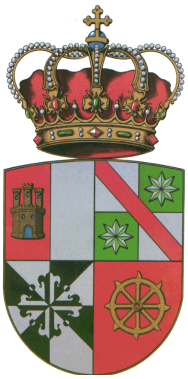
UNIVERSITY OF CASTILLA-LA MANCHA

Computing Systems Department



H.264/AVC and HEVC Collaborative Video Coding

A dissertation for the degree of
Doctor of Philosophy in Computer Science
to be presented with due permission of
the Computing Systems Department,
for public examination and debate.

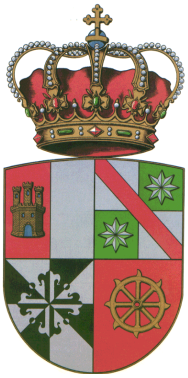
Author: Antonio Jesús Díaz Honrubia

Advisors: Dr. Pedro Ángel Cuenca Castillo
Dr. José Luis Martínez Martínez

Albacete, November 2016

UNIVERSIDAD DE CASTILLA-LA MANCHA

Departamento de Sistemas Informáticos



H.264/AVC and HEVC Collaborative Video Coding

Tesis doctoral presentada al
Departamento de Sistemas Informáticos
de la Universidad de Castilla-La Mancha
para la obtención del título de Doctor
en Tecnologías Informáticas Avanzadas.

Autor: Antonio Jesús Díaz Honrubia

Directores: Dr. Pedro Ángel Cuenca Castillo
Dr. José Luis Martínez Martínez

Albacete, noviembre de 2016

Agradecimientos

Hace ya unos cuantos años que decidí que quería estudiar un Doctorado y parecía que quedaba mucho tiempo hasta llegar a la meta. En lo cierto o no (todo depende lo que se considere "mucho tiempo"), parece que ya se ve el fin de este camino. Decía Antonio Machado que "al andar se hace camino, y al volver la vista atrás, se ve la senda que nunca se ha de volver a pisar". Sin embargo, al volver la vista atrás también vemos la senda que sí volveríamos a pisar y sin duda, este camino, como muchos otros, lo volvería a pisar.

Al volver la vista atrás también se ve a todas aquellas personas que te han ayudado a andar el camino y que no se puede dejar de estarles agradecido. Desde luego, son muchas, tantas que no cabrían todas en estas líneas, espero que sepan perdonarme todos aquellos que no nombro expresamente por no hacer interminable esta sección. Sin lugar a dudas, gracias en primer lugar a mi familia, a mis padres, Ramón y Antonia, a mi hermano, Daniel, a mis abuelos, tíos, primos, etc. Gracias por todo lo que se han esforzado por mí y por lo que me han enseñado que no se aprende en colegios ni universidades.

No puedo olvidarme de mis directores, Pedro y José Luis, a los que también considero mis amigos. Gracias porque, a pesar de que a veces he podido estar algo distraído, siempre habéis confiado en mí. Gracias por todos los consejos y también por los momentos que me llevo y por los que quedan por venir. En este punto también debo agradecer al resto de profesores de la ESII, que tan magníficamente forman a tantos estudiantes. Especialmente, por su estrecha colaboración en esta Tesis, debo agradecer sus aportaciones a José Miguel Puerta y José Antonio Gámez.

Muchas gracias también a mis compañeros del RAAP, que hacen mucho más amenas las horas de trabajo y siempre están dispuestos a ayudar cuando surge algún problema. Son muchos los que han pasado en estos años y algunos han emprendido caminos distribuídos por universidades y empresas de toda España y del extranjero, otros siguen su camino hacia la meta del Doctorado. A todos, gracias y ánimo en vuestros caminos.

I must not forget to mention all the people who I met during my research stays in Ghent and Boca Raton. I had (and I still have) the chance to learn a lot from them. The researchers at Ghent University in particular are responsible of some of the ideas included in this Thesis. I enjoyed my time there enormously, in such a great and historic city and country. The researchers at Florida Atlantic University also gave me some good ideas for the future, and I had a good time there too, enjoying the good weather and the American way of life. My best wishes for you all.

Parte indispensable para poder sobrellevar el día a día y no morir en el intento son mis amigos. Muchas gracias por el ánimo necesario y por vuestros consejos. Disculpadme que no nombre a ninguno en concreto, ya que serían muchos los frentes que tendría que abordar para no dejar a nadie. Como ya he dicho en alguna ocasión, cualquier persona que pasa por nuestra vida, aunque sea efímeramente, nos marca de una forma u otra y no somos nosotros mismos sin todo lo que hemos vivido.

Por último, pero no por ello menos importante, gracias a Dios, Él es el que nos ayuda a seguir nuestro camino, el Viento que nos empuja aún cuando ni nosotros mismos podemos mover los pies.

En fin, este camino se acaba, pero estoy seguro de que empieza otro que será igual de emocionante, la cosa es seguir andando y dónde quiera que vaya este camino, seguro que también merecerá la pena pisarlo.

La investigación desarrollada ha sido financiada por el Ministerio de Educación, Cultura y Deporte a través de la beca FPU con referencia FPU 12/00994; y por el Ministerio de Economía y Competitividad y Fondos FEDER a través de los proyectos TIN2009-14475-C04-03, TIN2012-38341-C04-04 y TIN2015-66972-C5-2-R.

Resumen

En los últimos años la cantidad de tráfico de vídeo en Internet se ha visto incrementado sustancialmente, llegando a ocupar hasta el 70% del tráfico total de Internet en 2015. Además, se espera que este porcentaje siga creciendo en los próximos años con la llegada de contenidos de Ultra Alta Definición, vídeo 3D y multivista, tasas de fotogramas muy altas, mayores profundidades de bits para las muestras, patrones de submuestreo con más muestras para las crominancias, etc. Por esta razón, cada vez se demandan nuevos códecs de mejores prestaciones que sean capaces de comprimir más la señal de vídeo, sin que ello suponga una degradación de la calidad.

Con este objetivo, el *Joint Collaborative Team on Video Coding* lanzó un nuevo estándar de codificación de vídeo en 2013 llamado *High Efficiency Video Coding* (HEVC). Además, en octubre de 2014 se lanzaron las extensiones escalable y multivista y los perfiles de extensión de rango. Por otra parte, en abril de 2015 se amplió el estándar con la extensión para vídeo 3D. Este nuevo códec es capaz de reducir el ancho de banda de su predecesor, esto es, H.264/MPEG-4 Parte 10 – *Advanced Video Coding* (AVC), un 50% manteniendo la misma calidad objetiva. Sin embargo, esto se consigue introduciendo herramientas que requieren un coste computacional muy alto, por lo que HEVC requiere más del doble de tiempo que H.264/AVC para codificar la misma secuencia.

Por ello, muchos de los contenidos que hoy en día están codificados con H.264/AVC, serán próximamente convertidos a HEVC para obtener un ahorro en el ancho de banda cuando sean enviados por Internet u otras redes. Sin embargo, al mismo tiempo, debería conservarse una versión de los vídeos codificada con H.264/AVC para aquellos dispositivos más antiguos que no sean capaces de decodificar HEVC. Así pues, se definen dos escenarios que nos conducen a la temática de esta Tesis: el primero de ellos es la transcodificación a HEVC de secuencias ya existentes y codificadas con H.264/AVC. El segundo consiste en codificar las secuencias de nueva producción en ambos estándares, para que pueda ser decodificada tanto por dispositivos antiguos, como por aquellos que sean compatibles con HEVC, de forma que éstos puedan aprovechar su superior capacidad de compresión.

Sin embargo, el mayor problema de estos dos procesos es la gran cantidad de tiempo que HEVC requiere para codificar una secuencia. No obstante, como en ambos casos dicha secuencia estaba también codificada en H.264/AVC o se está codificando en ese momento

en ese estándar, parece lógico que exista información que pueda extraerse del flujo de H.264/AVC y que pueda ser reutilizada para acelerar la toma de decisiones en HEVC, reduciendo así su coste computacional.

Con todo esto en mente, esta Tesis propone cuatro escenarios diferentes en los que ambos estándares están presentes y, en los cuáles, el codificador de HEVC es acelerado mediante la reutilización de información extraída del flujo de H.264/AVC. El primer escenario es el bien conocido problema de transcodificación heterogénea de vídeo. En segundo lugar, se propone el problema de la codificación simultánea de ambos estándares. Entonces, para ello se propone también otra opción para proveer una versión decodificable en ambos estándares mediante la versión híbrida escalable de HEVC, que utiliza H.264/AVC para codificar la capa base y HEVC para las capas de mejora. Finalmente, por analogía con el caso anterior, se propone el problema de la codificación híbrida multivista de HEVC, en la que la vista base va codificada con H.264/AVC y el resto de vistas con HEVC.

Los resultados para la configuración *Random Access* (acceso aleatorio) muestran que el algoritmo rápido de transcodificación propuesto obtiene una reducción del tiempo de codificación de HEVC del 54% en media, con una penalización en términos de BD-rate (que mide el incremento de la tasa de bits necesario para mantener la misma calidad objetiva) de sólo un 2.7%. En el caso de la codificación simultánea de H.264/AVC y HEVC, se consigue aproximadamente la misma aceleración, pero con un BD-rate de sólo 2.4%. En lo que respecta al codificador híbrido escalable de HEVC, se obtiene una aceleración del 60% con un BD-rate del 2.6%. Finalmente, el codificador híbrido multivista de HEVC obtiene una aceleración del 70%, pero en este caso, debido a las diferencias de iluminación y ruido entre vistas, el BD-rate se incrementa hasta el 4.8% en el caso de 2 vistas y hasta el 5.9% en el caso de 3 vistas.

Summary

In the last few years the amount of video traffic on the Internet has increased substantially, amounting to as much as 70% of all Internet traffic during 2015. Moreover, this percentage is expected to increase further in the next few years with the advent of Ultra High Definition contents, multiview and 3D video, very high frame rates, higher bit depths for samples, subsampling patterns with more samples for chrominances, etc. Because of this, new codecs which can compress the video signal to a greater extent without a drop in the quality are required.

With this objective, the *Joint Collaborative Team on Video Coding* released the *High Efficiency Video Coding* (HEVC) standard in 2013. In addition to this, in October 2014 the scalability and multiview extensions, as well as the range extension profiles, were released. And in April 2015 the 3D extension for HEVC was included in the standard too. This new codec is able to reduce the bit rate of its predecessor, namely H.264/MPEG-4 Part 10 – *Advanced Video Coding* (AVC), by 50% while maintaining the same objective quality. However, this is achieved by the introduction of very computationally complex tools that make HEVC take twice time that H.264/AVC requires to encode the same sequence.

Therefore, many contents that are currently encoded with H.264/AVC will be converted to HEVC in order to save bandwidth when they are sent via the Internet or a network. Nevertheless, at the same time, a version of the videos encoded with H.264/AVC should be preserved for legacy devices which cannot decode HEVC yet. Thus, there are two main scenarios that lead us to the aims of this Thesis: the first one is to transcode the already existing H.264/AVC sequences to HEVC; the second one is to encode newly-produced sequences using both standards, so that they can be decoded by legacy devices, while HEVC-capable devices can take advantage of the superior compression performance of the new standard.

The main challenge for these two processes is the large amount of time that HEVC requires to encode a sequence. Nevertheless, as in both cases this stream is or was also encoded with H.264/AVC, it seems logical that some information that can be fetched from the H.264/AVC stream might be re-used in order to accelerate the decisions on the HEVC encoder, thus reducing its computational complexity.

With all the above facts in mind, this Thesis proposes four different scenarios in which both standards are present, and in which the HEVC encoder is accelerated by the re-use of information fetched from the H.264/AVC stream. The first scenario is the well-known problem of heterogeneous transcoding. Secondly, the problem of simultaneous encoding both standards is shown. Then, another solution that is designed to provide a decodable version in both standards by means of a hybrid scalable HEVC is presented. In this solution the base layer is encoded in H.264/AVC, while the enhancement layers are encoded in HEVC. Finally, by analogy with the previous case, the hybrid multiview HEVC problem is described; here the base view is encoded with H.264/AVC and the other views are encoded by the use of HEVC.

The results for the *Random Access* configuration show that the proposed fast trancoding algorithm obtains a 54% reduction in HEVC encoding time on average, with a penalty in BD-rate terms (which measures the increment in the bit rate to obtain the same objective quality) of only 2.7%. In the case of the simultaneous H.264/AVC and HEVC encoder, the same acceleration is achieved, but with a BD-rate of only 2.4%. Regarding the hybrid scalable HEVC, an acceleration of 60% is reached with a BD-rate of 2.6%. Finally, the hybrid multiview HEVC obtains an acceleration of the HEVC encoder of 70%, but in this case, due to the differences in lighting or noise between views, the BD-rate increases to 4.8% in the 2-view case and 5.9% in the 3-view case.

Acronyms

AFQLD Adaptive Fast Quadtree Level Decision

AVC Advanced Video Coding

BDR BD-Rate

CBF Coded Block Flag

CFM CBF Fast Mode

CTU Coding Tree Unit

CU Coding Unit

DVC Distributed Video Coding

ECU Early CU Termination

ESD Early Skip Detection

fps Frames per second

FQLD Fast Quadtree Level Decision

FSS Feature Subset Selection

GOP Group Of Pictures

GPU Graphics Processor Unit

HD High Definition

HEVC High Efficiency Video Coding

ISO/IEC International Organization for Standardization and International Electrotechnical Commission

ITU-T International Telecommunication Union, Telecommunication Standardization Sector

JCT-VC Joint Collaborative on Video Coding

JVET Joint Video Exploration Team (on Future Video coding)

KDD Knowledge Discovery from Data

LB Low Delay B

LP Low Delay P

MB MacroBlock

ME Motion Estimation

ML Machine Learning

MPEG Moving Picture Experts Group

MV Motion Vector

MVC Multiview Video Coding

MV-HEVC Multiview High Efficiency Video Coding

NB Naïve-Bayes

PU Prediction Unit

QP Quantization Parameter

RA Random Access

RD Rate-Distortion

RDO Rate-Distortion Optimization

RExt Range Extension

RMD Rough Mode Decision

ROC Receiving Operator Characteristics

SAD Sum of Absolute Differences

SAO Sample Adaptive Offset

SHVC Scalable High Efficiency Video Coding

SI Spatial Index

SVC Scalable Video Coding

TI Temporal Index

TR Time Reduction

TU Transform Unit

UHD Ultra High Definition

VCEG Visual Coding Experts Group

Contents

Chapter 1

Introduction

This chapter introduces the common framework and the motivation for this Thesis, providing a justification for the work and an initial approach to the problem. Then, a description of the objectives is given, as well as a set of tasks to be carried out in order to achieve those objectives.

After this introduction, Chapter 2 shows the results (in the form of publications) which comprise the Thesis. Finally, the conclusions which can be drawn from the Thesis, as well as the future work that can be performed in this field, are presented in Chapter 3.

1.1 Motivation and Justification

Nowadays, video services represent the large majority of Internet traffic. The latest Cisco Visual Networking Index [13] reports that IP video traffic accounted for 70% of all Internet traffic in 2015, and they expect it to rise up to 82% by the year 2020. In the last few years many applications which make use of video streaming services have emerged (video conferences, *High Definition* (HD) online games, IP TV, etc.) and the number of these applications increases every year. Moreover, these applications tend to use much higher resolutions and frame rates, thus increasing the amount of data to be broadcast.

