Noname manuscript No.
(will be inserted by the editor)

Reducing the Complexity of a Multiview H.264/AVC and HEVC Hybrid Architecture

A. J. Díaz-Honrubia · J. De Praeter · J. L. Martínez · P. Cuenca · G. Van Wallendael

Received: date / Accepted: date

Abstract With the advent of 3D displays, an efficient encoder is required to compress the video information needed by them. Moreover, for gradual market acceptance of this new technology, it is advisable to offer backward compatibility with existing devices. Thus, a multiview H.264/Advance Video Coding (AVC) and High Efficiency Video Coding (HEVC) hybrid architecture was proposed in the standardization process of HEVC. However, it requires long encoding times due to the use of HEVC. With the aim of tackling this problem, this paper presents an algorithm that reduces the complexity of this hybrid architecture by reducing the encoding complexity of the HEVC views. By using Naïve-Bayes classifiers, the proposed technique exploits the information gathered in the encoding of the H.264/AVC view to make decisions on the splitting of coding units in HEVC side views. Given the novelty of the proposal, the only similar work found in the literature is an unoptimized version of the algorithm presented here. Experimental results show that the proposed algorithm can achieve a good tradeoff between coding efficiency and complexity.

Keywords H.264/AVC · HEVC · Multiview Hybrid Coding · CTU Splitting

A. J. Díaz-Honrubia, J. L. Martínez, P. Cuenca
Albacete Research Institute of Informatics (I3A), University of Castilla-La Mancha, Spain
E-mail: {Antonio.DHonrubia, JoseLuis.Martinez, Pedro.Cuenca}@uclm.es

J. De Praeter, G. Van Wallendael
Ghent University - iMinds - Multimedia Lab, Ledeberg-Ghent, Belgium
E-mail: {Johan.DePraeter, Glenn.VanWallendael@ugent.be}@ugent.be

1 Introduction

Nowadays, H.264 or Advance Video Coding (AVC) [15] is the most widely used video compression standard for High Definition (HD) video coding in general, and for 3D HD in particular, for which its *Multiview Video Coding* (MVC) extension [14] is used. However, in April 2015 the 3^{rd} edition of the *High Efficiency Video Coding* (HEVC) [16] standard was completed with four important extensions. This new edition of the HEVC standard with its extensions will greatly help the industry to achieve effective interoperability between products using HEVC, and it will provide valuable information to facilitate the development of such products.

The first extension is the *Scalability Extension*, known as SHVC [4], which adds support for embedded bitstream scalability in which different levels of encoding quality are efficiently supported by adding or removing layered subsets of encoded data. The second one is the *Multiview Extension* of HEVC, known as MV-HEVC [22], which provides an efficient representation of video content with multiple camera views and optional depth map information, such as that required for 3D stereoscopic and autostereoscopic video applications. MV-HEVC is one of the 3D video extensions of HEVC. The third extension is the so-called *Range Extension* (REext), which includes support for more color formats, while offering greater bit depths. Finally, a 3D extension has also been released. This extension allows an HEVC stream to include depth map layers for 3D video applications, and it represents the second 3D extension of HEVC.

Thus, on the one hand, the 3D video coding technology based on H.264/MVC either lacks high quality 3D perception or has a limited coding

efficiency compared with the new HEVC *High Efficiency Video Coding* (HEVC) standard. On the other hand, 3D HEVC-based techniques have a high coding efficiency, but are not supported by H.264/AVC decoders. Therefore, HEVC-based systems cannot immediately be incorporated in the network without the high cost of upgrading the existing network infrastructure (such as encoders, streaming servers, transcoders, etc.) and the decoder install base.

In order to enable a system which offers 3D functionality, a low overall bit rate, and compatibility with currently existing H.264/AVC-based systems, a multiview H.264/AVC and HEVC hybrid architecture was proposed in the context of 3D applications and standardized in [23]. The standardization of this hybrid architecture was aligned with the HEVC extensions by the MPEG. The architecture is hybrid in the sense that the base view and the other views apply a different encoding standard. This is achieved by combining H.264/AVC encoding for the base view and HEVC encoding for the other views. This architecture reduces the bandwidth by exploiting redundancy with the base view stream (which is decodable by already existing systems), while the functionality of those systems is maintained in the mid-term. It can be noticed that depth maps are not used for the purpose of this paper, since as the aim is to maintain interoperability, if a device cannot decode the HEVC views, it will very likely not be able to decode the depth maps either, since H.264/AVC did not include a specification about texture views plus depth maps [15].

In terms of Rate-Distortion (RD) performance, HEVC is able to double the RD compression performance of H.264/AVC [19]. However, this improvement comes at the cost of extremely high computational complexity and memory requirements during encoding [19]. In the case of a hybrid architecture with an H.264/AVC base view and two HEVC views, 34% of the bit rate is saved for the same quality as MVC while maintaining backward 2D-compatibility with existing devices [24]. HEVC includes multiple new coding tools (which affect the encoding time of the HEVC-based views), such as highly flexible quadtree coding block partitioning, which includes new concepts such as *Coding Tree Unit* (CTU), *Coding Unit* (CU), *Prediction Unit* (PU) and *Transform Unit* (TU) [21].

