \) mode evaluation in (3). These motion vectors identify the motion category of a CU, \( \alpha_i \), which interestingly has a structural relationship with the PU modes as illustrated in the Fig. 6. This relationship can be exploited to skip the motion estimation when a particular PU mode is being evaluated for a CU. For example, when a CU possesses the motion category \( \alpha_0 \) (i.e., all four motion vectors are equal and point to the same reference picture), the motion vector \( MV_0 \) available in (3) can be reused for the \( Inter 2N \times 2N \) mode, thereby skipping the motion estimation phase for that PU mode. However, not all PUs can be identified in this fashion; thus, some PUs require motion estimation, e.g., the PUs denoted by “ME” and the PU modes that are not illustrated in Fig. 6 will require the usual motion estimation.

Hence, this capability to reuse the motion information extracted during the CU size prediction emerges as a direct secondary benefit of the initial \( Inter N \times \times \times \) mode evaluation. As a result, the proposed framework is supplemented with this feature to further expedite the encoding process.

D. The Overall Fast Encoding Algorithm

The performance improvements of the fast encoding algorithm proposed in this paper can be described as a result of two distinct operations; the content-adaptive CU size prediction in Section V-A and the motion reuse operation in Section V-C, that both exploit the initial \( Inter N \times \times \times \) mode evaluation. A high level flow diagram of the resulting algorithm, identifying the major decision making components and the operations of the individual blocks, is summarized in Fig. 7.

At an implementation level, if the CU split decision is in the negative, the CU is encoded at the selected depth level. However, if it is positive, the encoding cycle evaluates the next depth level of the CU. During the first \( N \) frames for the sequence, and whenever the CU split decision cannot be predicted, the traditional RD optimization is triggered via the initial and intermediate training processes described in
Section V-B. The statistics calculated during these periods are simultaneously used to update and refine the split likelihood models as described earlier. The shaded area in the Fig. 7 depicts the PU mode selection operation. Here, the available PU modes of the CU are evaluated in the traditional evaluation order [26], and the best PU mode is selected using an RD optimization. However, the motion estimation phase for a subset of PUs is skipped and the Inter $N\times N$ motion vectors are reused where appropriate as described in Section V-C.

VI. EXPERIMENTAL RESULTS AND DISCUSSION

The following section presents the experimental results of the proposed content-adaptive fast CU size selection and encoding algorithm for low delay HEVC video encoding. The RD and encoding time performance of the proposed algorithm are compared with several state-of-the-art algorithms in the literature. These include the HM 16.0 [26] reference implementation, the CU size selection method proposed by Shen et al. [16], the fast encoding algorithms proposed by Lee et al. [13], the fast block partitioning algorithm proposed by Lu et al. [29], two versions of PU mode decision algorithms proposed by Vanne et al. [9], and the offline data mining approach to CU early termination proposed by Correa et al. [19].

A. Simulation Configurations and Performance Metrics

The algorithms are evaluated for a range of HD and UHD video sequences composed of both natural and synthetic content ranging from simple to highly complex motion with diverse spatial and temporal characteristics. Table IV summarizes the experimental setup and encoding configurations.

The impact on the RD performance is evaluated using the Bjøntegaard Delta Bit Rate (BDBR) [27] and the average

5The CU classification models proposed in this paper are based on the motion features extracted from the preceding video frames. Adopting the same approach for the Random Access configuration must also consider the impact of future frames and is therefore outside the scope of this work.
TABLE IV  
SIMULATION SETUP AND CONFIGURATIONS

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>QPs</td>
<td>22, 27, 32, 37</td>
</tr>
<tr>
<td>Encoding Configurations</td>
<td>Low Delay P Main, Low Delay B Main</td>
</tr>
<tr>
<td>HEVC software version</td>
<td>16.0</td>
</tr>
<tr>
<td>Video sequence types</td>
<td>HD, UHD</td>
</tr>
<tr>
<td>Frame rates</td>
<td>HD: 30 fps, UHD: 50 fps</td>
</tr>
<tr>
<td>Number of frames</td>
<td>200</td>
</tr>
<tr>
<td>Machine</td>
<td>Intel Core i5, 8GB RAM, Ubuntu 14.04 LTS</td>
</tr>
</tbody>
</table>

Fig. 8. The effect of the window length $W$ (when $\tau = 0$) on the coding efficiency and the encoding time reduction of the proposed encoding algorithm.