Because of this, it is very important to design new efficient video codecs which can reduce the bit rate of a video sequence with a minimal impact on *Rate-Distortion* (RD) performance. With this fact in mind, different video coding standards have been released since 1988. Figure 1.1 shows the history of the video coding standards released by the *Moving Picture Experts Group* (MPEG), which belongs to the *International Organization for Standardization* and the *International Electrotechnical Commission* (ISO/IEC), and the *Visual Coding Experts Group* (VCEG), which belongs to the *International Telecommunication Union*, Telecommunication Standardization Sector, (ITU-T).

Specifically, the H.264/MPEG-4 part 10 – *Advanced Video Coding* (AVC) standard [46] has been widely used to encode HD contents in the last twelve years. However, in April 2013, the *High Efficiency Video Coding* (HEVC) standard [47] was released by the *Joint Collaborative Team on Video Coding* (JCT-VC), which is a work group composed of experts

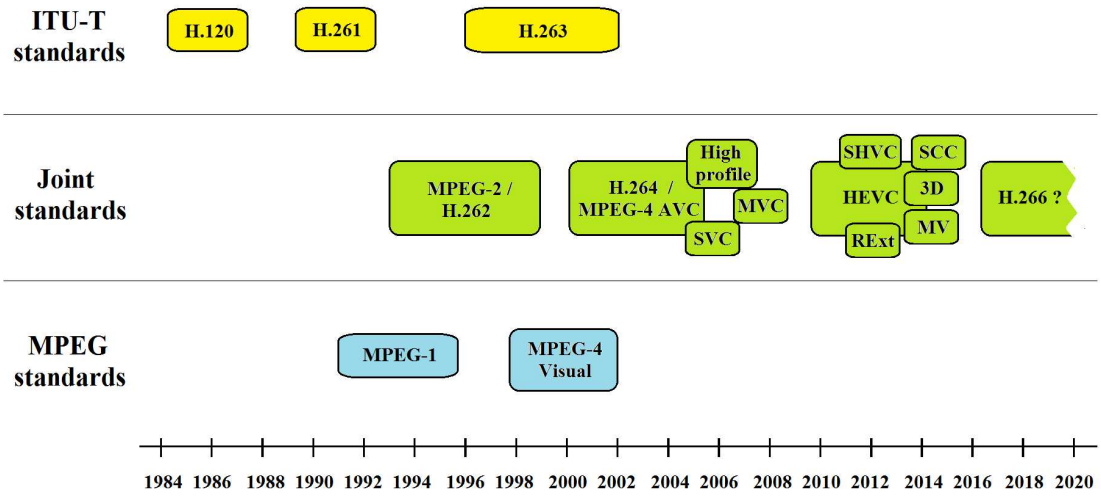


Figure 1.1: History of video coding standards.

from the MPEG and VCEG groups. This new standard is conceived as the natural successor to H.264/AVC and is able to double its compression performance in terms of objective quality, but at the expense of extremely high computational complexity [52]. In fact, when measuring the subjective quality, it has even been reported that HEVC is able to reduce the bit rate by 59% while maintaining the same quality perceived by the viewer [60].

HEVC has been specifically developed by taking into account the particularities of *Ultra HD* (UHD) resolutions, such as 4K UHD (3840×2160 pixels) or 8K UHD (7680×4320 pixels), in which the size of the frames is much higher than in HD. For this reason, the blocks into which a frame is split in order to be encoded may be much bigger than in H.264/AVC and previous standards, and the coding structure is much more flexible [59].

Thus, considering the superior RD performance of HEVC, as well as the number of sequences which are currently encoded with H.264/AVC, an adaptation of these sequences to the new HEVC standard is desirable. Moreover, backward compatibility with legacy H.264/AVC devices is required so that they do not become obsolete.

In this context, the conversion of a sequence which is encoded with a specific standard and configuration (i.e. bit rate, resolution, frame rate, ...) is called *transcoding* [65]. In our case, the conversion is focused on changing the standard that is used to encode the sequence, and this is called *heterogeneous transcoding*. If the objective were to change the configuration, it would be *homogeneous transcoding*. Heterogeneous transcoding is very helpful nowadays, since there are a wide range of devices on the market with different decoding capabilities, as can be seen in Figure 1.2. Therefore, content providers should transcode older video streams so that newer devices receive an HEVC encoded version of the video in order to decrease network traffic.

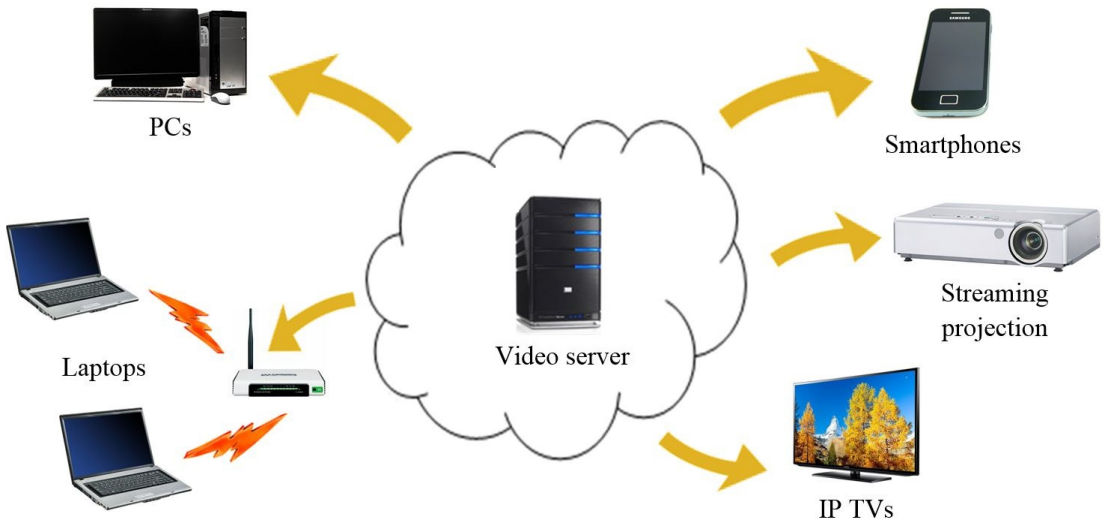


Figure 1.2: Different devices with different video decoding capabilities requesting the same sequence from a video server.

The transcoding process can be performed in an inefficient cascade way (i.e. decoding the original H.264/AVC stream and re-encoding it in HEVC), but this process requires a lot of time due to the high computational complexity of the HEVC encoder. It is logical to think that some information from the original H.264/AVC encoded stream can be reused to speed up the HEVC encoding. This idea of reusing information in a transcoder has already been addressed in the literature. For example, fast MPEG-2 [45] to H.264/AVC transcoding algorithms can be found in [34, 35]. There are even some works that focus on other codecs which have not been designed by a standardization organization, such as *Distributed Video Coding* (DVC) [1] or VP9 [41]. For example, some authors propose a DVC to H.264/AVC transcoder [15, 50], while other authores proposes a novel HEVC to VP9 transcoder [16]. Furthermore, as there still exists a big legacy of MPEG-2 encoded material, a fast MPEG-2 to HEVC transcoder has also been proposed in [58].

Another important question is what a content provider should do when they add a new video sequence to their repository that was not previously available in any format, since it can also be requested by different devices, as shown in Figure 1.2. In this case, once more, HEVC-compliant devices should be able to take advantage of the bit rate reduction provided by this standard, but backward compatibility is also necessary. In order to address this situation, as explained by *Adhikari et al.* in [2], in *Netflix*, one of the biggest companies in the video distribution sector, each movie is encoded with different bit rates and formats. When a movie is requested, a manifest file indicates the formats that the device can decode and the server decides which is the best one from among those contained in the manifest file.

This process of encoding the same sequence with different codecs and bit rates requires a very computationally complex system. However, in a similar way to that in the transcoding

scenario, information from the first encoding can be reused to accelerate the following encodings. The problem of encoding the same sequence with different bit rates using the same standard, which can be called *homogeneous simultaneous encoding* as an analogy of the transcoding scenario, has already been addressed in the literature for the case of VP8 [32, 36] and HEVC [18, 56].

However, *heterogeneous simultaneous encoding*, which consists in accelerating the encoding of a sequence in a specific standard by re-using the information from the encoding of that same sequence in other standard, has not been addressed before. Moreover, a heterogeneous simultaneous encoder can be more efficient than a homogeneous one, since the reference stream can be encoded with a much less computationally complex standard, thus increasing the speed-up of the overall simultaneous encoder.

Nevertheless, simultaneous encoding is not the only solution that can be used to provide backward compatibility to legacy devices. With the second version of the HEVC standard [48], the *Scalable* extension of HEVC (SHVC) [10] was formally defined. In this context, the scalable extension of H.264/AVC, namely H.264/*Scalable Video Coding* (SVC) [46], already defined 3 types of scalability: quality, temporal, and spatial. They make it possible to have a base layer with a specific quality (bit rate or quantization parameter), frame rate and frame resolution and one or more enhancement layers with a higher quality, more frames to increase the frame rate and/or a higher resolution.

Besides these types of scalability, SHVC defines a new type of scalability: hybrid scalability [6]. This allows the base layer of a video stream to be encoded using a different standard than HEVC, which, however, must be used for all the enhancement layers. Thus, hybrid SHVC allows those legacy devices to decode the base layer (which can be encoded using H.264/AVC), while the HEVC-compliant devices will be able to decode a higher quality version of the video. On the one hand, this approach has the advantage with respect to simultaneous encoding that the content provider does not need to encode the video several times, thus saving storage capacity. Moreover, the user does not need to send the manifest file, since the server will always send the SHVC compliant stream and according to user decoding capabilities and/or network traffic some layers will be discarded or not.

On the other hand, the main drawback of using SHVC is the overhead introduced in the bit stream, not only because of the replication of the same frames with a different resolution or quality in the enhancement layers, but also because even the HEVC-compliant devices will receive the H.264/AVC base layer, thus not taking full advantage of HEVC. Furthermore, HEVC enhancement layers will always be sent through the network, even for those devices which cannot decode them, adding an unnecessary load to the network.

It can be noticed that, while there are some previous works which aim at accelerating the scalability extensions of H.264/AVC [38, 39] or HEVC [40, 49], the research carried out in the Thesis is the first which involves hybrid scalability, accelerating the HEVC enhancement views with the information collected in the H.264/AVC base view.

This concept of hybrid video coding to enable backward compatibility has also been used in another extension of HEVC which was also released with the second version of the standard: *Multiview HEVC* (MV-HEVC) [61]. Multiview video consists in encoding several video views captured by multiple cameras at the same time, but with different angles. In the 2-view case, it is called *stereo video*, and it makes it possible to show a different view to each eye by means of *stereo glasses*, creating a sense of depth in the brain. When more views are available, *free viewpoint video* is the technology that switches which view is shown to each eye depending on the user's position. In this field, it can be easily seen that the structure of layers used in scalable video can also be utilized in multiview video in order to reduce the spatial redundancy among views, what makes both extensions very similar from a technical viewpoint.

Therefore, the hybridity concept from SHVC can also be used in MV-HEVC [64], allowing it to encode the base view in a different standard than HEVC, which is however used for all the other views. Thus, a more modern 3D device will be able to decode all the views and provide stereo or free viewpoint video to the user, while a legacy device (stereoscopic-capable or not) will be able to decode the base view and provide a non-3D version of the video at least. Once more, as in the SHVC case, the problem with hybrid MV-HEVC is the overhead that is introduced in the bit stream because of the use of H.264/AVC in the base view. However, in this case the frame replication problem on several layers is not present, since the frames in different views are actually different from each other. It should be pointed out that, as with the hybrid SHVC problem, there are no previous works aimed at accelerating hybrid MV-HEVC.

It is easy to see that the problem in the simultaneous encoding, the SHVC and the MV-HEVC cases, as in the transcoding scenario, is that a sequence takes much more time to be encoded with HEVC than with previous standards. This problem is even more acute in the SHVC and the MV-HEVC cases, since they have to perform inter-layer and inter-view motion estimation, respectively, making the encoder much more computationally complex. Therefore, in the real scenario of a company which wishes to implement these technologies, the process must be accelerated, even at the expense of introducing a small overhead in the bit stream.

1.2 Objectives

In line with what has been mentioned above, the main objective of the Thesis is to design a fast HEVC encoder for multiple scenarios which makes use of information extracted from H.264/AVC-compliant streams in order to take advantage of the superior compression capabilities of HEVC while maintaining backwards compatibility with H.264/AVC legacy devices. In order to address this main objective, the following goals are proposed as specific objectives:

- **Goal 1**: to carry out a thorough review of video coding technology, emphasizing understanding of the H.264/AVC and HEVC standards. This review should also include the study of the state-of-the-art proposals for HEVC acceleration.

- **Goal 2**: to design and evaluate a fast H.264/AVC to HEVC transcoding architecture to adapt already encoded sequences to the new HEVC standard.

- **Goal 3**: to develop and evaluate a fast simultaneous encoding algorithm for those new sequences so that any device on the market can decode them. In this case, the above transcoding architecture can be adapted to this new scenario.

- **Goal 4**: to design and evaluate an algorithm to reduce the computational complexity of an H.264/AVC and HEVC scalable encoder with hybrid and quality scalability. This algorithm needs to cope with the fact that the H.264/AVC and the HEVC layers are encoded with different Quantization Parameters (QPs).

- **Goal 5**: to reduce the computational complexity of a hybrid multiview H.264/AVC and HEVC encoder. In this case, the design must take into account the displacement between views, making it necessary to compensate for the different positions of elements in the frames.

1.3 Methodology and Work Plan

In order to achieve the main objective formulated in the previous section some specific objectives or goals have been formulated. This section details the methodology that has been used by defining the tasks into which each goal can be divided. The successful completion of these tasks will equate to the achievement of each goal. These are the proposed tasks (grouped by goal):

- **Goal 1** - *To review video technology and the state of the art*. The tasks that have been defined to achieve this goal are:

    a. To study the HEVC standard which, at the time of starting the Thesis, had not even been established by the standardization bodies. This study is motivated by the fact that HEVC introduced important changes with respect to its predecessors.

    b. To study the main differences between H.264/AVC and HEVC, and perform a comparison of coding efficiency and encoding time between the two standards.

    c. To review the first research works on HEVC fast encoding algorithms. Since, in general, most of the proposed algorithms do not use information from H.264/AVC to accelerate HEVC, these algorithms were the main field of study (though not limited to it).

d. To identify those variables that can be found during the H.264/AVC encoding or decoding stages and which can be profitably re-used during the HEVC encoding stage.

e. To review different *machine learning* algorithms from those available in the literature.

- **Goal 2** - *To design and evaluate fast H.264/AVC to HEVC transcoding architecture*. This is the most complex goal of the Thesis since, as can be observed, the methodology in the following goals is similar to the methodology in this goal. The following tasks are proposed for the achievement of this goal:

  a. To perform an analysis of the H.264/AVC reference software JM 18.4 [62] and fetch the variables identified in the previous goal from the decoder, dumping them into a file to build databases which will be used to train classifiers.

  b. To carry out an analysis of the HEVC reference software HM 16.2 [51], identifying the fragment(s) of code where the software implements the quadtree structure, which is the most complex tool among those introduced by HEVC.

  c. To design the proposed pruning algorithm, deciding which CUs and/or PUs will be checked or not. The specific machine learning algorithm will be plugged into this algorithm once it is chosen.

  d. To prepare different databases with part of the data fetched from H.264/AVC which will be used as training sets and test the machine learning algorithms with them, choosing the most appropriate algorithm to take the decisions on quadtree pruning.

  e. To implement the selected algorithm in the HM 16.2 software, reading the H.264/AVC information and taking the decision whether a branch of the quadtree should be pruned or not.

  f. To test the proposed algorithm on a wide range of sequences to measure its performance in terms of acceleration and RD-rate, which measures the increment in the bit rate needed to maintain the same objective quality [3].