In order to greatly reduce the complexity of the encoding of the HEVC views, this paper presents an algorithm as part of a low complexity multiview H.264/HEVC hybrid architecture. A preliminary version of the algorithm was published in [8], but since then several improvements providing a better adaptation of the algorithm to the sequence contents and to the encoder configuration have been incorporated.

Thus, the proposed technique exploits the information gathered while the H.264/AVC center view is being encoded in this multiview hybrid architecture, and uses this information to accelerate the CTU splitting decisions in the HEVC side views by using a statistical Naïve-Bayes (NB) classifier to avoid an exhaustive RD Optimization (RDO) search of all possible CU sizes and PU modes. This algorithm takes two facts into account dynamically (i.e. while the HEVC views are being encoded): 1) the displacements of some objects between the views, trying to compensate for them dynamically; and 2) the calculation of different thresholds for the NB models according to the content of the sequence. Experimental results show that the proposed algorithm obtains a time reduction of 70% on average in the hybrid architecture, while in the best case it can achieve a time reduction of up to almost 75% without significant loss in RD performance (a 4.8% bit rate increment for 2 views to preserve the same objective quality, as shown in Section 4).

The remainder of this paper is organized as follows. Section 2 includes the technical background and related work which is being carried out on the topic. Section 3 introduces our proposed low complexity algorithm. Experimental results are shown in Section 4. Section 5 concludes the paper and includes ideas for future work.

2 Technical Background and Related Work

HEVC introduces new coding tools while improving others which were already used by its predecessor, namely H.264/AVC [21] [19]. These improvements notably increase compression efficiency. One of the most important changes affects picture partitioning. HEVC discards the terms of Macro-Block (MB), Motion Estimation (ME) block, and transform block to respectively replace them by three new concepts: CU, PU and TU. Each picture is partitioned into square regions of variable size called CUs, which replace the MB structure of previous standards. Each CU, whose size is limited to between 8x8 and 64x64 pixels, may contain one or several PUs and TUs. To determine the size of each CU, a picture is divided into 64x64 pixel areas, which are called Coding Tree Units (CTU), and then each CTU can be partitioned into 4 smaller sub-areas of a quarter of the original area. This partitioning can be performed with each

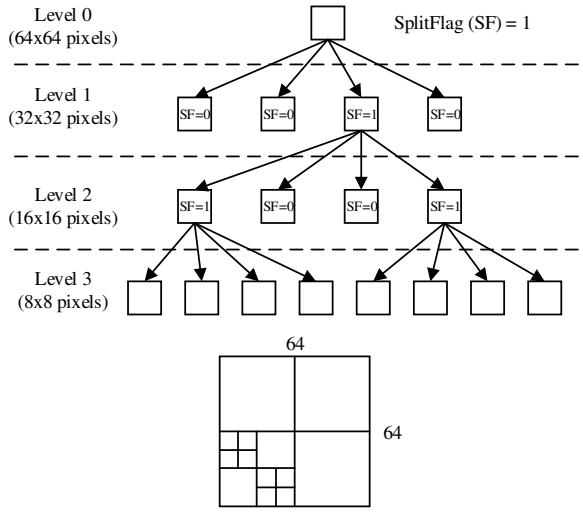

Fig. 1 CTU splitting illustration.

sub-area recursively until it has a size of 8x8 pixels, as is depicted in Figure 1.

HEVC may check up to eight possible PU partitions for each CU size to determine the optimal trade-off between rate and distortion. Furthermore, in the case of inter prediction, for each of these PU partitions an ME algorithm is called. This wide range of possibilities makes HEVC much more computationally expensive than H.264/AVC. HEVC introduces changes in other modules too, such as Intra Prediction (where a total of 35 different coding modes can be selected), PU modes (it introduces asymmetric modes), new image filters and new transform sizes.

In relation to H.264/AVC, two compatibility scenarios can be distinguished, and both hybrid architectures are proposed in [23]. The first scenario maintains backward compatibility with monoscopic video (H.264/AVC), whereas the second scenario targets backward compatibility with MVC and frame compatible coding. The former, allowing backward compatibility with H.264/AVC, results in a system where the base view of 3D video can still be transmitted using current 2D technologies and therefore no separate broadcasting infrastructure for 2D and 3D is required. The latter introduces backward compatibility for stereoscopic 3D. This allows 2D and stereoscopic 3D systems to remain operational while additional 3D video data is transmitted, without the need for a separate 3D broadcasting service. Both proposed systems are unlike fully HEVC-based 3D video. For fully HEVC-based 3D scenarios, a simulcast transmission of H.264/AVC and MVC bitstreams is required. Therefore, the encoding complexity is limited since the encoder only has to encode the center view once

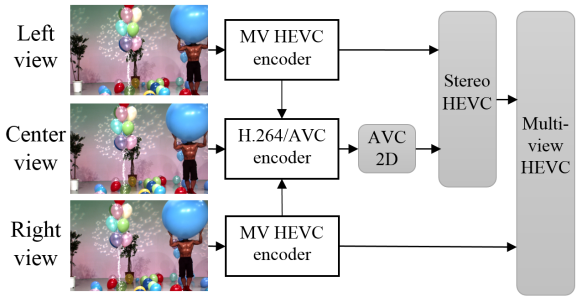

Fig. 2 Hybrid multiview architecture.