Fig. 9. The effect of the complexity control parameter $\tau$ (when $W = 30$) on the coding efficiency and the encoding time reduction of the proposed encoding algorithm.

percentage encoding time saving, $\Delta T$, is evaluated for the proposed and state-of-the-art algorithms by comparing the implementations of the respective algorithms with the HM 16.0 reference software [26]. In this context, $\Delta T$ is given by,

$$\Delta T = 100 \times \frac{T_{HM} - T_{\rho}}{T_{HM}},$$  

(17)

where $T_{HM}$ is the encoding time of HM16.0 and $T_{\rho}$ is the encoding time required for each fast encoding approach.

B. Results and Performance Analysis

The window length $W$ and the complexity control $\tau$ parameters defined in Section IV will naturally affect the performance of the proposed algorithm. This section first discusses how they can be selected and their impact, and is followed by an analysis of the overall performance and implications of using the proposed fast coding framework.

1) Window Length and Complexity Control Parameter Selection: The experimental results illustrate the performance impact of $W$ on $\Delta T$ and BDBR in Fig. 8 for HD sequences. Here, the value of the complexity control parameter is set as $\tau = 0$, which results in $HTh_{spt}$ and $HTh_{nsp}$ being exactly similar to the mean $\gamma$ of the CU split, and non-split Gaussian distributions (as seen in Fig. 4 the mean values are already good approximations for these thresholds). It can be observed that the BDBR improves with smaller window sizes, while $\Delta T$ increases with increasing window size and vice-versa. Intuitively, this is due to less training being required for long $W$, yet longer window lengths also suggest less adaptability and sub-optimal quality. Evidently, the opposite is true when the $W$ is shorter, where a smaller window size eventually increases the amount of training required, thus, resulting in a decrease in $\Delta T$ as well as BDBR. An empirically determined window size of $W = 30$ that both provides comparable average BDBR increases to that of the state-of-the-art algorithms and also facilitates the adequate accumulation of statistical data (in general, an average of approximately 25 training occurrences are observed for a typical feature vector), is used in the analysis in the remainder of this discussion.

The experimental results illustrated in Fig. 9 for different $\tau$ show that $\Delta T$ tends to increase with a corresponding increase of the BDBR. This is due to the fact that when the middle region is smaller, split decisions of more CUs become incorrect due to fewer RD optimization occurrences, which negatively impacts BDBR but improves $\Delta T$. Naturally, when the region becomes larger the opposite is true, and is reflected in Fig. 9. Hence, the performance results of the proposed algorithm discussed in the remaining sections use $\tau = 0$, which together with the previously selected $W$, corresponds to BDBR increases comparable to the state-of-the-art solutions.

In addition, as discussed in Section IV-A, the threshold value $T$ in (10) decides the portion of CUs that will be decided to split, for a given feature vector $F$. In this context, an empirically determined value of $T$ ($T = 0.6$) is used as the threshold, by considering it’s impact on the RD efficiency.

2) Overall Performance of the Proposed Algorithm: The performance of the proposed algorithm is presented in the Tables V - VII for the low delay P and low delay B configurations. These results first address the impact of only the CU size selection aspect (described in Section V-A and V-B) denoted by SI. Then in SII the impact of including the motion vector reuse from the Inter $N \times N$ mode evaluation (described in Section V-C) is evaluated. The following discussion further analyses these results in terms of the variations seen for different content types, QPs and other relevant attributes.

a) Performance Variation with QP: Examining the encoding time performance results illustrated in the Table V for a subset of sequences representing diverse content types and QPs ($QP = 22, 27, 32, 37$), a variation in $\Delta T$ with the corresponding bit rate of the video sequences can be observed. For example, $\Delta T$ tends to increase with the decreasing bit rates (i.e., increasing QP) for the proposed as well as the state-of-the-art algorithms. In general, this behaviour can be explained as follows. Typically, when encoding a CU at larger QPs, larger CUs and prediction modes such as SKIP and merge modes are favoured; thus, the algorithms that early terminate a CU at smaller depths and early detect SKIP/merge modes, demonstrate an increased $\Delta T$ compared to smaller QPs that yield smaller CUs and fewer SKIP mode PUs.

However, interestingly, the variation of $\Delta T$ with the QP is
relatively large for the methods proposed by Lee et al. [13], Shen et al. [16] as well as Correa et al. [19]. This is due to the evaluation of the CU Early Termination (ECUT) and CU Skip Estimation (CUSE) conditions [13] that require the encoder to evaluate the current CU prior to determining whether the CU should be split further. Similarly, the verification of the decision trees introduced in [19] is performed as the final operation at a particular depth level, and results in a similar behavior. Here, video sequences that tend to use smaller CUs when using a lower QP will result in the algorithm unnecessarily evaluating the upper CU depth levels before early termination. In contrast, the proposed algorithm predicts the CU split decision prior to the encoding of a CU; thus, the unnecessary evaluations of larger CUs are avoided. This leads to far less performance variation between QPs, and suggests that the proposed algorithm is suitable for encoding videos across a wider range of bit rates unlike the existing approaches.