- **Goal 3** - *To develop and evaluate a fast simultaneous encoding algorithm*. In order to achieve this goal, the following tasks have been defined:

  a. To extend the analysis of the JM 18.4 software to fetch variables that can only be found on the encoder side, since in this new scenario the H.264/AVC encoder is also present, not only the decoder.

  b. To prepare new databases with the new features and re-train the machine learning algorithm with them. In this case it is not necessary to investigate other machine learning algorithms, since the scope of the algorithm is essentially the same as in the previous goal (quadtree pruning) and if the algorithm worked for the previous case, it should also work in this case.

c. To implement the modifications of the classifier in the HM 16.2 software used for the previous goal in order for it to be adapted to the simultaneous encoding scenario.

d. To test the proposed algorithm using other sequences to measure its performance.

- **Goal 4** - *To design and evaluate an algorithm to reduce the computational complexity of an H.264/AVC and HEVC scalable encoder with hybrid and quality scalability*. The tasks that have been defined to achieve this goal are:

  a. To analyze the SHVC reference software SHM 6.1 [11] and implement the classifier obtained in the previous goal in the corresponding fragment of code. In this case a new classifier is not required, since the available information is the same as the previous case: that information that can be obtained from the H.264/AVC base layer encoder.

  b. To adapt the decision rule to the current scenario, since in this case the QP of each layer is different.

  c. To test the proposed algorithm in the scalable scenario with a wide range of sequences in order to evaluate it.

- **Goal 5** - *To reduce the computational complexity of a hybrid multiview H.264/AVC and HEVC encoder*. To achieve this goal we have defined the following tasks:

  a. To adapt the proposal for the SHVC scenario to the multiview case. Once more, the classifier is the same and, in this case, the software is also the same, since SHM can also encode multiview video. However, the adaptation is not trivial since in this case the views have a displacement of the scene and some new areas might appear, while others disappear. For this adaptation an algorithm to compensate for the difference between views when assigning an H.264/AVC *MacroBlock* (MB) to an HEVC *Coding Unit* (CU) is needed.

  b. To evaluate the proposed algorithm by the use of different multiview sequences.

1.4 General Discussion and Brief Description of the Main Proposal

As can be deduced from the objectives, the main proposal of the Thesis is a quadtree pruning algorithm based on machine learning and its adaptation to several scenarios by making certain changes. Once the review of video technology was completed, several machine learning algorithms were considered. In an initial approach, we thought of using decision trees, such as *C4.5* [55], which has been demonstrated to perform well on previous transcoders [35, 38]. However, we thought that a probabilistic approach might perform even better than classification trees and several probabilistic algorithms were tried. On the one hand, general *Bayesian networks* [44] were discarded since their computational complexity makes them unsuitable for our problem. On the other hand, the two algorithms which were

considered to model the problem in the most correct way were *Random Forest* [7] and *Naïve-Bayes* [37]:

- **Random Forest:** this kind of classifiers train several decision trees with a random subset of samples and a random subset of features from those contained in the training set. The final decision of the classifier is that label, c^*, which was chosen by the highest number of trees, that is, the label with the highest probability using the trees' outputs as frequencies.

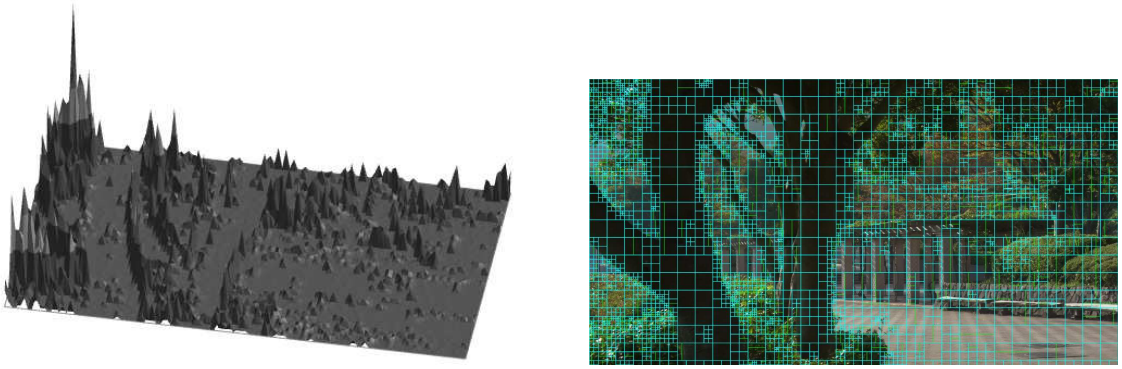
- **Naïve-Bayes:** in this case the classifier is based on probability theory. It calculates the *a posteriori* probability of each class label, $C_i \in \{C_1, C_2, \ldots, C_m\}$, given the observed set of features, $\{f_1, f_2, \ldots, f_n\}$: $P(C_i|f_1, f_2, \ldots, f_n)$. This calculation is performed according to the *Bayes' Theorem*, and the class label with the highest probability is chosen as:

$$\begin{aligned} c^* &= \underset{C\in\{C_1,C_2,\ldots,C_m\}}{argmax} P(C|f_1,\ldots,f_n) \\ &= \underset{C\in\{C_1,C_2,\ldots,C_m\}}{argmax} \frac{P(f_1,\ldots,f_n|C)P(C)}{P(f_1,\ldots,f_n)} \\ &= \underset{C\in\{C_1,C_2,\ldots,C_m\}}{argmax} P(f_1,\ldots,f_n|C)P(C). \end{aligned} \tag{1.1}$$

  However, the calculation of $P(f_1, \ldots, f_n|C)$ in Equation (1.1) requires a very high storage complexity and it might even produce overfitting. For this reason, the *naïve* assumption is made, which consists in assuming that the features are independent of one another given the class. Then, the previous probability can be approximated as $P(f_1, \ldots, f_n|C) \approx \prod_{i=1}^{n} P(f_i|C)$. Thus, during the training stage, this classifier learns the probability of each feature given the class, $P(f_i|C)$, by just counting the number of instances in the training set.

First of all, when preparing the databases, and regarding the class variable whose task is to decide the most appropriate level of the quadtree, there are two main options: a single classifier whose output is the maximum allowed depth (i.e. $C \in \{C_0, C_1, C_2, C_3\}$), or going through each node of the quadtree and deciding whether it should be further split or not (i.e. $C \in \{C_S, C_N\}$). As in the second case the class variable is of a binary nature, it is considered to be much easier to predict than the first case, so we decided to tackle the problem in this way.

Regarding the set of features to be used, a detailed analysis was carried out by taking into account those features previously used in other transcoders, as well as trying to identify the variables that might model the spatial and temporal complexity of the scene. Figure 1.3 shows the analysis which was performed. While Figure 1.3(a) shows the number of bits that H.264/AVC used to encode an MB in a specific frame, Figure 1.3(b) shows the CU splitting of HEVC for that same frame. One can clearly see the relationship between the number of



(a) Bits per MB in the H.264/AVC stream. (b) Original CTU splitting in HEVC.

Figure 1.3: Visual relationship between CTU splitting in HEVC and the number of bits used to encode the frame in H.264/AVC ($QP = 27$).

bits and the CU size, since those areas with bigger CUs correspond to the areas in which H.264/AVC used fewer bits. The choosing of these features is not limited to data from the H.264/AVC stream, but also to those which can be extracted from the frame aspect or those that can be calculated during the HEVC encoding stage.

During this process many features were added to the training set, since later it was cleaned up in order to remove those which in fact do not correctly model the problem and those that are redundant when another feature is given. More specifically, for this data cleaning up process, the *Fayyad and Irani* discretization algorithm [33] was used to avoid the assumption that they follow a *Normal* distribution when the classifier is learnt. Moreover, a subset of the available features was selected [42], since probabilistic classifiers are quite sensitive to the set of features that is used to learn them.

Furthermore, given the nature of the selected features, different databases should be built in accordance with two factors. Firstly, depending on the quadtree level a different number of H.264/AVC MBs will be mapped into a CU and a different scale will be applied to the features, so the databases were split according to the quadtree level: depth levels 0 (CU size of 64×64 pixels) and 1 (CU size of 32×32 pixels). At depth level 2, as the CU size is the same as the MB size in H.264/AVC (16×16 pixels), a more direct choice can be made by simply re-using the MB mode.

Then, as most digital video streams make use of hierarchical B frames [57], it should be taken into account that the scale of features in frames at different hierarchy levels may differ, as can be seen in Figure 1.4, where the number of bits of several frames after being encoded in HEVC is shown. In this case, which uses the *Random Access* configuration recommended in [5], besides the first column, which represents an intra-encoded frame, 4 different levels of hierarchy are defined (represented by columns with different patterns in Figure 1.4). Therefore, 4 databases should also be considered, which combined with the quadtree depths gives a total of 8 different training models.

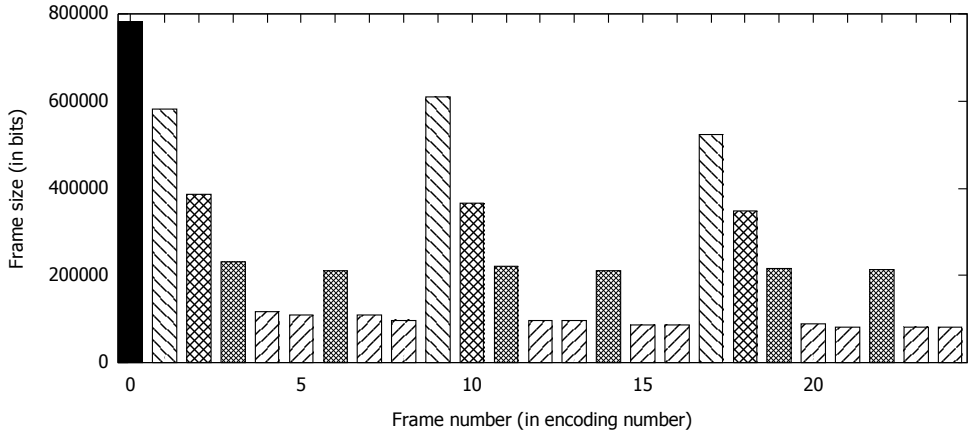


Figure 1.4: Number of bits of different frames according to hierarchy level. *Kimono* sequence with $QP = 22$.

Once the databases are ready, the *Random Forest* and *Naïve-Bayes* (NB) algorithms can be tested. Both models were trained and, by the use of *cross validation*, NB proved to the classifier that achieved the highest accuracy, which is why it was chosen. As a final step in the classifier learning process, it must be considered that a wide range of sequences might be encoded or transcoded by the use of the proposed collaborative encoding architecture, with different spatial and temporal characteristics. To address this issue, the probabilistic models that were learnt can be adjusted thanks to their binary nature. In this case, the classification rule can be described as choosing C_N if $P(C_N) > P(C_S)$, and C_S otherwise. Then, weighting parameters can be introduced to this rule to make it more costly to adopt one of the decisions: $P(C_S) \times Cost_{SN} > P(C_N) \times Cost_{NS}$, where $Cost_{SN}$ represents the cost of incorrectly classifying as C_N, and $Cost_{NS}$ is the cost of the opposite error. These costs can be learnt for each sequence in an *on-line* process by the use of the first few frames.

Adaptation to each specific scenario

Once the main guidelines for the algorithm are given, the adaptation for each specific scenario must be studied. In the case of the transcoder the sequence encoded in HEVC is the same as the one encoded in H.264/AVC (with a re-quantification process), so it can be considered as the base case, and then one can state the differences in other cases in which the sequences are not exactly the same in both cases. In the *transcoder scenario* the H.264/AVC decoding information and the HEVC encoding information can be obtained, so the feature list must be built with these data.

The next case is the *simultaneous encoding scenario*, in which, once more, both streams are the same for H.264/AVC and HEVC (without the re-quantification). However, in this case the H.264/AVC encoder is also present and more information can be re-used, such as the Lagrangian costs of each MB mode and the best *Motion Vector* (MV) for each mode.

Moreover, the $Cost_{SN}$ in the classification rule must be higher than in the transcoding problem since the absence of re-quantification makes it more likely for a CU to be split than in the previous case.

Regarding the *hybrid SHVC with quality scalability*, once more the sequence that will be encoded is the same in both standards and the information from the H.264/AVC encoder can be used, but with a different QP in each case. However, as in the previous case, none of the sequences is subjected to a re-quantification process, so in this case the same original stream is encoded with different QP values. Because of this, once more the $Cost_{SN}$ is higher than in the transcoding problem, but in this case the probabilities obtained from the NB are weighted before the costs are applied. Higher QP values tend to produce larger CU sizes, since when the quantization is too aggressive, the area looks smoother. Because of this, and given that the QP values of the enhancement layers (HEVC layers) are lower than the QP of the base layer, the scaled splitting probability is given by: $\tilde{P}(C_S|\mathbf{F}) = P(C_S|\mathbf{F}) \times QP_{EL}/QP_{BL}$, where QP_{BL} and QP_{EL} are the QP values in the base and the enhancement layers, respectively.

Finally, in the *multiview scenario* the H.264/AVC encoding information can be used again and the $Cost_{SN}$ is, once more, higher than in the transcoding case. However, the streams are not exactly the same in both standards, since the views have a horizontal displacement of some pixels between them. In this case, to tackle this issue a fast pre-fetch of that displacement should be performed, since an MB in the base H.264/AVC view might not correspond to the collocated CU in the other HEVC views. Then, 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.

1.5 Results

The first step in the course of the Thesis was, as stated in **Goal 1** in Section 1.3, the review of video coding technology, as well as machine learning algorithms. A very primary work that correspondes to this step was the study of the differences between H.264/AVC and HEVC, as well as the coding performance comparison. This comparison was firstly presented in the paper *'HEVC: a Review, Trends and Challenges'* [23], published in the *"II Workshop on Multimedia Data Coding and Transmission (WMDCT 2012)"*.

After the first goal was completed, the tasks associated with **Goal 2** were started, resulting in a transcoding architecture for inter frames which can save up to 54% of HEVC encoding time with a BD-rate [3], which measures the increment in the bit rate to maintain the same objective quality, of only 2.7%. The achievement of this goal was not trivial at all, since it involved the most complex goal of the Thesis. Once both codes, namely JM and HM, were studied, an initial idea was to re-use the reference frames used in H.264/AVC in order to

restrict the search of the Motion Estimation process in HEVC to those frames. However, after several experiments, we reached the conclusion that this way was not going to provide good results given that the BD-rate was not as low as firstly expected. Moreover, the theoretical maximum acceleration that could be reached was not very high. Nevertheless, these results were shown in the paper *'Multiple Reference Frame Transcoding from H.264/AVC to HEVC'* [24], published in *"The 20th International Conference on Multimedia Modeling (MMM 2014)"*.