(for H.264/AVC instead of for both H.264/AVC and HEVC). Furthermore, for the decoder side a hybrid architecture will also reduce the complexity.

For monoscopic compatibility, the current H.264/AVC infrastructure (network infrastructure, access networks, set-top boxes, decoders, storage systems, etc.) can still be used for 2D video delivery. Meanwhile, new or upgraded decoders are able to decode the full 3D bitstream such that autostereoscopic displays, for example, can generate synthesized views. Figure 2 shows the proposed hybrid architecture for multiview video with three views, where compatibility with monoscopic and stereo video is maintained [23]. The center view is encoded using H.264/AVC. The decoded center view output is used for inter-view prediction by both side views. Therefore, the side HEVC encoders have an additional reference picture available that can be used for prediction, as was the case for MV-HEVC. The decoded center view picture is stored in a shared memory buffer, which is accessible for the left and right views. The HEVC encoder indicates with a flag for each PU whether inter-view prediction is used or not. On the decoder side, the decoded center view will be used by the HEVC decoders to add the decoded residual data to the current view data. This inter view prediction flag is transmitted for each PU. Note that by applying this mechanism only to the pixel domain, no mapping issues between MB boundaries (H.264/AVC) and coding unit boundaries (HEVC) need to be solved.

As far as the authors of this paper know, at the moment the only approach which tries to deal with the problem of simultaneous encoding with H.264/AVC and the new HEVC standard in a hybrid multiview scenario is the one presented by these same authors in [8], in which a preliminary version of this algorithm is described. In that paper, NB classifiers (see e.g. [12]) are already used since both the training and the classification stages are very efficient and, moreover, it achieves good results, obtaining a 64%

acceleration for HEVC side views with only a 3.8% bit rate increment for 2 views while preserving the same objective quality.

In [8], the CTU partitioning of the side HEVC views of the multiview hybrid architecture is already accelerated, but that work has been improved upon in the following ways:

1. In [8], only information from the H.264/AVC decoder was used, while in this version, information which is only available in the encoder has been included. It can be noted that in this hybrid multiview scenario, the H.264/AVC encoder is present.
2. An adaptive energy-based model of classifiers which fits the characteristics of different hierarchical layers of B frames has been included.
3. The algorithm for level 2 (i.e. CUs of size 16×16 pixels) classification, which is not only based on the current H.264/AVC MB decision but also on the decisions of adjacent MBs, has been improved.

3 Proposed Algorithm

This paper proposes a software algorithm which aims to reduce HEVC's computational complexity in deciding the most appropriate depth for each quadtree in the hybrid architecture described. The algorithm presented is an improved version of the algorithm published in [8] for H.264/AVC and HEVC hybrid multiview video. This new version of the algorithm has been called *Adaptive Fast Quadtree Level Decision* (AFQLD), where the term adaptive refers to the fact that the improved version adapts to the particularities of the video sequence which is being encoded in each case.

Even though the algorithm presented in this paper is a non-parallel algorithm, it can easily be combined with parallel algorithms aimed at parallelizing HEVC to speed up the encoding process. For instance, [7] proposes a fast software transcoding algorithm which is combined with parallelization algorithms at CPU and GPU levels. Moreover, as the algorithm consists of a fast quadtree decision and no changes to the syntax have been made, the decoder need not be changed in any way.

The algorithm has an incremental design, so that for each level in the quadtree the algorithm decides whether it is more likely to split the CU (C_S), and descend a level in the quadtree, or not to split the CU (C_N), and choose the current level as the maximum allowed depth. Therefore, C_S and C_N are the two *class labels* to be predicted by the decision function or *classifier*.

If C_S is chosen, only Skip and 2Nx2N PUs are checked for levels 0 and 1 (since the decision might be wrong and these calculations let the quadtree go back to upper levels if RD costs are worse at deeper levels), while all PUs are checked at level 2. On the other hand, if the decision is C_N, then the current depth is considered as final, all the PUs at this CU depth are evaluated and the algorithm for this CTU terminates. This process is described in Algorithm 1.