b) Performance Variation with the Content: It can be observed that all the state-of-the-art algorithms demonstrate a relatively large encoding time reduction for less textured sequences such as “Poznan Street” (Table VI and VII). This reduction in $\Delta T$ is mainly due to the skipping of rarely used CU depth levels [16] and the ECUT methods that have been employed [13], [29]. Hence, the less complex the content (less texturized and simple motions), the more likely it becomes that they will be encoded with larger CUs, causing the ECUT to exhibit an increased $\Delta T$. In addition, static or content with simple motions can exploit the CU depth range estimation algorithms in [16] to skip the unnecessary CU depth levels.

However, in the case of more textured sequences with average or low motions (e.g., “Kimono”, “Musicians” etc.), much smaller CU sizes are required; thus, the ECUT checks at upper depth levels become ineffective, leading to a much lower $\Delta T$. Moreover, the skipping of the rarely used CU depth levels in [16] eventually leads to relatively high coding losses, especially in the case of sequences that exhibit multiple localized motions (i.e., “Poznan Street”). The depth range estimations in this case often become less accurate, and the errors in these are propagated across the frame to further deteriorate the coding efficiency.

That said, the method proposed by Shen et al. performs reasonably well even with sequences that generally exhibit uniform motion across the frame (e.g., “Kimono”), to the detriment of the encoding time performance. In contrast to this approach, the proposed framework exhibits a performance that varies much less with the content. The effect of the content's
complexity therefore appears trivial to the proposed algorithm, due to its early prediction of the CU split decision prior to the actual encoding of the CU. Furthermore, the proposed algorithm only evaluates the selected CU depth level, thus, the encoding time consumed for unnecessary CU depth level evaluation is avoided, providing an edge over the prevailing state-of-the-art encoding solutions when encoding a wider range of video sequences.

However, for sequences such as “Traffic”, which has fast moving objects throughout the sequence, the proposed method and the state-of-the-art algorithms demonstrate a relatively large BDBR in the range of 4%. This suggests that the complex motions and rapidly changing content has caused the decision making algorithms to make fewer efficient decisions compared to the sequences with average motion complexity. One of the solutions envisioned in this case is to reduce the window size \( W \) to increase the content adaptability of the proposed algorithm. In this context, the proposed algorithm is sufficiently flexible in its design parameters to cater for the diversities of the video sequences. In addition to the joint utilization of the two independent models, and the complexity control parameter coupled with the intermediate training phases provide a more attractive training solution for the proposed framework in contrast to the use of fully RD optimized training frames at pre-defined intervals as proposed by Lee et al. in [13].

c) PU Level Complexity Reduction: The impact of skipping PU modes is illustrated in the Tables VI and VII, for two algorithms proposed by Vanne et al. [9]. The algorithm \( S_{14} \) which skips a limited number of PU modes in each CU depth level based on the selection of SKIP or merge modes, results in a smaller complexity reduction, whereas the algorithm \( S_{25} \) produces greater complexity reductions to the detriment of the coding efficiency. Generally, the BDBR increase is relatively small in PU level optimization algorithms that skip the evaluation of asymmetric or certain symmetric partitions, as is the case here. This is due to the higher tendency of the RD optimization based PU mode selection preferring the selection of the \textit{Inter} \( 2N \times 2N \) mode over the remaining PU modes at each CU depth level. However, the intermittent increases in the BDBR seen in the tables with respect to certain sequences stem from the ignorance of the remaining PU modes which eventually lead to the selection of less efficient CU and TU structures. Moreover, PU skip decisions derived based on the selection of SKIP and merge modes in [9], results in an increased \( \Delta T \) for less complex sequences encoded at lower quality levels (ref. Table V), whereas the opposite is true when the sequences are encoded at higher quality levels. In contrast, the proposed algorithm maintains a consistent encoding time reduction (\( \approx 50\% \)) across all quality levels due to its SKIP/merge mode agnostic decision making models which perform the CU split decision prediction.

d) Content Adaptability and Online Training: The content adaptability and the effectiveness of the online training used in the proposed algorithm are further corroborated when comparing its performance with that of the offline trained algorithm proposed by Correa et al. in [19], whose BDBR exceeds 10% for some sequences due to its fixed thresholds and decision trees [19], [30]. These coding losses emphasize the inherent drawback of using offline trained algorithms, QP and content agnostic RD cost thresholds, and rigid decision tree topologies [30] on previously unseen data sets. Furthermore, the overlooking of the ambiguous regions, which often exist in the CU split/non-split statistical distributions [30] (cf. Fig. 4), eventually lead to the less efficient split decisions in [19]; thus, degrade the RD performance. Consequently, retraining the algorithm to the new data sets is a potential solution, yet this may become a time consuming and tedious process.