After this way was ruled out, the next way which was explored was to accelerate the quadtree splitting by means of machine learning techniques, as discussed in the previous section. The final algorithm was developed in several steps. A first approach to the final algorithm was implemented on HM 12.0 (a version which was later turned out to contain a bug which was affecting the performance of our algorithm [63]), and it only used information from the H.264/AVC decoder, and not from the HEVC encoder, as described above. Furthermore, this first version, which was called the *Fast Quadtree Level Decision* (FQLD) algorithm, did not take into account the different hierarchy levels of B frames, the classifying algorithm at level 2 was simpler than in the final version, and the sequence adaptation given by the misclassification costs was not implemented. The results for this version of the algorithm were presented in the paper entitled *'Fast Quadtree Level Decision Algorithm for H.264/HEVC Transcoder'* [31], published in the *"IEEE International Conference on Image Processing (ICIP 2014)"*. The results at this point showed up to a 55% time saving with a BD-rate of 4.8%. It can be noted that the acceleration was approximately the same as in the final version, as mentioned above, but with a higher BD-rate, which means that the subsequent efforts were made to decrease the BD-rate.

Then, this algorithm was also combined with hardware parallelization algorithms in collaboration with another researcher. More specifically, it was combined with a thread-level algorithm which encoded different *Groups of Pictures* (GOPs) in parallel threads and with a *Graphics Processor Unit* (GPU) algorithm which calculated the SAD of different MVs for the same PU in parallel. The combination of the algorithms reduced the HEVC encoding time by 86% with a BD-rate of 7.4%. These results were presented in *'Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding'* [19], which was published in the *"Journal of Real-Time Image Processing (2016)"*.

A second version of the algorithm was published in the *"Data Compression Conference (DCC 2015)"*, in the paper entitled *'Fast Quadtree Level Decision Algorithm for H.264/HEVC Transcoder'* [27]. This time, the algorithm was implemented on HM 16.6 and the sequence adaptation was by means of costs calculation. HEVC information was used for classification, and the newer version for level 2 classification was implemented.

The final version of the algorithm was re-named at this point as *Adaptive FQLD* (AFQLD), in reference to its capability of adaptation to each specific sequence. It also included the difference of applying the algorithm to different hierarchical levels of B frames and is discussed in *'Adaptive Fast Quadtree Level Decision Algorithm for H.264/HEVC*

Video Transcoding' [29], published in the *"IEEE Transactions on Circuits and Systems for Video Technology (2016)"*. This article is part of the Thesis (see page 19). Moreover, this paper included a detailed performance evaluation for different configurations, the algorithm was compared with another state-of-the-art transcoding algorithm and with a non-transcoding algorithm, the so called *Early CU Termination* (ECU) [12], which is included in the HM reference software, and the results showed a substantial improvement with respect to it. Finally, the actual hit rate of the classifiers was also analyzed, showing that it is higher than 90%, which is a very high value for a real application, and very difficult to improve upon.

Another paper, *'A Statistical Approach of a CTU Splitting Algorithm for a H.264/AVC to HEVC Video Transcoder'* [28], published in the *"Proceedings of the 15th International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE 2015)"*, showed a complete comparison of the proposed algorithm with the aforementioned non-transcoding algorithm included in the HM software. This paper concluded with statistical support (by means of hypothesis contrasts) that the proposed algorithm performs better in both acceleration and BD-rate terms than the ECU algorithm. Moreover, this paper also demonstrated that the combination of the proposed algorithm with the *Early Skip Detection* (ESD) [66] and the *Coded Block Flag* (CBF) *Fast Mode* (CFM) [43] algorithms is also more efficient than the combination of these two algorithms with ECU.

Later, this transcoding architecture for inter frames was also extended to intra frames, since sequences in which all frames are coded as intra can be very valuable in some scenarios, such as video editing or post-producing. Nevertheless, it should be noted that it might not be a good idea to use this fast algorithm in GOP structures other than all intra, since introducing errors in intra frames (even if they are small) may propagate the error to inter frames, leading to a substantial performance drop. Regarding the technical aspects, in this case the scenario changes, since no hierarchical frames are present, which results in having only 2 models (for depth levels 0 and 1), and less information is present. For instance, there are no MVs and there is only one PU to check at each depth level, i.e. 2N×2N (except at level 3, in which the N×N PU can also be tested). The adaptation of the proposed transcoding architecture for intra frames was presented in the paper entitled *'A Fast Splitting Algorithm for an H.264/AVC to HEVC Intra Video Transcoder'* [30], which was published in the *"Data Compression Conference (DCC 2016)"*.

Furthermore, in this intra transcoder, the intra mode decision was also accelerated. HEVC introduces up to 33 directional modes for intra prediction, which is a much larger number than the 8 available modes in H.264/AVC. Even though the HM reference already makes use of a fast algorithm to select the mode, it still requires a lot of time. The algorithm used by HM is the one presented by *Piao et al.* in [54]. It performs a fast *Rough Mode Decision* (RMD) over all possible modes and only performs the full *Rate-Distortion Optimization* (DRO) over the best few modes in the previous phase. However, as in this case H.264/AVC already chose a direction, it can be re-used so that the search in HEVC only

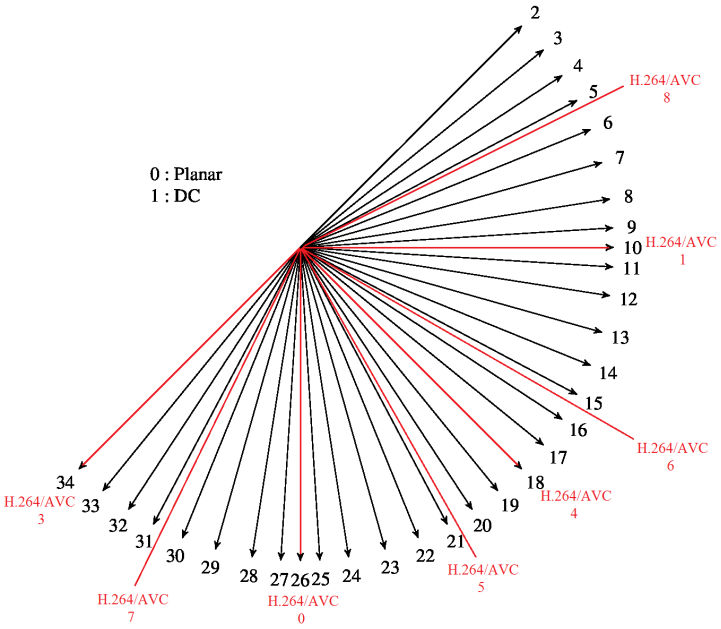


Figure 1.5: Correspondence between HEVC and H.264/AVC intra directional modes.

needs to be performed over the modes adjacent to the direction chosen by HEVC (see Figure 1.5). Moreover, this algorithm can be combined with the quadtree level decision algorithm to achieve a further accelerated transcoder. A paper entitled *'A Fast Intra H.264/AVC to HEVC Transcoding System'* [26], which includes the combination of these two intra algorithms, has been sent to the *"Multimedia Tools and Applications"* journal and is currently under review.

Regarding the other scenarios of collaborative video coding, some results have also been achieved. As a first step for the H.264/AVC and HEVC hybrid video encoder, the second version of the FQLD algorithm (the one presented in [27]) was adapted to this scenario. To achieve this, the difference between views was compensated for as described in the previous section. This work was published in the paper *'Using Bayesian Classifiers for Low Complexity Multiview H.264/AVC and HEVC Hybrid Architecture (MLSP)'* [21], which was presented in the *"IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2015)"*. This work was ranked among the top 10% papers in the conference and, therefore, we were invited to submit an extended version to an associated journal.

The extension consisted in using the final version of the AFQLD algorithm in the hybrid multiview scenario, achieving a 70% time reduction in the HEVC views. To carry out this task, the final AFQLD algorithm was also extended with the MB position compensation between views algorithm. Furthermore, information from the H.264/AVC encoder, which was not being taken into account in the previous version, was also used. Finally, it was published in the *"Journal of Signal Processing Systems (on-line 2016)"* under the title *'Reducing the Complexity of a Multiview H.264/AVC and HEVC Hybrid Architecture'* [20], a paper that can be found in the Thesis on page 69.

Regarding simultaneous H.264/AVC and HEVC encoding, after the adaptations described in the previous section were made to adapt the proposal to this scenario, a quantitative time reduction of 53% was achieved. This proposal and its results were described in *'CTU splitting algorithm for H.264/AVC and HEVC simultaneous encoding'* [22], a paper that was published in *"The Journal of Supercomputing (on-line 2016)"* (see page 37). Furthermore, another part of the HEVC encoder in the framework of the simultaneous encoder was accelerated. As the H.264/AVC encoder is present, the best MV for each MB partition (and specifically for the 16×16 partition, which is the biggest available and most similar to bigger CUs or PUs in HEVC), these MVs were re-used in order to provide an initial predictor for the integer MV search. This work was presented in the paper entitled *'A Motion Vector Re-Use Algorithm for H.264/AVC and HEVC Simultaneous Video Encoding'* [8], published in *"The 13th International Conference on Advances in Mobile Computing and Multimedia (MoMM 2015)"*.

Finally, the article *'A Fast Hybrid Scalable H.264/AVC and HEVC Encoder'* [25], which was published in *"The Journal of Supercomputing (on-line 2016)"* and is part of the Thesis (page 53), presents the adaptation of the AFQLD algorithm to the hybrid SHVC scenario with quality scalability as described in the previous section. In this case the algorithm obtains a 60% time reduction for the HEVC layer with a slight penalty of 3.0% in terms of BD-rate. Currently, the work is being extended so that the algorithm may be used with temporal scalability. However, this adaptation is not trivial, since when using temporal scalability, the frames in all views are not the same (this also applies for spatial scalability). In this case, a motion compensation procedure should be developed between different frames.

Summary of Results

To sum up, once all the goals proposed in Section 1.2 have been achieved by the completion of the different tasks enumerated in Section 1.3, this is the list of works that have been published (or that are currently under review):

- **Goal 1** - *To review of the video technology and the state of the art*: one conference article.
  - *'HEVC: a Review, Trends and Challenges'*, published in the *"II Workshop on Multimedia Data Coding and Transmission (WMDCT 2012)"*: a performance comparison of H.264/AVC and HEVC is made.
- **Goal 2** - *To design and evaluate a fast H.264/AVC to HEVC transcoding architecture*: six conference articles, one journal article (JCR Q1), and one journal article under review.
  - *'Multiple Reference Frame Transcoding from H.264/AVC to HEVC'*, published in *"The 20th International Conference on Multimedia Modeling (MMM 2014)"*:

it presents a reference frame re-use algorithm to reduce the number of reference frames in HEVC.

  - *'Fast Quadtree Level Decision Algorithm for H.264/HEVC Transcoder'*, published in the *"IEEE International Conference on Image Processing (ICIP 2014)"*: the first version of the FQLD algorithm is presented.
  - *'Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding'*, published in the *"Journal of Real-Time Image Processing (2016)"*: this paper proposes a combination of the FQLD algorithm with parallelization techniques.
  - *'Fast Quadtree Level Decision Algorithm for H.264/HEVC Transcoder'*, published in the *"Data Compression Conference (DCC 2015)"*: an improved version of the FQLD algorithm is presented.
  - *'Adaptive Fast Quadtree Level Decision Algorithm for H.264/HEVC Video Transcoding'*, published in the *"IEEE Transactions on Circuits and Systems for Video Technology (2016)"*: the final version of the (now) called AFQLD algorithm is presented. This paper is included in the Thesis on page 19.
  - *'A Statistical Approach of a CTU Splitting Algorithm for a H.264/AVC to HEVC Video Transcoder'*, published in the *"Proceedings of the 15th International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE 2015)"*: it shows a statistical comparison of the AFQLD algorithm with other state-of-the-art acceleration approaches.
  - *'A Fast Splitting Algorithm for an H.264/AVC to HEVC Intra Video Transcoder'*, published in the *"Data Compression Conference (DCC 2016)"*: it presents the AFQLD algorithm for intra frames in the framework of the transcoder.
  - *'A Fast Intra H.264/AVC to HEVC Transcoding System'*, under review for publication in the *"Multimedia Tools and Applications"* journal: a combination of the intra AFQLD algorithm and a fast algorithm for intra directional mode decision is described in this paper.

- **Goal 3** - *To development and evaluate a fast simultaneous encoding algorithm*: one conference article, and one journal article (JCR Q2).

  - *'Using Bayesian Classifiers for Low Complexity Multiview H.264/AVC and HEVC Hybrid Architecture'*, presented in the *"2015 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)"*: an adaptation of the FQLD algorithm to the hybrid multiview scenario is presented. Selected among the top 10% best papers of the conference.
  - *'Reducing the Complexity of a Multiview H.264/AVC and HEVC Hybrid Architecture'*, published in the *"Journal of Signal Processing Systems"*: extension of the previous paper, using the final AFQLD algorithm and

information from the H.264/AVC encoder (not only from the decoder). This paper is included in the Thesis on page 69.

- **Goal 4** - *To design and evaluate an algorithm to reduce the computational complexity of an H.264/AVC and HEVC scalable encoder with hybrid and quality scalability*: one conference article, and one journal article (JCR Q2).
  - *'CTU splitting algorithm for H.264/AVC and HEVC simultaneous encoding'*, published in *"The Journal of Supercomputing"*: it presents an adaptation of the AFQLD algorithm to the simultaneous encoding scenario, making use of the H.264/AVC encoder information. This paper is included in the Thesis on page 37.
  - *'A Motion Vector Re-Use Algorithm for H.264/AVC and HEVC Simultaneous Video Encoding'* [8], published in *"The 13th International Conference on Advances in Mobile Computing and Multimedia (MoMM)"*: an algorithm for MV re-use in order to build a good initial predictor is presented in the article.
- **Goal 5** - *To reduce the computational complexity of a hybrid multiview H.264/AVC and HEVC encoder*: one journal article (JCR Q2).
  - *'A Fast Hybrid Scalable H.264/AVC and HEVC Encoder'*, published in *"The Journal of Supercomputing"*: it presents an adaptation of the AFQLD algorithm to be used in the hybrid encoder with quality scalability. This paper is included in the Thesis on page 53.

Finally, even though the following papers are not directly related to the topic of the Thesis, the candidate is also co-author of them and has collaborated in their production:

- *'Fast Motion Estimation for Closed-Loop HEVC Transrating'* [53], published in *"2014 IEEE International Conference on Image Processing (ICIP)"*.
- *'Fast Simultaneous Video Encoder for Adaptive Streaming'* [18], published in *"2015 IEEE International Workshop Multimedia Signal Processing (MMSP)"*. Selected among the top 10% best papers of the conference.
- *'Simultaneous Encoder for High-Dynamic-Range and Low-Dynamic-Range Video'* [17], under revision for publication in *"IEEE Transactions on Consumer Electronics"*.

Chapter 2

Publications

This chapter shows the selected publications which are included in the Thesis and in which the candidate is the first author.