Algorithm 1 AFQLD algorithm

```
if level==3 then
  Calculate all PUs
  return best CU/PU in RD terms
else
  Classify this CU as C_S or C_N by using the corresponding M_i model.
  if C_S then
    if level==2 then
      Calculate all PUs
    else
      Calculate Skip and 2Nx2N PUs
    end if
    Split CU into 4 sub-CU
    Apply this algorithm for each sub-CUs
  else
    Calculate all PUs
    return best CU/PU in RD terms
  end if
end if
```

The different models, $\mathcal{M}_l$, which make the decision at each level $l = 0, 1, 2$ need to be built. Specifically, we rely on a *data mining* approach for levels 0 and 1 of the quadtree (CU sizes of 64x64 and 32x32 pixels, respectively), while at level 2 a much simpler strategy is followed. Basically, as the CU size at level 2 is 16x16 pixels, we take advantage of the fact that this is the MB size in H.264/AVC too, so the proposed algorithm mimics H.264/AVC as described in Algorithm 2 by taking into consideration the adjacent blocks in some cases too.

Therefore, at levels 0 and 1 of the quadtree, the task under study is a supervised classification problem, where our aim is to predict the correct value of a binary class variable. Specifically, a probabilistic Naïve-Bayes classifier has been selected, and this computes the posterior probability of each label C_i given the set of features $\mathbf{F} = \{w_1, \ldots, w_n\}$ as input: $P(C_i|\mathbf{F}) \propto P(C_i) \prod_{j=1}^{n} P(w_j|C_j)$, and it chooses the output label with the highest probability.

Therefore, an initial training stage to learn the models which will be used in the algorithm is needed. Several models must be learnt offline, depending on

Algorithm 2 Splitting algorithm for quadtree level 2.

```
if MB mode is Skip or 16x16 then
  Classify as C_N
else if MB mode is 16x8 or 8x16 then
  if Adjacent MB modes are Skip, 16x16, 16x8 or 8x16 then
    Classify as C_N
  else
    Classify as C_S
  end if
else
  Classify as C_S
end if
```

the CU depth (0 or 1) and the average energy of the residue, where 4 levels of energy have been considered (1, 2, 3 and 4), where 1 represents high residual energy and 4 represents low residual energy. Thus, each frame in the *Random Access* (RA) configuration can be identified with a different energy according to its hierarchical layer in the *Group of Pictures* (GOP) structure.

The classifiers have been trained using 4 QP values (22, 27, 32, 37), where a higher QP implies greater compression but with a higher quality loss, and 4 sequences from those described in [3] (*PeopleOnStreet, ParkScene, PartyScene* and *BQSquare*), with one sequence per class (A, B, C and D), so that they can also be representative of the wide range of resolutions. The first 1000 CUs of each QP-sequence pair using the RA configuration were selected as a training set. The initial set of features, **F**, is fetched from the H.264/AVC base view encoder, and these are calculated for the area covered (in MBs) by the current CU in the HEVC views.

According to problem domain knowledge, the following families of features can be good predictors to help in the decision making: 1) features which correctly model the spatial and temporal complexity; 2) according to the framework of this work, information fetched from the encoding stage of the H.264/AVC view is available; 3) statistical data, such as the variance of the residue [11], have been shown to work well in previous transcoders; 4) information which could summarize both the spatial and the temporal information simultaneously; and 5) dynamical information fetched from the HEVC views can also be extracted.

According to the above information, the initial set of features, **F**, contains a total of 53 continuous variables. The first 24 can be fetched from the H.264/AVC decoder, while the next 27 features can only be extracted from the H.264/AVC encoder (which is present in this hybrid multiview scenario). Finally, the last 2 features are dynamically calculated during the encoding of the HEVC views, where:

- w_{QP}: QP value used to encode the stream.
- w_{bits}: number of bits used to encode all the MBs for the current CU after applying the context-adaptive binary arithmetic coding (CABAC) operation.
- w_{intra}, w_{skip}, w_{16}, w_4, w_{inter}: number of Intra, Skip, Inter 16x16, Inter 4x4 and other Inter MBs, respectively.
- w_{DCTno0}: number of non-zero DCT coefficients.
- w_{width}, w_{height}: frame width and height, respectively.
- w_{MVsum}: sum of all the MV components contained in the frame.
- w_{resAvg}, w_{resVar}: average and variance of the residue for the area covered, respectively.
- $w_{resAvgSubCU1}$, $w_{resAvgSubCU2}$, $w_{resAvgSubCU3}$, $w_{resAvgSubCU4}$: average of the residue for each sub-CU: 1, 2, 3 and 4, respectively.
- $w_{resVarSubCUs}$: variance of the 4 previous values.
- w_{sobelH}, w_{sobelV}: sum of applying the Sobel operator [20] to the residue in horizontal and vertical directions, respectively.
- $w_{RDCostMode[i]}$: the RD cost of the i mode, where i is Skip, 16x16, 16x8, 8x16, 8x8 (the best RD cost of all possible 8x8 and smaller partitions), Intra 16x16, Intra 8x8, Intra 4x4, and Intra PCM.
- $w_{AvgMVx[i]}$, $w_{AvgMVy[i]}$: average of all x and y MV components, respectively, of each i mode, where i is 16x16, 16x8, 8x16 and 8x8.
- $w_{VarMVx[i]}$, $w_{VarMVy[i]}$: variance of all x and y MV components, respectively, of each i mode, where i is 16x16, 16x8, 8x16 and 8x8.
- $w_{VarIntraDir8x8}$, $w_{VarIntraDir4x4}$: variance of all Intra directions of Intra 8x8 and Intra 4x4 modes.
- w_{MVxAvg}, w_{MVyAvg}, w_{MVxVar}, w_{MVyVar}: average and variance of x and y MVs components, respectively, for the area covered.
- $w_{SkipCost}$, $w_{2Nx2NCost}$: the HEVC Lagrangian cost of choosing Skip and 2Nx2N, respectively.