In contrast, a relatively constant BDBR increase is observed for the proposed algorithm as a result of its content adaptive nature. For example, the content adaptability features of the proposed algorithm ensure that the data upon which the CU split decisions are made always correspond to the content being encoded. Moreover, the experimental results further reveal that the content adaptive approaches introduced in the proposed algorithm are far more effective in capturing the content specific information, as opposed to the state-of-the-art approaches which typically rely on spatial and co-located CU statistics [16] for content adaptation.
e) Motion Estimation Complexity Reduction: The effect of the proposed fast CU selection method when supplemented by the motion vector reuse mechanism to optimize the PU level motion estimation, is presented under “Proposed SI” in Tables VI and VII. In this case, the experimental results demonstrate an additional average encoding time saving of 3% and 4% for the low delay P and low delay B configurations, respectively. However, BDBR increases of 0.37% and 0.19% for the respective configurations are exhibited in comparison to the “Proposed SI” solution, which is mainly due to the skipping of certain motion estimation instances.

Finally, it should be noted that the computational cost of the training phases and the decision making stages are all included in the encoding performance results that are reported in this paper. Therefore, it is evident that the additional complexities introduced by the proposed algorithm are negligible in comparison with the significant time saving that can be achieved by incorporating these algorithms into the encoding cycle.

VII. Conclusion

This paper proposes a content adaptive fast CU size selection algorithm for HEVC based low delay video encoding. In this context, two CU split likelihood models (based on a motion feature-based and a RD cost threshold-based CU classification approaches) are introduced to model the CU split and non-split decisions. These models are dynamically generated and are continuously adapted using initial and intermediate training phases, such that they independently predict the split decision for a given CU. Moreover, the possibility of reusing motion vectors identified during the modeling stage, for motion estimation in the remaining PU modes, is also investigated to supplement the proposed algorithm.

One major conclusion to be drawn from this analysis is that the initial evaluation of the Inter N×N mode provides motion and complexity properties of the underlying CU, which can be used to classify a CU, in order to model the split likelihood. Furthermore, the use of two independent models facilitates the split decision refinement as well as the identification of when the models require training dynamically during the encoding cycle. The window based approach used in the model adaptation and decision making ensures that the resultant split decisions are content-adaptive and less susceptible to the dynamic variations such as scene changes; a non-trivial advantage over the state-of-the-art methods.

In conclusion, the simulation results for the proposed CU size selection and encoding algorithm reveal an average encoding time saving of 58% and 61% for the low delay P and low delay B configurations, respectively. Moreover, the experimental results reveal that the proposed encoding algorithms can achieve a relatively uniform average encoding time saving across a wide range of QPs and content ranging from low to highly complex textures and motion characteristics, due to its SKIP/merge mode agnostic early CU size prediction. The capacity of the proposed algorithm to maintain a consistent performance, in terms of both the encoding time saving as well as BDBR increase (which is 2.29% on average), across diverse content types and QPs is especially notable when considering the performance fluctuations observed in the state-of-the-art solutions. The future work will focus on extending algorithm to the remaining coding structures (i.e., PUs and TUs) and other configurations in order to further expedite the encoding process with minimal impact on the coding efficiency.

REFERENCES


Thanuja Mallikarachchi (S’12) received his B.Sc. (Eng.) degree with honors in Electronic and Telecommunication Engineering from University of Moratuwa, Sri Lanka in 2011. From 2011 to 2013, he was a Senior Engineer at Virtusa (pvt) Ltd., Colombo, Sri Lanka. He is currently pursuing his Ph.D. in the Centre for Vision, Speech and Signal Processing (CVSSP) at the University of Surrey, United Kingdom.

His research interests are in the areas of video coding, video communication and video processing.

Dumidu S. Talagala (S’11-M’14) received the B.Sc. Eng (Hons) in Electronic and Telecommunication Engineering from the University of Moratuwa, Sri Lanka, in 2007. From 2007 to 2009, he was an Engineer at Dialog Axiata PLC, Sri Lanka. He completed his Ph.D. degree within the Applied Signal Processing Group, College of Engineering and Computer Science, at the Australian National University, in Canberra, Australia, in 2013.

He is currently a research fellow in the Centre for Vision, Speech and Signal Processing at the University of Surrey, United Kingdom. His research interests are in the areas of video processing and coding, sound source localization, spatial soundfield reproduction, array signal processing, audio-visual signal processing, sparse sensing and convex optimization techniques.