2.1 Adaptive Fast Quadtree Level Decision Algorithm for H.264 to HEVC Video Transcoding

- **Title:** Adaptive Fast Quadtree Level Decision Algorithm for H.264 to HEVC Video Transcoding.
- **Authors:** Antonio Jesús Díaz-Honrubia, José Luis Martínez, Pedro Cuenca, José Antonio Gámez, and José Miguel Puerta
- **Type:** Journal
- **Journal:** IEEE Transactions on Circuits and Systems for Video Technology
- **Publisher:** IEEE
- **ISSN:** 1051–8215
- **State:** Published
- **Year:** 2016
- **DOI:** 10.1109/TCSVT.2015.2473299
- **URL:** http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7225148
- **Category**: Engineering, Electrical & Electronic
- **Impact Factor:** 2.254
- **JCR ranking:** Q1

Adaptive Fast Quadtree Level Decision Algorithm for H.264 to HEVC Video Transcoding

Antonio Jesús Díaz-Honrubia, *Student Member, IEEE*, José Luis Martínez, Pedro Cuenca, José Antonio Gamez, and José Miguel Puerta

***Abstract*— High Efficiency Video Coding (HEVC) was developed by the Joint Collaborative Team on Video Coding to replace the current H.264/Advanced Video Coding (AVC) standard, which has dominated digital video services in all segments of the domestic and professional markets for over ten years. Therefore, there is a lot of legacy content encoded with H.264/AVC, and an efficient video transcoding from H.264/AVC to HEVC will be needed to enable gradual migration to HEVC. In terms of rate–distortion (RD) performance, HEVC roughly doubles the RD compression performance of H.264/AVC at the expense of a high computational cost. HEVC adopts a quadtree-based coding unit (CU) block partitioning structure that is flexible in adapting various texture characteristics of images. However, this causes a dramatic increase in computational complexity due to the necessity of finding the best CU partitions. This paper presents an adaptive fast quadtree level decision algorithm that is designed to exploit the information gathered at the H.264/AVC decoder in order to make faster decisions on CU splitting in HEVC using a Naïve–Bayes probabilistic classifier that is determined by a supervised data mining process. The experimental results show that the proposed algorithm can achieve a good tradeoff between coding efficiency and complexity compared with the anchor transcoder; moreover, it outperforms other related works available in the literature.**

***Index Terms*— Coding unit (CU) splitting, H.264/Advanced Video Coding (AVC), High Efficiency Video Coding (HEVC), software partitioning, transcoding.**

I. Introduction

IN APRIL 2013, the High Efficiency Video Coding (HEVC) standard [1] was finalized by the Joint Collaborative Team on Video Coding (JCT-VC). More recently, in October 2014, the second edition of the HEVC standard [2] was completed with three important extensions, which include support for more color formats and higher bit depths (Range Extension), support of spatial and fidelity scalability (Scalable High Efficiency Video Coding Extension), and support of multiview video coding (MultiView High Efficiency Video Coding Extension).

The new HEVC standard is a natural evolution of its predecessor, namely, H.264/MPEG4 part 10—Advanced Video Coding (AVC) standard [3]. HEVC was initially conceived with the purpose of not only achieving a highly efficient performance for delivering high-quality multimedia services over bandwidth-constrained networks but also giving support to ultrahigh definition (so-called 4k, 3840 × 2160 pixels, and 8k, 7680 × 4320 pixels), which demand a high bandwidth. While HEVC can bring respite to content producers, aggregators, distributors, and consumers with a higher quality content at the same bitrate, the adoption curve could still be years away.

In terms of rate–distortion (RD) performance, HEVC roughly doubles the RD compression performance of H.264/AVC, but at the cost of extremely high computational and storage complexities during encoding [4]. Among other features, HEVC includes multiple new coding tools, namely, highly flexible quadtree coding block partitioning, which includes new concepts such as coding unit (CU), prediction unit (PU), and transform unit (TU) [5].

Considering both the superior compression performance of HEVC and the large body of content that is currently encoded using the H.264/AVC standard, a transcoder that can convert H.264/AVC bitstreams into HEVC bitstreams is of great value in many applications, especially before dedicated HEVC encoder systems become widely available, while at the same time, various software-based HEVC decoders have been demonstrated [6]. Furthermore, there is a wide availability of H.264/AVC encoders on the market with a good tradeoff in terms of RD performance and low cost. Thus, an H.264/AVC encoder working in tandem with an efficient H.264/AVC to HEVC transcoder may provide a cost-effective means of performing HEVC encoding for many applications in the absence of dedicated HEVC encoders. Therefore, there is a double motivation for an H.264/AVC to HEVC transcoder: on the one hand, to provide interoperability for the legacy video encoded with H.264/AVC when new devices using HEVC emerge and, on the other hand, to take advantage of the superior RD performance of the HEVC standard.

However, all the previously mentioned new coding tools involve a considerable increase in the encoding time in HEVC. With this challenge in mind, this paper presents a soft computing approach, which we have called Adaptive Fast Quadtree Level Decision (AFQLD), which aims to

Manuscript received October 14, 2014; revised March 17, 2015 and June 26, 2015; accepted August 13, 2015. Date of publication August 26, 2015; date of current version January 6, 2015. This work was supported in part by the Ministry of Economy within Competitiveness and European Commission through FEDER Funds under Project TIN2012-38341-C04-04 and Project TIN2013-46638-C3-3-P and in part by the Spanish Ministry of Education, Culture and Sports under Grant FPU 12/00994. This paper was recommended by Associate Editor G. J. Sullivan.

A. J. Díaz-Honrubia, J. L. Martínez, and P. Cuenca are with the High Performance Networks and Architectures Group, University of Castilla–La Mancha, Albacete 13071, Spain (e-mail: antonio.dhonrubia@uclm.es; joseluis.martinez@uclm.es; pedro.cuenca@uclm.es).

J. A. Gamez and J. M. Puerta are with the Intelligent Systems and Data Mining Group, University of Castilla–La Mancha, Albacete 13071, Spain (e-mail: jose.gamez@uclm.es; jose.puerta@uclm.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2015.2473299

exploit the information gathered in the H.264/AVC decoder in order to assist decisions on CU splitting in HEVC using a statistical Naïve–Bayes (NB) classifier to avoid an exhaustive rate–distortion optimization (RDO) search over all possible CU sizes and their modes. Adaptive refers to the fact that the algorithm can dynamically be adapted to the content of each sequence. In an offline data mining process, all the knowledge needed is extracted from the H.264/AVC decoding statistics by means of machine learning (ML) techniques, and is then converted into mathematical models that can be executed in the on-line transcoding process. In other words, our proposal substitutes the brute force scheme used in HM implementation with a low complexity algorithm based on an NB classifier. The experimental results show that the proposed algorithm, compared with the anchor transcoder, can achieve speedups of 2.5× on average over the full set of the HEVC common test sequences, with about a 4% loss in efficiency in terms of Bjontegaard delta rate (BD rate) [7], which measures the increment in bitrate while maintaining the same objective quality.

The remainder of this paper is organized as follows. Section II includes technical background to the new HEVC standard, focusing on coding block partitioning, while Section III identifies the related work being carried out on the topic. Section IV introduces our proposed AFQLD algorithm, and the experimental results are given in Section V. Section VI concludes this paper.

II. Technical Background

HEVC can be considered an evolution of the current H.264/AVC, since it maintains the same block-based hybrid approach used in all previous video compression standards. In addition, new tools have been introduced in HEVC that increase its coding efficiency compared with H.264/AVC, such as the new CU partitioning based on a hierarchical block structure named the coding tree unit (CTU), new transform sizes of 16×16 and 32×32, and a new tool in the decoding loop called sample adaptive offset, which is applied to the reconstructed samples after the deblocking filter with the goal of improving the perceived quality of the decoded sequence. Due to space constraints, a more complete description of these tools and a general HEVC architecture description can be found in [5].

One of the most important changes affects picture partitioning. HEVC defines a new flexible CTU structure [8], which is a replacement of the CU based on nonoverlapping 16×16 pixel blocks (MacroBlocks), as was defined in the previous video coding standards. With the aim of achieving an optimal adaptation of the CU to the content details, the CTU size can vary from a size of 64×64 pixels to a size of 16×16 pixels, and each CTU can iteratively be partitioned into four square sub-blocks of half resolution, named CUs, with a minimum allowable size of 8×8 pixels. Therefore, a CTU can be further partitioned into four depth levels, from $d = 0$ (CU size of 64×64) to $d = 3$ (CU size of 8×8), having 4^d CUs in each depth level. Thus, a CU in depth level d can be denoted by $\mathrm{CU}_{d,k}$ ($k = 0, 1, \ldots, 4^d - 1$).

HEVC increases CTU flexibility, each $\mathrm{CU}_{d,k}$ becoming a root of two new trees containing two new unit types: PUs, and the TUs. For intra-picture prediction, a PU uses the same $2N \times 2N$ size as for the $\mathrm{CU}_{d,k}$ to which it belongs, allowing it to be split into quad $N \times N$ PUs only for CUs at the minimum depth level. Therefore, the PU size can range from 64×64 to 4×4 pixels. For inter-picture prediction, several nonsquare rectangular block shapes are available in addition to square ones, allowing eight different PU sizes ($2N \times 2N$, $2N \times N$, $N \times 2N$, $N \times N$, $2N \times nU$, $2N \times nD$, $nL \times 2N$, and $nR \times 2N$). The prediction residue obtained in each PU is transformed using the residual quadtree structure [9], which supports various TU sizes from 32×32 to 4×4. In Fig. 1, an example of the partitioning is shown, depicting how a CTU is structured in a hierarchical tree where each CU branch ends in a leaf ($\mathrm{CU}_{d,k}$), which is the root for the two new prediction and transform trees, containing the PU and TU trees.

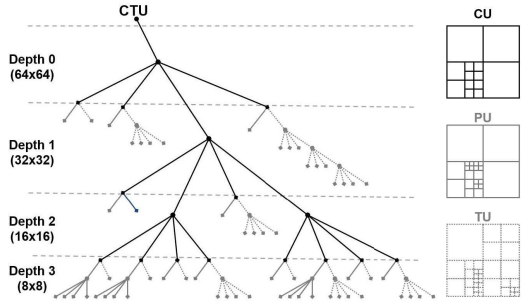


Fig. 1. Partitioning of CTU into CUs, PUs, and TUs.

HEVC checks most of the PUs (inter and intra modes) to decide whether it should split a CU or not by choosing the best RD case by computing the well-known RDO model [10], which requires the evaluation of the total number of possible combinations. This means that the RDO model needs to evaluate the total number of available CUs sizes for the CTU, the total number of prediction modes for each CU, and also the total number of TU sizes. Furthermore, in the case of inter prediction, for each of these PU partitions, a motion estimation (ME) algorithm is called, and in the case of intra prediction, for each of these PU partitions, a total of 35 different coding modes are evaluated. As expected, the selection of the optimal partitioning for each of the three trees is an intensive time-consuming process due to the huge number of combinations that have to be evaluated in order to achieve the best performance [11], [12].

To reflect the practical importance of reducing the HEVC encoder complexity, three optional fast mode decision schemes have been adopted in HEVC reference software, called HEVC test model (HM) [13]. For fast PU decision, the early skip detection (ESD) [14] method and the coded block flag (CBF) fast mode (CFM) [15] method terminate remaining PU decision processes when predefined specific conditions are met. In other words, the ESD checks inter $2N \times 2N$ first and terminates if motion vector (MV) difference is zero and CBF is equal to zero (i.e., values of luma and two chromas are zero after encoding current PU mode). The CFM terminates if PU has CBF equal to zero. On the other hand, the early CU termination (ECU) [16] method terminates the CU splitting if skip is

determined as the best mode of current CU. In this way, a fast decision of CU depth is achieved. The proposed optional fast mode decision schemes reduce the HEVC encoder complexity by reducing the coding efficiency.

As mentioned previously, in the current HEVC reference software, in order to encode a given CU, the encoder tries all possible prediction modes. It is obvious that this scheme, by trying all and selecting the best, obtains the best partitioning of CU in RD terms at the expense of an extremely high computational complexity.

The complexity analysis described in [11] and [12] shows that examining all possible modes is the most time-consuming part of the total encoding process. Based on these observations, we present an innovative approach to jointly optimize the decision mode and ME. In our proposal, the CTU splitting computation problem is posed as a data classification problem where the information gathered in the H.264/AVC decoder assists decisions on CU splitting using a statistical NB classifier to avoid an exhaustive RDO search over all possible CU sizes and its possible modes, as mentioned in the previous sections. Unlike the reference software where MVs are estimated for all inter-mode PU types, in our approach, no ME is required for a particular CU size if that size is not selected by the statistical NB classifier. In the same way, no intra-prediction is required for a particular CU size if that size is not selected by the statistical NB classifier. The proposed approach will perform the ME and the intra prediction only for all the possible PU types belonging to the final CU size determined by the classifier, considerably reducing the huge RDO complexity in the CTU splitting algorithm of the HEVC.

III. Related Work

Video transcoding is the process of converting a compressed video stream encoded with a given format or characteristics into another video stream encoded with a different codec or features. The simplest transcoding process should perform the complete process of decoding and fully re-encoding [17], but this is not time effective. Proposals available in the literature try to avoid unnecessary operations in the second part of the transcoder (encoding stage) or even to accelerate complex modules simply using information collected in the decoding process as part of the transcoder.

In the framework of this paper, there are currently few approaches that deal with the problem of converting streams already encoded in H.264/AVC into the new HEVC standard. The first approach was proposed in 2012 and presented in [18], focusing on reducing the number of CU and PU partitions to be checked by means of an improved RDO metric. The time reduction achieved is an acceptable 70%–80% at the expense of a great RD penalty of 30%. In the same year, Peixoto and Izquierdo [19] proposed the reuse of MVs as well as a similarity metric to decide which HEVC CU partitions should be tested. This proposal obtains a maximum of 4.13× speedup with a rate penalty up to 10.92%. One year later, Peixoto *et al.* [20] proposed two alternatives to map H.264/AVC MBs into HEVC CUs based on an ML model: one of them works offline and the other uses a dynamic training on-line stage. An extension of [19] and [20] was published in [21], where the first k frames of the sequence are used to compute the parameters so that the transcoder can learn the mapping for that particular sequence. Then, two different types of mode mapping algorithms are proposed. In the first solution, a single H.264/AVC coding parameter is used to determine the outgoing HEVC partitions using dynamic thresholding. The second solution uses linear discriminant functions to map the incoming H.264/AVC coding parameters to the outgoing HEVC partitions; this solution is called proposed transcoder for content modeling using linear discriminant functions (PTCM-LDF) and will be used in this paper to compare our approach with related works. The first solution obtains a tradeoff between the speedup and bitrate increases of 3.08× and 16.2%, respectively. More recently, Peixoto *et al.* [22] have proposed a further step in the framework of H.264/AVC to HEVC transcoding; the proposal deals with a solution also based on ML to map H.264/AVC MBs into HEVC CUs and a statistical model of the HEVC RD model to perform an early termination, obtaining on average a speedup to 3.83× with a rate and distortion penalty of around 4%.

Other approaches available in the literature deal with H.264/AVC to HEVC transcoding for surveillance by considering the long static background characteristics of surveillance videos [23]. Jiang *et al.* [24] proposed a transcoder algorithm based on region feature analysis. The main idea consists in dividing each frame into three regions in terms of CTU on the basis of the correlation between image complexity and the coding bits of the H.264/AVC source bitstream. The results obtained in terms of speedup and BD rate are, on average, 1.93× and 1.73%, respectively.

Finally, there are other related transcoding-based approaches available in the literature that involve the HEVC standard, but they do not focus on H.264/AVC to HEVC. For example, Van *et al.* [25] and De Praeter *et al.* [26] proposed a framework for bitrate adaptation depending on the behavior of the network. Another scheme available in the literature in the framework of HEVC-based transcoding is presented in [26], where a spatially misaligned HEVC transcoding is proposed. Another HEVC-based transcoder scheme is presented in [27], where an MPEG-2 to HEVC transcoder is proposed based on content modeling.

The authors of this paper have already published some work related to this field. In fact, in [28], a multiple frame transcoding algorithm is proposed in which frames used in H.264/AVC as reference for P or B predictions are the only frames checked in the HEVC ME algorithm. And more recently, in [29], a preliminary version of this paper was presented, which was called fast quadtree level decision (FQLD). In this work, an algorithm to determine CU depth based on a Bayesian probabilistic model is proposed, although some improvements and refinements have now been implemented with respect to this paper.