3.1 Data preprocessing

After the above features have been fetched and calculated, a step prior to the start of the training process is to obtain more accurate datasets than the original ones. Initially, the 53 features are of a numerical nature but, to avoid the improbable assumption that the values of each feature given the class follow a parametric distribution (e.g. a normal distribution), they are discretized using the

entropy-based Fayyad-Irani algorithm [10], that is, the intervals are chosen in such a way that the resulting variable has as much discriminative power regarding the class as possible.

Then, a *Feature Subset Selection* (FSS) is applied to select the proper subset of features [13]. We chose a greedy strategy with wrapper evaluation. Thus, the process starts with an empty set and iteratively incorporates the best remaining feature at each step. In the wrapper approach, the best feature is the one that, when joined to the current subset, induces the classifier with the maximum accuracy.

The NB algorithm is used during the FSS search to evaluate the goodness of each subset, removing those redundant and irrelevant variables that may reduce its accuracy. The FSS process finishes when the addition of features no longer improves the accuracy of the classifier.

After these steps, the 8 classifiers, for 2 depth levels (i.e. 64x64 and 32x32) and 4 levels of energy, are learnt using an NB training process, which finishes the offline stage of the algorithm.

3.2 Online stage of the algorithm

Once all the 8 base classifiers have been learnt, an online stage is carried out for each HEVC view at encoding time. This stage is made up of two steps: the learning of a classifying threshold and the displacement of some MBs from their original location according to the difference between views.

On the one hand, regarding threshold learning, it should be noted that the basic classification rule in our NB classifiers (which only have two classes) is to choose $P(C_S)$ if $P(C_S) > P(C_N)$. However, intuitively, the cost of making the error of not splitting when the standard HEVC decides to split should be more costly because, if we decide to split, the speed drops but the quality of the image is preserved.

In order to take this fact into consideration, the classification rule can be modified by adding misclassifying costs, i.e. choose C_S if $P(C_S) \times Cost_{SN} > P(C_N) \times \times Cost_{NS}$, where $Cost_{SN}$ is the cost of choosing C_S when the correct decision would have been C_N and $Cost_{SN}$ is the cost of the opposite error.

To measure these costs, the Lagrangian costs of splitting (L_S) and of not splitting (L_N) have been used, as well as the concept of absolute error, as shown in Equation (1), where $i, j \in \{S, N\}$ (i being the predicted decision and j the correct one) and ωij is a weight associated to each particular cost.

$$Cost_{ij} = |L_j - L_i| \times \omega_{ij}, \quad (1)$$

In similar approaches in a transcoding scenario [9], the weighting values were $\omega_{NS} = 2.0$ and $\omega_{SN} = 1.0$ (since the cost of not splitting is higher due to the fact that no more CUs will be checked if the decision is not to split). However, it should be taken into account that in this scenario the Lagrangian costs are lower than in a transcoding scenario, since the sequence has not been encoded and decoded previously and, therefore, the differences between the original and the encoded sequences are less. Moreover, the absolute error, which is a scale-sensitive metric, is used to calculate the costs. Thus, these two facts jointly mean that the costs calculated in the hybrid multiview scenario are lower than in the transcoding scenario, which might cause misclassification. In order to solve this problem, the ω_{ij} values have been changed. After heuristically trying several weights, it has been concluded that the best weights are $\omega_{NS} = 2.0$ and $\omega_{SN} = 1.0$.

On the other hand, it should be taken into account that in a Multiview Video different views have a displacement between them, and this displacement is not constant: objects which are closer to the camera have a greater displacement than those which are in the background. As the training process does not take this fact into account, it should be compensated for during the encoding process since, otherwise, the mapping between H.264/AVC and HEVC would not overlay the same regions of the picture. To solve this problem, when an MB from the H.264/AVC base view is going to be assigned to a CU in one of the HEVC views for the mapping, the original mapping may be displaced by up to 1 CU at level 0, and by up to 2 CUs at level 1.

In order to obtain an approximation of the displacement, the *Sum of Absolute Differences* (SAD) is calculated between the current and the base views with different displacements in pixels, $d_p \in \{0, 1, ..., 63\}$, and the best SAD value (SAD_{best}) is chosen. The maximum displacement has been set to 63 pixels since the displacement between views (even for those objects which are in the foreground) is not expected to be bigger; in fact, after some preliminary tests, about 90% of the MBs have a displacement smaller than 25 pixels.

Then, if $SAD_{best} \in [0, 7]$, the MB is considered not to have any displacement and the displacement in MBs, d_{MB}, is 0. If $SAD_{best} \in [8, 23]$, then $d_{MB} = 1$ (the direction of the displacement is determined by

Table 1 Results of the proposed AFQLD algorithm for 2 and 3 views.

		2 views		3 views	
Resolution	Sequence	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
1024x768	Balloons	4.9	71.43	6.1	71.48
	Kendo	11.0	74.62	13.5	74.58
	Newspaper_CC	3.7	72.35	4.4	72.48
	Average	*6.5*	*72.80*	*8.0*	*72.85*
1920x1088	GT_Gly	4.7	64.09	5.5	64.06
	Poznan_Hall2	4.8	74.00	5.8	74.10
	Poznan_Street	1.8	72.24	2.2	72.39
	Undo_Dancer	3.7	67.86	4.5	67.83
	Shark	4.2	64.36	5.2	64.30
	Average	*3.8*	*68.51*	*4.6*	*68.54*
Global average		*4.8*	*70.12*	*5.9*	*70.15*

the position of the current view relative to the base view). If $SAD_{best} \in [24, 39]$, then $d_{MB} = 2$ MBs. If $SAD_{best} \in [40, 55]$, then $d_{MB} = 3$ MBs, and if $SAD_{best} \in [56, 63]$, then $d_{MB} = 4$ MBs. Finally, given d_{MB} and the level, the displacement in CUs for the given level d_{CU_i} is easily calculated.

One last thing should be considered when using this displacement technique: the original features were calculated with 16 MBs mapping onto a 64x64 CU and 4 MBs mapping onto a 32x32 CU. Since taking displacement into account might result in a different number of MBs being mapped onto a CU, a weighting of the features is performed: the features calculated online are divided by the actual number of MBs and multiplied by the original number of MBs (e.g. 16 at level 0 and 4 at level 1). Finally, if a CU does not have any mapped MB, the algorithm is not applied and the full encoding is performed (i.e. the borders of the frames).

4 Performance Evaluation

The multiview encoder has been tested with the sequences and test conditions approved in [18]. The QP values used were $\{22, 27, 32, 37\}$, and the configuration was Random Access Main (RA) with the hybrid multiview flag enabled and using 3 views, namely the H.264/AVC base view and two HEVC views. The results are shown for each sequence. Note that the testing sequences are those defined by [18], while the training sequences were chosen from those defined by [3], so the training and testing sets are disjoint sets. According to [18], two different multiview resolutions should be checked, so the average of each resolution and the global average have been calculated.

The JM 18.4 [17] software was used for the H.264/AVC view, whereas SHM 6.1 [5] was used for HEVC views. It should be noted that the HTM software (which is the reference software for general HEVC multiview video coding) cannot be used, since it does not implement the hybrid H.264/AVC and HEVC coding, whereas SHM implements it and also allows the encoding of multiview video with this hybrid option. The remainder of the coding parameters are kept as default in the configuration file. The process to generate these results is the following:

1. Encode the YUV file of the base view with the JM software using *HM-like* configuration files.
2. Decode that file, producing the decoded one so that SHM can carry out the inter-view prediction, as well as all the information needed for the proposed algorithm.
3. Encode the YUV files of the side views with the original SHM software (*reference*).
4. Encode the YUV files of the side views with the SHM software using the proposed algorithm (*proposed*).
5. Compare the *reference* and the *proposed* streams in order to obtain the BD-rate [2] and the Time Reduction (TR).

Measurements have been performed on a six-core Intel Core i7-3930K CPU running at 3.20GHz (parallelization techniques have not been used though, so the proposal has been run using only one core at a time). The results are presented in terms of TR and *BD-rate* (which measures the increment in bit rate while maintaining the same objective quality), which are calculated as indicated in (2), where $t_{reference}$ is the encoding time of the HEVC views using the non-accelerated *reference* encoder, and $t_{proposed}$ is the encoding time of the HEVC views using the *proposed* fast encoder. The global BD-rate is the weighted average of the Y, U and V components (given that the luminance is four times larger than the chrominances).

$$\begin{aligned} \text{TR } (\%) &= \frac{t_{anchor} - t_{proposed}}{(t_{anchor})} \times 100 \\ \text{BD-Rate}(\%) &= \frac{4 \times \text{BD-rate}_\text{Y} + \text{BD-rate}_\text{U} + \text{BD-rate}_\text{V}}{6} \end{aligned} \quad (2)$$

Table 1 contains the results for the above configuration in terms of the TR of the global encoding time and the BD-rate for 2 and 3 views. As the final stream is composed of several views, the BD-rate has been calculated using the sum of the bit rates of all the views, while the PSNR is calculated as the average value of the PSNR of the views which are displayed at the same timestamp. The TR is calculated using the acceleration of the HEVC views, since the base view can be encoded with an already optimized H.264/AVC encoder, such as *x264* [1], and the H.264/AVC encoding time is negligible compared to the HEVC encoding time. While the displacement calculation process described above might seem to consume a lot of time, it can be noticed in these results that this time is negligible, since the TR results include the calculation of these displacements.