1) A new online threshold procedure to determine whether a CU should be split.
2) The HEVC information is also used in the Bayesian model and in that incoming from H.264/AVC.
3) New models based on each temporal layer and frame type (P or B).

4) A new classification algorithm for a CU size of 16×16 pixels.
5) The proposal has been migrated to the newest version of HM reference software available at this moment.
6) A full performance evaluation has been carried out.

A comparison between the proposed and the previous approach can be seen in Section V-D. Moreover, the authors of this paper would like to emphasize the improvements achieved using a Bayesian model instead of decision trees as they is the most commonly used option in [19]–[21]. These are basically the following.

1) Models based on decision trees are more complex on the training stage (exponential against linear with respect to data, in the worst case).
2) Once new training data are generated, updating the model is not so easy as in the NB model; NB models are linear on the number of data for training and also the number of variables in classification time.
3) Models based on decision rules are not suitable for adapting dynamic thresholds that can be useful to adapt model/probabilities based on video content.

IV. Data-Driven Probabilistic CTU Splitting Algorithm

As mentioned above, the HEVC encoding process can take an immensely long time. But as can be seen in the previous section, there are also some techniques that can reduce the time needed for transcoding to previous standards and, even, to HEVC. Most of these techniques are based on the use of data mining techniques [30]. In particular, mainly decision trees and decision rules have been considered. In this paper, we propose the use of probabilistic models, which are capable of intrinsically dealing with the uncertainty inherent to the targeted problem, so providing more flexible classifiers in order to suit the particularities of each video scene.

As a general description, we can say that our objective is to design models that help the transcoder to make the decision of splitting the CU under study and then descend a level in the quadtree or, on the contrary, to make the decision of stopping and choosing the current level as the maximum allowed depth. Our proposal consists of designing this decision function by following a knowledge discovery from data (KDD) process. Thus, our ultimate goal is to learn a (set of) model(s)

$$\mathcal{M}(f_1, f_2, \ldots, f_n) \longrightarrow \{C_S, C_N\}$$

to be plugged into the transcoder. Therefore, our objective corresponds to the well-known supervised classification task, where we must decide on a set of categories, to split the CU (C_S), and descend a level in the quadtree, or not (C_N), and choose the current level as the maximum allowed depth, using the values of a set of features that describe our current problem instance.

Before entering into the details of the KDD process followed, let us give a general description of the proposed CTU splitting algorithm.

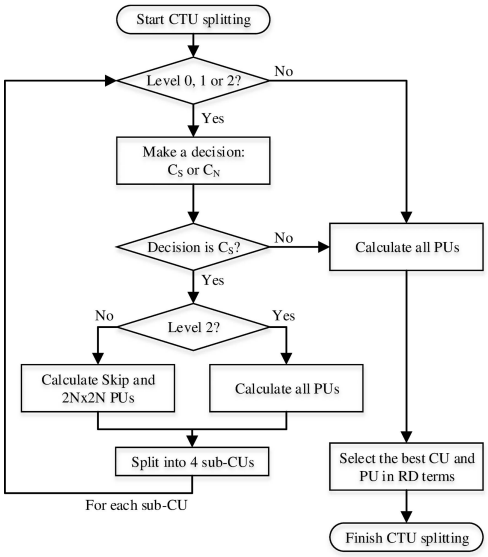


Fig. 2. Diagram of the proposed AFQLD algorithm.

A. General Description of Algorithm

As described above, C_S and C_N are the two categories (or class labels) to be predicted by our decision function or classifier. Furthermore, as the selection can be carried out in a different way depending on the quadtree depth and the type of frame being processed, several classifiers are needed.

If the chosen decision is C_S, we need to take into account some considerations. Thus, if the algorithm obtained a 100% hit rate in all cases, it would be logical to obviate all the PU computations in the levels predicted as C_S. However, this hit rate cannot be obtained in real applications, since each label is associated with a probability of being chosen, so it might occasionally misclassify partitions. With this fact in mind, if C_S is chosen, only skip and $2N \times 2N$ PUs are checked for levels 0 and 1, 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 and all PUs at this CU depth are evaluated and the algorithm for this CTU finishes. Thus, if some of the RD costs that were calculated for higher levels are better than the best RD cost for the final level, the quadtree is allowed to return to the best one among those calculated.

Fig. 2 schematically describes the proposed CU splitting algorithm, which we have called AFQLD, where the term adaptive refers to the fact that the algorithm can be dynamically adapted to the content of each sequence, as will be described below.

From the description of the global behavior of the proposed algorithm, it is clear that we need to design a model $\mathcal{M}_l$ for making the decision at each level $l = 0, 1, 2$. In this study, we rely on a KDD-based approach (see Section IV-B) for levels 0 and 1 of the quadtree (CU sizes of 64×64 and 32×32 pixels, respectively), while at level 2, a much simpler strategy is followed. Basically, as CU size at level 2 is 16×16 pixels, we take advantage of the fact that this

Algorithm 1 Overall Transcoding Process

```
1: Decode H.264/AVC sequence and save each feature F_i
2: Cost[1] = Cost[2] = Cost[3] = Cost[4] = 0
3: for all GOPs in sequence do
4:   for all frames in GOP do
5:     for all CTUs in frame do
6:       if First GOP and cost[currentFrameEnergy]==0
         then
7:         cost[currentFrameEnergy] += SplittingError
8:       else
9:         for all Not finished CUs in CTU do
10:          if Level ∈ {0, 1, 2} then
11:            if Level == 2 then
12:              if MB mode is Skip or 16 × 16 then
13:                Classify as C_N
14:              else if MB mode is 16 × 8 or 8 × 16 then
15:                if Adjacent MB modes are Skip, 16 × 16,
                   16 × 8 or 8 × 16 then
16:                  Classify as C_N
17:                else
18:                  Classify as C_S
19:                end if
20:              else
21:                Classify as C_S
22:              end if
23:            else
24:              Fetch the needed features F_1, F_2, ..., F_n
25:              Classify as C_S or C_N according to the deci-
                 sion of the appropriate model M_l using the
                 fecthed features
26:            end if
27:            if Decision == C_S then
28:              if Level == 2 then
29:                Calculate all PUs
30:              else
31:                Calculate Skip and 2Nx2N PUs
32:              end if
33:              Split into 4 sub-CUs
34:            else
35:              Calculate all PUs
36:              Select the best CU and PU in RD terms and
                 finish this CU
37:            end if
38:          else
39:            Calculate all PUs
40:            Select the best CU and PU in RD terms and
               finish this CU
41:          end if
42:        end for
43:      end if
44:    end for
45:  end for
46: end for
```

is the MB size in H.264/AVC too, so the proposed algorithm mimics H.264/AVC as it can be seen in lines 12–22 of Algorithm 1.

TABLE I
THEORETICAL EFFICIENCY OF THE AFQLD ALGORITHM

	CU size probability				Avg. RDO	RDO op.
	64x64	32x32	16x16	8x8	operations	saving (%)
2560x1600	0.087	0.147	0.402	0.364	1258.58	46.2
1920x1080	0.069	0.137	0.378	0.416	1354.78	42.1
832x480	0.031	0.104	0.342	0.522	1561.73	33.2
416x240	0.019	0.084	0.323	0.574	1660.44	29.0
1280x720	0.221	0.172	0.397	0.211	904.27	61.3

Finally, in order to give an approximation of the expected efficiency for the proposed algorithm, Table I shows the probability of selecting each CU size for the resolutions given in [31] (obtained from sequences that have been fully decoded from H.264/AVC and encoded with HEVC), the number of RDO operations that the proposed algorithm would carry out on average (according to the number of RDO operations carried out at each level), and the percentage of RDO operations that would theoretically be saved.

B. Learning Splitting Model

KDD [32] can be defined as the overall process of finding and interpreting patterns or models in data. The KDD process consists of iterating some of the following steps: creating a target dataset, where we mainly identify the variables related to the targeted problem and select a subset of samples; data cleaning; preprocessing; and data selection, where we devise a subset of variables; and possibly cleaning and transforming them; learning form data, where we have to select the data mining (ML) algorithms and apply them over the selected dataset in order to obtain the models/patterns; model/pattern evaluation/validation; and finally knowledge exploitation.

The problem domain, which is the transcoding problem, has been reviewed in the previous sections. As a rough summary, we can state the following.

1) The task under study can be modeled as a supervised classification problem, where our aim is to predict the correct value for a binary class variable: to split the CU under study (C_S) and not (C_N).
2) Several models must be learnt, depending on the CU depth (0 or 1) and the average energy of the residue, where four levels of energy have been considered (1–4), where 1 represents high residual energy and 4 represents low residual energy. Thus, each frame in RA configuration can be identified with different energy according to its hierarchical layer and frames in low-delay B (LB) and low-delay P (LP) configurations can be associated with levels 3 and 2 of energy, respectively, since LB frames have low energy, but not as low as those with the lowest energy in RA configuration, and LP frames have higher energy than the previous ones.
3) The following families of features can be good predictors to help in decision making.
   a) Features that correctly model the spatial and temporal complexity.
   b) As the framework of this work is a transcoder, information fetched from the decoding stage is available.

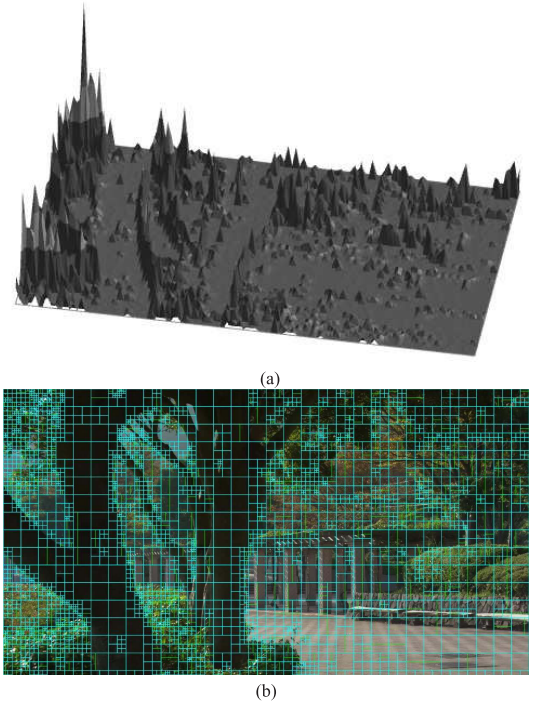


Fig. 3. Visual relationship between CTU splitting in HEVC and the number of bits used to encode the frame in H.264/AVC (QP = 27). (a) Bits per MB in H.264/AVC stream. (b) Original CTU splitting in HEVC.

c) Statistical data, such as the variance of the residue [33], have been shown to work well in previous transcoders.
d) Information that could summarize both the spatial and the temporal information simultaneously.
e) Information from the HEVC coding stage can also be extracted dynamically.

Regarding the variables, we have identified a number of groups of candidate features to be included. This decision comes from our prior knowledge of the problem and also from ad hoc analyses. As an example, the following analysis has been carried out to include the variable w_{Bits} as a representative feature to summarize both spatial and temporal information. The number of bits used to code an MB in the original H.264/AVC stream (w_{Bits}) could also give some information about the Lagrangian cost of each MB in the original sequence. The selection of this feature is based on the study of Fig. 3. In particular, in Fig. 3(a), there is a 3D graphic of the number of bits used to encode each MB from frame 2 in the sequence *ParkScene*, where taller parts of the graphic represent a higher number of bits, while in Fig. 3(b), the CU partitioning after transcoding is shown. It can be seen at a glance that taller parts of the graphic (that is, higher numbers of bits used to encode an MB) correspond to parts of the frame where the CUs are smaller, so there seems to be some kind of correspondence between the number of bits used to encode an MB in H.264/AVC and the CTU partitioning in HEVC. One can even make out the shape of the trees in the frame as hills between plains in the 3D graphic.

This type of analysis has been carried out with different features in order to decide whether to include them or not in our initial dataset. In case of doubt, the mere suspicion that a variable may be correlated with the CTU splitting is enough for us to include it, since it will be discarded later during the data selection process if considered redundant or irrelevant for the task considered.

After this study, each feature was fetched from the H.264/AVC stream or from the HEVC encoding process. In the case of data extracted from the H.264/AVC stream, to determine the correspondence between the different partitions of both standards, for each CU, an overlap of the MBs in the CU covering area was made, e.g., a 64×64 pixel CU corresponds to 16 MBs and a 32×32 pixel CU corresponds to 4 MBs. Thus, our initial set of features, $\mathbf{F}$, contains the following 26 variables:

1) w_{QP}: QP value used to encode the stream;
2) w_{Bits}: number of bits used to encode all the MBs for the current CU after applying the context-adaptive binary arithmetic coding operation;
3) w_{Intra}, w_{Skip}, w_{16}, w_4, and w_{Inter}: number of intra, skip, inter 16×16, inter 4×4 and other inter size MBs, respectively;
4) w_{DCTno0}: number of nonzero DCT coefficients;
5) w_{Width} and w_{Height}: frame width and height, respectively;
6) w_{MVsum}: sum of all the MV components contained in the frame;
7) w_{ResAvg} and w_{ResVar}: average and variance of the residue for the covered area, respectively;
8) $w_{\text{ResAvgSubCU1}}$, $w_{\text{ResAvgSubCU2}}$, $w_{\text{ResAvgSubCU3}}$, and $w_{\text{ResAvgSubCU4}}$: average of the residue for each sub-CU: 1, 2, 3 and 4, respectively;
9) $w_{\text{ResVarSubCUs}}$: variance of the above four values;
10) w_{SobelH} and w_{SobelV}: sum of applying the Sobel operator [34] to the residue in horizontal and vertical directions;
11) w_{MVxAvg}, w_{MVyAvg}, w_{MVxVar}, and w_{MVyVar}: average and variance of x and y MV components, respectively, for the covered area;
12) w_{SkipCost} and $w_{2N\times 2N\text{Cost}}$: Lagrangian cost of choosing skip and $2N \times 2N$, respectively, at the HEVC coding stage.

Regarding the samples, in order to create an adequate data set to train the models, five sequences have been selected from those described by the JCT-VC in [31]. The criterion to choose them has been the use of the spatial index (SI) and temporal index (TI) according to the ITU-T P.910 recommendation [35], so that the chosen sequences cover a wide range of the peculiarities of an image. It is useful to compare the relative SI and TI of the test sequences, since the compression difficulty is usually directly related to both spatial and temporal complexity.

In particular, the *PeopleOnStreet, ParkScene, PartyScene*, and *BQSquare* sequences have been selected. These represent one sequence per class (A, B, C, and D), so that they

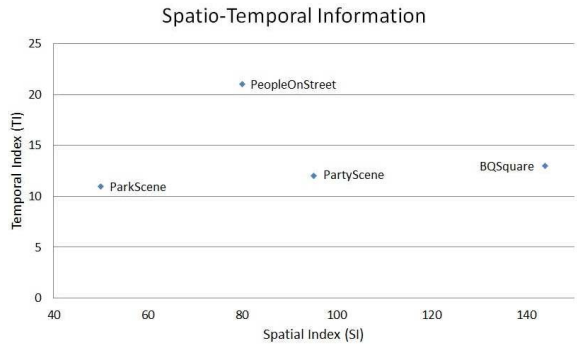


Fig. 4. SI and TI of the selected sequences.

can also be representative of the wide range of resolutions (see Section V-A for more details). Fig. 4 shows the SI and the TI of each sequence. As can be observed, they cover a wide range of peculiarities. The first 1000 CUs of each combination of sequence, QP, CU depth, and temporal layer (with RA configuration and QP values $\{22, 27, 32, 37\}$) after skipping the first frame (since it is coded as intra), were selected as input for the supervised learning process, which results in eight classifiers and a total of 16 000 to learn each of them.

Therefore, the initial dataset can be viewed as a $16\,000\times27$ bidimensional matrix, which corresponds to 16 000 instances in which a decision about splitting a CU or not is needed. For each one of these cases, we know the correct decision (C_N or C_S) and the value of the 26 predictive features. Now, our goal is to learn classifiers that are able to make the decision on unseen new cases.

Specifically, a different dataset has been built for each CU depth (0 and 1) and each level of residual energy (1, 2, 3, and 4): $\mathrm{DB}_{i,j}, i \in \{0, 1\}, j \in \{1, 2, 3, 4\}$. For the training, RA configuration has been used, since it contains frames from all levels of residual energy (according to its hierarchical GOP structure). Therefore, this process involves eight different datasets with a total of 128 000 instances.

In the field of the data mining and data analysis, there are a large number of tools devoted to deal and analyze data sets. One of the most popular is Weka [30], which is a tool written in java and it implements the most popular data mining algorithms. Moreover, Weka is open-source software issued under the GNU General Public License. Because of this, we have selected this tool to conduct the data mining tasks in our problem.

C. Preprocessing Data: Discretization

Most of the previous selected features are of numerical nature. Although the selected probabilistic classifier (NB) and the ML algorithm to train it are able to cope with numerical variables, they impose an assumption that the values for a feature projected for each class value follow a univariate Gaussian distribution, which is quite improbable in practice. Because of this, we decided to preprocess the feature by discretizing them.

From the existing range of available discretization techniques, we have selected the entropy-based supervised discretization algorithm proposed by Fayyad and Irani [36]. This algorithm fits well with our task because its supervised nature allows splitting the numerical variable into a set of intervals that yields more discriminative power regarding the class variable. Two well-known advantages of this algorithm are: 1) resulting intervals are of different widths and 2) the number of intervals for each feature is automatically selected by the algorithm (see [36] for details). Once we have the data sets collected in the previous step, they are used as input for the Weka package. In order to apply the Fayyad and Irani discretization algorithm, the supervised.Discretize filter has to be selected in Weka (default parameter setting leads to canonical Fayyad and Irani algorithm [36]).

The output of the discretization process is a new dataset in which all the variables are discrete/categorical, heterogeneously defined regarding the number of intervals and their widths. It should be noted that some of the features ($w_{\mathrm{QP}}, w_{\mathrm{bits}}, w_{\mathrm{width}}, \ldots$) have numerical nature, but they already have discrete values (they belong to natural numbers); in this case, this process is still valid and it splits $\mathbb{N}$ into intervals. An example of this process is shown in Section IV-G.

Using this discretization strategy, all the learnt models have increased their performance. As an example, the accuracy of the NB model learnt increases about 10% for $\mathrm{DB}_{0,2}$ with respect to the use of numerical variables.

D. Feature Subset Selection

Probabilistic classifiers in general, and NB in particular, are quite sensitive to the feature set used to induce the classifier. Thus, the presence in the training set of irrelevant (uncorrelated with the class) variables and/or redundant [correlated with other feature(s)] may significantly affect the precision of the learned classifier. Furthermore, different tasks may require different subsets of features, and this is our case as we need to learn different classifiers (levels and frames).

To cope with the aforementioned problems a feature subset selection (FSS) is applied in order to select the proper subset of features for each particular task [37]. We selected a greedy strategy in combination with wrapper evaluation, as it is computationally efficient and robust against overfitting. The forward selection process starts with the empty set and iteratively incorporates the best remaining feature at each step. In the wrapper approach, the best feature is decided to be the one that, when joined to the current subset of selected features, allows the ML algorithm to induce the classifier with the maximum accuracy among those trained in that iteration. The FSS process finishes when the addition of a new feature no longer improves the accuracy of the induced classifier. The advantage of using a wrapper technique is that the same ML algorithm to be used for model discovering can be used at this step, thus maximizing the knowledge transferred between these two KDD steps.

To cope with this step, we have used again the Weka data mining suite. This time we have used the dataset discretized in the above step as the input data. To do FSS described in the last paragraph in Weka, the filter supervised.AtributeSelection is used. As an attribute subset evaluator, we have used

WrapperSubsetEval coupled with the NB as subrogate classifier. As a search method, we have selected GreedySteepwise instantiated to a sequential forward search.

Following the example for the database $DB_{0,2}$, once the variable selection is carried out, the model accuracy achieves 92%. Note that the initial accuracy, with continuous variables, was about 70%. In all the learnt models, the cardinality of the selected subset has been in the range 2–5, with an average of 3.6. That means an average reduction of 86%.

E. Mining the Data

After the previous steps, we have a higher quality dataset than the one initially selected, and from better data, we will be capable of *mining* better models. Let us first briefly introduce the ML algorithm selected in this paper and then we will outline the data mining process carried out.

1) Naïve–Bayes Classifier: The NB classifier [38], [39] is perhaps the simplest representative of probabilistic classifiers. Despite its simplicity, it has been successfully used in many domains and continues to be one of the most commonly used algorithms in the field of machine learning and data mining. This simplicity is the main reason for its choice in the transcoding algorithm.

NB relies on a strong independence assumption: all features are conditionally independent given the class. This can be a considerable drawback in theory, but in practice, NB shows a competitive performance with respect to more complex ML algorithms. Furthermore, this independence assumption brings the following important advantages: 1) NB is memory and space efficient at both learning and classification; 2) NB has the ability of learning good models even from scarce data; and 3) NB is robust against overfitting, thanks to the small number (and low-order) of parameters (conditional probability tables), it needs to learn.

As a probabilistic classifier, NB inherently deals with the uncertainty present in the data, behaving in this way as a flexible classifier. The classification rule is based on the maximum *a posteriori* probability principle, that is, given the values of the input features for an object $x = (f_1, \ldots, f_n)$, it computes the posterior probability distribution for the class variable C and chooses the class label $\{c_1, \ldots, c_K\}$ with the highest probability

$$c^* = \underset{C \in \{c_1, \ldots, c_K\}}{\operatorname{argmax}} P(C | f_1, \ldots, f_n). \tag{1}$$

Applying Bayes' Theorem and realizing that the denominator is the same for all class labels, we get

$$\begin{aligned} c^* &= \underset{C \in \{c_1, \ldots, c_K\}}{\operatorname{argmax}} \frac{P(f_1, \ldots, f_n | C) P(C)}{P(f_1, \ldots, f_n)} \\ &= \underset{C \in \{c_1, \ldots, c_K\}}{\operatorname{argmax}} P(f_1, \ldots, f_n | C) P(C). \end{aligned}$$

Obviously, managing the joint probability distribution of all the variables involved $P(f_1, \ldots, f_n, C)$ becomes unfeasible even for small datasets (few variables), not only because of the number of parameters needed (exponential in the number of variables) but also because of the amount of data required for the distribution to be representative. However, thanks to the conditional independence assumption considered in the NB model, the joint distribution is approximated by the following factorization:

$$P(f_1, \ldots, f_n, C) = P(C) \prod_{i=1}^{n} P(f_i | C)$$

and so the NB classifier can be formulated as

$$c^* = \underset{C \in \{c_1, \ldots, c_K\}}{\operatorname{argmax}} P(C) \prod_{i=1}^{n} P(f_i | C).$$

Thus, the algorithm to induce the NB model only needs to calculate the marginal distribution for C

$$P(C = c_k) = \#(C = c_k)/N \tag{2}$$

where $\#(C = c_k)$ is the frequency that C is equal to the value c_k in data and N is the total amount of data, and the probability distribution of each feature conditioned to the class $P(f_i|C)$. If f_i is a discrete variable, as it is our case, its conditional probability is computed by counting relative frequencies. This set of frequencies is computed in only one reading of the data, actually the computational complexity in learning and NB is one of the fastest algorithms, which takes $\mathcal{O}(Nn)$, being N the number of instances and n the number of features in database.[1] In addition, NB is linear in classification phase and it takes $\mathcal{O}(n)$.

2) Learning the Models: Starting from the original set of features and the eight datasets, one for each combination of CU depth level (0 or 1) and residual energy level (1, 2, 3, or 4), eight independent data mining processes are carried out to learn the corresponding models ($\mathcal{M}_{ij}, i \in \{0, 1\}$, $j \in \{1, 2, 3, 4\}$). Each model corresponds to the NB classifier induced by taking as training set the dataset obtained by applying discretization followed by FSS for the corresponding initial dataset. Once an NB classifier has been learned using the Weka package, the corresponding model has been implemented, which means that the Weka package is not used during the transcoding process.

F. Model/Pattern Evaluation

When evaluating the learnt models using standard scores such as cross-validated *accuracy* or the area under the receiving operator characteristics (ROC),[2] curve apart from discovering that discretization and FSS improves the models both in terms of simplicity and accuracy, we also realized that our models should be calibrated taking into account the specific problem we are tacking. By calibration, we mean the problem of setting the specific classifier rule. From (1) and taking into account that our class only has two labels, it is easy to see that the induced classification rule is to choose C_N if $P(C_N) > P(C_S)$. Semantically, this fact means that both types of mistake, namely, classifying as C_N when the actual value is C_S or classifying as C_S when the actual value is C_N, have

[1] For example, the models used in related works, such as linear discriminant functions and decision trees, take $\mathcal{O}(Nn^2)$ and $\mathcal{O}(nN \log N) + \mathcal{O}(N(\log N)^2)$, respectively [40]. Moreover, probabilistic algorithms are easily adaptable to each specific instance, as described in Section IV-F.

[2] ROC.

the same *cost*. However, we have detected that this is not the case in the transcoding problem. First, intuitively and generally speaking, the cost of making an error of not splitting when HEVC decides to split should be more costly because if we decide to split, speed is decreased but the quality of the image is preserved. And second, the costs should be dependent not only on the model but also on the video sequence. Therefore, in order to achieve a good performance for the proposed algorithm, the costs of making every decision are dynamically adjusted.

There are four different costs in the classifying process: the cost of correctly choosing C_N (Cost_{NN}), the cost of correctly choosing C_S (Cost_{SS}), the cost of choosing C_N when the correct decision would have been C_S (Cost_{NS}), and the cost of choosing C_S when the correct decision would have been C_N (Cost_{SN}). Once we have estimated these costs, the decision rule should be $P(C_S) \times \text{Cost}_{\text{SN}} > P(C_N) \times \text{Cost}_{\text{NS}}$.

In order to measure this costs given the correct decision, the Lagrangian cost of splitting (L_S) and of not splitting (L_N) have been used, as well as the mathematical concept of absolute error, as shown in (3), where Cost_{ij} represents the cost of choosing C_i when the correct decision is C_j, and L_j is the Lagrangian cost of the correct decision, L_i is the Lagrangian cost of the prediction made by the classifier, where i, $j \in \{S, N\}$

$$\text{Cost}_{ij} = |L_j - L_i| \times \omega_{ij} \tag{3}$$

where ωij is a weight associated for each particular cost. The weight associated with the Cost_{NS} is 2.0 and 1.0 for the rest.[3] It should be noted that $\text{Cost}_{\text{NN}} = \text{Cost}_{\text{SS}} = 0$ as expected, since the cost of not making a mistake must be 0. On the other hand, all the costs can be normalized by dividing them by Cost_{NS} (Cost_{SN} could also have been chosen), so the normalized costs are the following: $\overline{\text{Cost}_{\text{NS}}} = 1$, $\overline{\text{Cost}_{\text{SN}}} = \text{Cost}_{\text{SN}}/\text{Cost}_{\text{NS}}$. Taking this fact into account, the rule will depend only on one value: Cost_{SN} normalized by Cost_{NS}, since all the others are constants. Therefore, the decision rule is transformed into $P(C_S) \times \overline{\text{Cost}_{\text{SN}}} > P(C_N)$.

As mentioned above, these costs are estimated for each sequence and each particular model (they are only learned at the beginning of the sequence; however, they could be learned again every n frames or when a scene change is detected, but are out of the scope of this paper). For the calculation process, the first frame of the current residual energy level is encoded by running the algorithm but letting the HEVC encoder make the correct decision so that it can calculate the actual and the predicted decision.[4] If Cost_{SN} or Cost_{NS} are 0, then the following frame is used to calculate the threshold, and so on (setting the process not to exceed the first GOP). In this way, the time spent to encode a few frames (with the maximum of a GOP) without the AFQLD algorithm is negligible compared with the whole encoding time, while letting us have a dynamic sequence content and QP-dependent cost and threshold.

[3]Note that this choice was totally heuristic. We have empirically tested several weights for $C_{\text{NS}} \in \{1, 2, 3, 4, 5\}$, and the best results were obtained with 2.0. Also note that this fact is in agreement with our intuition so as to preserve the quality of the image.

[4]Therefore, this adaptive scheme is carried out online in transcodification time by using the models implemented in the transcodification software.

TABLE II
MODEL OF w_{Bits} FEATURE IN $M_{1,4}$ CLASSIFIER

	$\#(w_{Bits} \in \cdot \vert C_S)$	$\#(w_{Bits} \in \cdot \vert C_N)$
$[0, 0]$	2214	2029
$(0, 46]$	3018	4382
$(46, 108]$	1051	976
$(108, 404]$	1207	617
$(404, +\infty)$	423	93
Total	7913	8097

G. Model/Pattern Interpretation and Knowledge Exploitation

The last stage of the KDD cycle is devoted to analysis/interpretation of the models learned and their exploitation in the final system. Here, we briefly describe one of the four discovered models, while the performance evaluation of the system integrating the discovered models deserves, in our opinion, a whole section (Section V).

We will now comment on some parts of the classifier learnt for residual energy level 4 and level 1 of the quadtree ($\mathcal{M}_{1,4}$). The final set of features, after discretizing and the selection of features subset, is the following:

1) w_{Bits}: $[0, 0]$, $(0, 46]$, $(46, 108]$, $(108, 404]$, $(404, +\infty)$;
2) w_{MVsum}: $[0, 14131]$, $(14131, 15064]$, $(15064, 16529]$, $(16529, 64264]$, $(64264, 2110792]$, $(2110792, +\infty)$;
3) w_{MVyVar}: $[0, 6.248047]$, $(6.248047, +\infty)$;
4) w_{SkipCost}: $[0, 3337]$, $(3337, 25743]$, $(25743, 274594]$, $(274594, +\infty)$.

It can be seen, as expected, that the w_{Bits} feature was selected. To give an example of the information which the classifier must store, Table II shows the number of instances for each interval of the w_{Bits} feature. The probability can be obtained by dividing each entry in Table II by the sum of all the elements in its column, i.e., $P(w_{\text{Bits}} \in (0, 46] | C_S) = 3018/7913 \approx 0.38$. The tables for the remaining variables are similar to Table II, but they are not included here due to space limitations.

H. Overall Transcoding Process

The overall transcoding process is shown in Algorithm 1. First, the H.264/AVC sequence is decoded (and all the needed information is generated) and the costs described in Section IV-F are set up to 0 for each level of energy (from 1 to 4). Then the HEVC encoding process is started, encoding each GOP using the AFQLD algorithm (Fig. 2) to encode the CTUs in the frames. It can be noted that the first GOP is used to learn the costs.

V. PERFORMANCE EVALUATION

This section aims to evaluate the AFQLD algorithm presented in this paper by showing its performance with different configurations and applying it to different depths of the quadtree. A comparison with another relevant state-of-the-art proposal and with other non-transcoding-specific optimizations will also be presented at the end of this section.