On the one hand, it can be seen that for both the 2-view and the 3-view cases (one H.264/AVC view and two or three HEVC views, respectively) the average TR is similar, and on average is about 70% (which represents a speed-up of 3.35x). In the best case, it reaches almost 75% (speed-up of 4.00x).

On the other hand, the average BD-rate is about 4.8% in the 2-view case and about 5.9% in the 3-view case, which are low increments in bit rate (in the best case, the BD-rate is as low as 1.8% and 2.2% for 2 and 3 views). The difference in the BD-rate when adding a third view is due to the fact that the bit rates of all the views are summed and 2 of the 3 views have a slightly increased bit rate.

4.1 Comparison with other proposals

As mentioned in Section 2, due to the novelty of the proposal presented in this paper, it cannot be fairly compared with any other proposal except the one presented in [8], which is a preliminary work on the algorithm presented here. The algorithm in [8] is called FQLD, while the current algorithm is called AFQLD, referring to the capacity of the algorithm to be *adaptive* to the content of the sequence, by using the dynamic threshold calculation, and to the GOP structure based on the energy of the frames according to the hierarchical layer they belong to instead of a specific given GOP.

Table 2 shows the results of the FQLD algorithm for multiview hybrid video coding [8] using the same configuration and setting which have been used in this paper so that the results can be fairly compared. As can be seen, the average results show that the TR has increased from 64% to 70%, showing the improvements in the algorithm. However, the BD-rate has also increased by 1% in the 2-view case and by 1.2% in the 3-view case, which, nevertheless, is not too high an increment. Furthermore, a TR of 70% means that HEVC views will achieve similar encoding times to the base H.264/AVC views, while taking advantage of the bit rate reduction of HEVC.

Finally, as mentioned above, given the novelty of the proposal and of the scenario, it cannot be fairly compared with other proposals since the authors have not been able to find any other similar works on hybrid coding which try to accelerate the HEVC views of a multiview video stream using the information of an H.264/AVC base view. However, it can be compared with an HEVC fast encoding algorithm, specifically the *Early CU termination* (ECU) algorithm [6], which is included in the HM software, and obtains a 2.3% BD-rate (for 1 view) with a 37% encoding time reduction. Comparing these values with the 2-view case (which is the fairest comparison since only one of the views is accelerated), it can be seen that the time reduction of the proposed algorithm is almost twice as large while the BD-rate increment is much lower than twice the figure.

5 Conclusions and Future Work

This paper proposes an algorithm which uses the information from the H.264/AVC base view in a multiview video stream and aims to accelerate the HEVC views of the stream by deciding which quadtree level is the most appropriate without the need for testing all the possible CUs/PUs. A dynamical approach is followed, since during the encoding process a view displacement compensation is performed and sequence-dependent classifying thresholds are learnt.

It has been demonstrated that a good tradeoff between quality loss and acceleration is achieved: a TR of 70% on average, with a slight increment of 4.8% in the BD-rate in the 2-view case, and 5.9% in the 3-view case.

As future work, the model could be improved by using perceptual video coding concepts, where not only the objective quality is taken into account, but also the subjective perception of the viewer, as well as the main saliency areas of the frames.

Acknowledgements This work has been jointly supported by the MINECO and European Commission (FEDER funds) under the projects TIN2012-38341-C04-04 and TIN2015-66972-C5-2-R. Likewise, this work has also been supported by the Spanish Education, Culture and Sports Ministry under grant FPU12/00994.

Table 2 Results of FQLD algorithm [8].

		2 views		3 views	
Resolution	Sequence	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
1024x768	Balloons	4.1	64.61	5.1	65.05
	Kendo	4.8	62.66	6.0	62.91
	Newspaper_CC	3.3	68.85	4.2	69.21
	Average	*4.1*	*65.52*	*5.1*	*65.99*
1920x1088	GT_Gly	5.0	57.27	6.0	57.31
	Poznan_Hall2	4.4	68.92	5.2	68.86
	Poznan_Street	1.6	69.94	1.9	70.03
	Undo_Dancer	2.6	58.57	3.3	58.64
	Shark	4.4	56.51	5.4	56.55
	Average	*3.6*	*63.24*	*4.4*	*63.24*
Global average		*3.8*	*64.16*	*4.7*	*64.29*

References

1. Aimar, L., Merritt, L., Glaser, F., Mitrofanov, A., Gramner, H.: x264 software (2013). URL http://www.videolan.org/developers/x264.html
2. Bjontegaard, G.: Improvements of the BD-PSNR model. ITU-T SG16 Q **6**, 35 (2008)
3. Bossen, F.: Common HM Test Conditions and Software Reference Configurations. In: Proc. 12th JCT-VC Meeting, Doc. JCTVC-L1100 (2013)
4. Chen, J., Boyce, J., Yan, Y., Hannuksela, M., Sullivan, G., Wang, Y.: Scalable High Efficiency Video Coding Draft 7. In: JCTVC-R1008, 18th JCTVC Meeting, Sapporo, Japan (2014)
5. Chen, J., Boyce, J., Ye, Y., Hannuksela, M.M.: Scalable HEVC (SHVC) Test Model 6 (SHM 6). In: Proc. 17th JCT-VC Meeting, Valencia, Spain, No. JCTVC-Q1007 (2014)
6. Choi, K., Park, S.H., Jang, E.S.: CodingTtree Pruning base CU Early Termination. In: JCT-VC document, JCTVC-F902 (2011)
7. Diaz-Honrubia, A.J., Cebrian-Marquez, G., Martinez, J.L., Cuenca, P., Puerta, J.M., Gamez, J.A.: Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding. Journal of Real-Time Image Processing pp. 1–17 (2014). DOI 10.1007/s11554-014-0477-z
8. Diaz-Honrubia, A.J., De Praeter, J., Van Leuven, S., De Cock, J., Martinez, J.L., Cuenca, P.: Using bayesian classifiers for low complexity multiview h.264/avc and hevc hybrid architecture. In: IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1–6 (2015). DOI 10.1109/MLSP.2015.7324366
9. Diaz-Honrubia, A.J., Martinez, J.L., Cuenca, P., Gamez, J.A., Puerta, J.M.: Adaptive fast quadtree level decision algorithm for h.264/hevc video transcoding. Circuits and Systems for Video Technology, IEEE Transactions on **26**(1), 154–168 (2016). DOI 10.1109/TCSVT.2015.2473299
10. Fayyad, U.M., Irani, K.B.: Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In: Proceedings of the International Joint Conference on Uncertainty in AI, pp. 1022–1027 (1993)
11. Fernandez-Escribano, G., Kalva, H., Cuenca, P., Orozco-Barbosa, L., Garrido, A.: A Fast MB Mode Decision Algorithm for MPEG-2 to H.264 P-Frame Transcoding. IEEE Transactions on Circuits and Systems for Video Technology **18**(2), 172–185 (2008)
12. Flores, J., Gámez, J.A., Martínez, A.M.: Supervised Classification with Bayesian Networks. In: Intelligent Data Analysis for Real-Life Applications: Theory and Practice, pp. 72–102 (2012)
13. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research **3**, 1157–1182 (2003)
14. H.264, I...R.I.T.: Advanced Video Coding for Generic Audiovisual Services. Annex H: Multiview Video Coding (MVC) (2009)
15. ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC) version 22: Advanced Video Coding for Generic Audiovisual Services (2012)
16. ITU-T Recommendation H.265 and ISO/IEC 23008-3 (Version 3): High Efficiency Video Coding (2015)
17. JCT-VC: Reference Software to Committee Draft, version 18.4 (2012)
18. Muller, K., Vetro, A.: Common Test Conditions of 3DV Core Experiments. In: Proc. 7th JCT3V Meeting, Doc. JCT3V-G1100 (2014)
19. Ohm, J., Sullivan, G., Schwarz, H., Tan, T.K., Wiegand, T.: Comparison of the Coding Efficiency of Video Coding Standards - Including High Efficiency Video Coding (HEVC). IEEE Transactions on Circuits and Systems for Video Technology **22**(12), 1669–1684 (2012)
20. Patnaik, S., Yang, Y.M.: Soft Computing Techniques in Vision Science, vol. 395. Springer (2012)
21. Sullivan G. J. Ohm, J.R., Han, W.J., Wiegand, T.: Overview of the High Efficiency Video Coding (HEVC) Standard. IEEE Transactions on Circuits and Systems for Video Technology **22**(12), 1649–1668 (2012)
22. Tech, G., Wegner, K., Chen, Y., Hannuksela, M., Boyce, J.: MV-HEVC Draft Text 9. In: JCT3V-I1002, 9th JCT3V Meeting, Sapporo, Japan (2014)
23. Van Leuven, S., Bruls, F., Van Wallendael, G., De Cock, J., Van de Walle, R.: Doc.MPEG-M23669: Hybrid 3D Video Coding. ISO/IEC JTC1/SC29/WG11 (MPEG) (2012)
24. Van Leuven, S., Van Wallendael, G., De Cock, J., Bruls, F., Luthra, A., Van de Walle, R.: Doc.MPEG-M24968: Overview of the coding performance of 3D video architectures. ISO/IEC JTC1/SC29/WG11 (MPEG) (2012)