2.1. Adaptive Fast Quadtree Level Decision Algorithm for H.264 to HEVC Video Transcoding

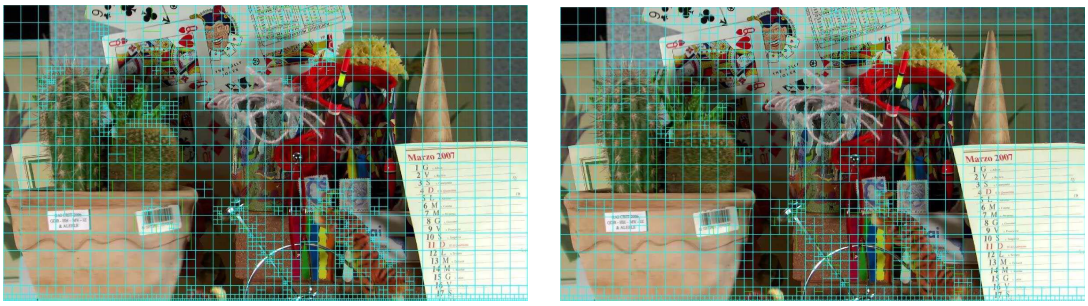


Fig. 5. CU splitting for the original and proposed algorithms. The fourth frame of the *Cactus* sequence. RA Configuration. Original splitting in the left column and splitting with the proposed algorithm in the right column.

A. Simulation Setup and Metrics

In order to ensure a common framework, the JCT-VC defined a document [31], where test conditions are set out to homogenize comparisons between experiments. Therefore, this performance evaluation has been carried out in accordance with these guidelines. Specifically, the QP values that were used are $\{22, 27, 32, 37\}$ and the configurations are RA main 10, LB main 10, and LP main 10.

The results for each sequence and the global average value are shown. Moreover, the sequences are grouped into *Classes* (A, B, C, D, and E), as described in [31], according to their resolution. It should also be noted that (according to [31]) *Class A* cannot be used with LB or LP configurations and *Class E* (videoconferencing sequences) cannot be used with the RA configuration:

1) *Class A (*2560 × 1600 *Pixels): Traffic*, *PeopleOnStreet*, *NebutaFestival*, and *SteamLocomotiveTrain* sequences;
2) *Class B (*1920 × 1800 *Pixels): Kimono*, *ParkScene*, *Cactus*, *BQTerrace*, and *BasketballDrive* sequences;
3) *Class C (*832 × 480 *Pixels): RaceHorsesC*, *BQMall*, *PartyScene*, and *BasketballDrill* sequences;
4) *Class D (*416 × 240 *Pixels): RaceHorses*, *BQSquare*, *BlowingBubbles*, and *BasketballPass* sequences;
5) *Class E (*1280 × 720 *Pixels): FourPeople*, *Johnny*, and *KristenAndSara* sequences.

The software used is JM 18.4 [41] for H.264/AVC and HM 16.2 [13] for HEVC. The remaining coding parameters not mentioned here are kept as default in the configuration file. Thus, the process to generate these results is the following:

1) Encode the YUV file with H.264/AVC reference software using HM-like configuration files that are included in the reference software JM 18.4 [41].
2) Decode each file with the decoder side of the proposed transcoder, producing a YUV' file as well as all the information needed for the proposed transcoding algorithm.
3) Encode the YUV' file with the encoder side of the original transcoder (anchor).
4) Encode the YUV' file with the encoder side of the proposed transcoder for each combination of the proposed algorithms (proposed).
5) Compare the anchor with each proposed stream in order to obtain the BD rate and the speedup.

Measurements have been performed on a six-core Intel Core i7-3930K CPU running at 3.20 GHz. The results are presented in terms of speedup and BD rate. The time reduction is also used when describing the results, which is an equivalent metric to the speedup. The speedup and time reduction are calculated as indicated in (4) and (5), respectively, where t_{anchor} is the execution time of the encoder side of the nonaccelerated reference transcoder and t_{proposed} is the execution time of the encoder side of the proposed transcoder. The YUV BD rate is the weighted average of the Y, U and V components as shown in (6) (given that the Y component is four times larger than the U and V components, since the video format is 4:2:0)

$$\text{Speed-up} = \frac{t_{\text{anchor}}}{t_{\text{proposed}}} \tag{4}$$

$$\text{Time reduction}\,(\%) = \frac{t_{\text{anchor}} - t_{\text{proposed}}}{t_{\text{anchor}}} \times 100 \tag{5}$$

$$\text{BD rate}\,(\%) = \frac{4 \times \text{BD rate}_Y + \text{BD rate}_U + \text{BD rate}_V}{6}. \tag{6}$$

B. Results and Analysis

As an initial visual evaluation, Fig. 5 shows the differences between the original CU splitting that is performed by the anchor transcoder (left column), and the CU splitting that is performed by the proposed transcoder, which makes use of the proposed AFQLD algorithm (right column). It can be seen that both CU splittings are quite similar, which demonstrates the accuracy of the classifying algorithm.

Tables III–V contain the results for RA, LB, and LP configurations, respectively. Tables III–V show the difference made by applying the AFQLD algorithm to incremental depth levels of the CU so that the evolution of the speedup and BD rate can be appreciated. Thus, it can be seen that the more levels the algorithm is applied to, the greater the speedup and the BD rate are. Moreover, the results show that Level 0 can achieve a moderate speedup without a significant loss. Consequently, the quality complexity could be adjusted by the user deciding whether to apply the AFQLD algorithm to one, two, or three levels.

It should be noted that the best performance of the proposed algorithm is not achieved for the sequences that the training frames came from. On the contrary, we have achieved the

TABLE III
SPEEDUP AND CODING EFFICIENCY RESULTS OF THE AFQLD ALGORITHM WITH RA CONFIGURATION

		BD-rate (%)												Speed-up		
		Level 0				Level 1				Level 2				L0	L1	L2
		Y	U	V	*YUV*	Y	U	V	*YUV*	Y	U	V	*YUV*			
Class A	Traffic	1.1	0.3	0.1	*0.8*	3.4	1.1	0.8	*2.6*	4.2	0.6	0.3	*3.0*	1.52	2.09	2.71
	PeopleOnStreet	0.1	-0.1	-0.1	*0.0*	0.7	0.6	0.5	*0.6*	2.0	0.7	0.6	*1.6*	1.12	1.33	1.73
	NebutaFestival	1.2	0.4	0.3	*0.9*	4.1	1.3	2.0	*3.3*	5.7	-2.0	-1.5	*3.2*	1.12	1.49	2.03
	SteamLocomotive	1.5	-0.2	-0.5	*0.9*	3.2	0.5	1.3	*2.2*	3.2	0.0	0.3	*2.2*	1.63	2.09	2.37
Class B	Kimono	0.9	0.5	-0.1	*0.7*	2.8	1.7	1.1	*2.4*	3.1	1.2	0.4	*2.3*	1.38	1.88	2.42
	ParkScene	1.2	-0.1	-0.1	*0.8*	4.3	1.1	0.5	*3.2*	4.7	0.5	0.1	*3.3*	1.45	1.95	2.52
	Cactus	0.7	0.3	0.3	*0.6*	3.1	1.6	1.5	*2.6*	3.8	1.1	1.4	*2.9*	1.42	1.90	2.39
	BasketballDrive	0.8	0.1	0.4	*0.6*	5.2	2.5	3.4	*4.4*	5.8	2.2	3.2	*4.7*	1.35	1.87	2.26
	BQTerrace	1.8	0.0	0.0	*1.2*	5.8	0.6	0.9	*4.1*	6.9	-1.3	0.7	*4.3*	1.59	2.09	2.74
Class C	BasketballDrill	0.2	0.3	0.3	*0.2*	3.1	2.6	3.0	*3.0*	3.9	3.0	3.3	*3.7*	1.25	1.62	2.10
	BQMall	0.5	0.3	0.4	*0.5*	3.0	1.2	1.8	*2.5*	3.7	1.2	1.7	*2.9*	1.27	1.69	2.22
	PartyScene	0.2	-0.3	0.1	*0.1*	1.4	0.5	0.6	*1.1*	1.9	0.1	0.2	*1.3*	1.12	1.47	1.96
	RaceHorsesC	0.1	-0.3	-0.1	*0.0*	1.3	0.7	0.7	*1.1*	1.5	0.9	1.0	*1.3*	1.11	1.45	1.46
Class D	BasketballPass	0.2	-0.2	0.4	*0.2*	4.5	1.6	3.8	*3.9*	5.1	1.8	4.6	*4.5*	1.12	1.51	1.86
	BQSquare	0.7	-0.3	-0.2	*0.4*	2.8	-0.4	-0.3	*1.7*	3.2	-1.2	-0.6	*1.8*	1.18	1.71	2.27
	BlowingBubbles	0.2	0.2	-0.1	*0.2*	1.5	0.1	0.3	*1.1*	2.1	-0.2	0.0	*1.4*	1.13	1.45	1.96
	RaceHorses	0.1	0.0	0.2	*0.1*	0.5	0.5	0.5	*0.5*	1.6	0.6	0.8	*1.3*	1.06	1.25	1.66
Average		0.7	0.1	0.1	*0.5*	3.0	1.1	1.1	*2.4*	3.7	0.5	0.9	*2.7*	1.28	1.70	2.16

TABLE IV
SPEEDUP AND CODING EFFICIENCY RESULTS OF THE AFQLD ALGORITHM WITH LB CONFIGURATION

		BD-rate (%)												Speed-up		
		Level 0				Level 1				Level 2				L0	L1	L2
		Y	U	V	*YUV*	Y	U	V	*YUV*	Y	U	V	*YUV*			
Class B	Kimono	1.0	0.1	0.6	*0.8*	3.8	0.9	1.2	*2.9*	4.3	0.7	-0.2	*2.9*	1.31	1.88	2.57
	ParkScene	1.6	-0.8	-1.7	*0.7*	7.6	-2.7	-2.9	*4.1*	8.1	-4.1	-3.9	*4.1*	1.35	1.86	2.50
	Cactus	1.7	0.0	0.1	*1.2*	5.3	2.9	2.1	*4.3*	5.9	1.3	1.5	*4.4*	1.38	1.86	2.44
	BasketballDrive	0.9	0.5	0.4	*0.8*	4.7	2.7	2.4	*4.0*	5.4	2.0	2.3	*4.3*	1.28	1.76	2.27
	BQTerrace	6.1	-3.1	-2.2	*3.2*	15.0	-2.2	-1.5	*9.4*	15.9	-8.2	-4.8	*8.4*	1.55	2.08	2.75
Class C	BasketballDrill	0.9	1.0	0.6	*0.8*	2.0	2.5	2.2	*2.1*	2.1	1.6	1.7	*2.0*	1.28	1.65	1.94
	BQMall	0.6	0.8	-1.0	*0.4*	1.8	2.2	0.6	*1.7*	2.2	1.2	0.0	*1.7*	1.26	1.59	1.99
	PartyScene	0.2	0.0	-0.9	*0.0*	0.8	0.9	1.5	*0.9*	1.3	-0.4	-0.3	*0.8*	1.15	1.42	1.83
	RaceHorsesC	0.1	-0.7	-0.3	*0.0*	0.8	-0.2	0.6	*0.6*	1.6	-0.5	0.7	*1.1*	1.16	1.35	1.74
Class D	BasketballPass	0.2	0.2	0.0	*0.2*	1.7	2.0	1.2	*1.7*	3.1	2.5	3.0	*3.0*	1.14	1.51	1.89
	BQSquare	0.3	-0.5	0.6	*0.2*	2.3	-1.5	0.9	*1.4*	5.0	-0.9	1.1	*3.4*	1.23	1.61	2.19
	BlowingBubbles	0.1	0.0	-0.8	*-0.1*	0.6	0.1	0.8	*0.6*	2.1	0.1	1.1	*1.6*	1.13	1.35	1.87
	RaceHorses	0.1	0.4	-0.3	*0.1*	0.6	1.2	1.0	*0.8*	2.2	2.0	2.6	*2.2*	1.08	1.30	1.68
Class E	FourPeople	3.9	0.3	-0.3	*2.6*	8.2	2.7	2.5	*6.3*	8.7	2.8	2.6	*6.7*	2.49	3.36	3.95
	Johnny	6.1	1.3	0.8	*4.4*	11.2	4.0	3.2	*8.7*	11.2	4.0	3.7	*8.8*	2.93	4.35	4.64
	KristenAndSara	6.9	2.2	2.6	*5.4*	11.8	5.4	5.6	*9.7*	11.9	5.1	4.9	*9.6*	2.54	3.51	4.07
Average		1.9	0.1	-0.1	*1.3*	4.9	1.3	1.3	*3.7*	5.7	0.6	1.0	*4.1*	1.52	2.03	2.52

best performance with other sequences that do not belong to the training set. This confirms that the selection of training sequences meets the requirement of being a representative sample of the common video sequences.

It can be observed that for Depth Level 0, most of the sequences achieve very good results in terms of BD-rate penalty (lower than 1% on average), with an average speedup of around 1.4× (with a time reduction of 28.6%), but some low complexity sequences in terms of SI even achieve a bigger time reduction. This proves that by applying a good classifier for just the largest CU sizes (64×64), we can achieve a moderate time reduction with a negligible rate penalty.

The proposed algorithm implementation for Depth Level 1 (i.e., Level 0 + Level 1) achieves better complexity reduction, with an average speedup of around 1.8× (Time Reduction of 45.1%) and a BD-rate penalty of around 3.1% on average. The full implementation of the algorithm for Depth Level 2 (i.e., Level 0 + Level 1 + Level 2) achieves a quantitative speedup of around 2.3× on average (with a time reduction of 56.6%) and a BD-rate penalty of around 3.4%.

In a more detailed analysis of Tables III-V, one can see some interesting results, such as obtaining negative BD rates (i.e., Level 0) or obtaining lower BD rates when adding one more level of depth (i.e., *BQTerrace* sequence when using the LB configuration). This happens because both ME and AFQLD algorithms are optimized for the luminance component, but not for chrominance. In fact, the BD rate of the luminance is worse when adding depth levels to the algorithm, but the BD rate of the chrominances obtains better results and, after weighting the components, the global BD rate is better. It can also be seen that the LB configuration obtains a higher speedup than RA, on average. However, this is due to the fact that Class E obtains very high speedups (as it contains videoconferencing sequences, which are easy to predict due

TABLE V
SPEEDUP AND CODING EFFICIENCY RESULTS OF THE AFQLD ALGORITHM WITH LP CONFIGURATION

		BD-rate (%)												Speed-up		
		Level 0				Level 1				Level 2				L0	L1	L2
		Y	U	V	*YUV*	Y	U	V	*YUV*	Y	U	V	*YUV*			
Class B	Kimono	0.4	-0.1	0.8	*0.4*	2.6	1.8	3.0	*2.5*	3.0	1.3	1.6	*2.5*	1.21	1.64	2.30
	ParkScene	0.4	1.2	0.1	*0.5*	5.7	2.2	2.4	*4.6*	6.4	0.4	0.2	*4.3*	1.24	1.74	2.29
	Cactus	0.3	-0.2	0.2	*0.2*	2.8	3.7	1.6	*2.8*	4.2	0.1	0.5	*2.9*	1.27	1.53	2.02
	BasketballDrive	0.1	0.3	0.2	*0.2*	3.6	3.1	2.7	*3.4*	4.5	2.4	2.4	*3.8*	1.16	1.58	2.00
	BQTerrace	0.4	0.8	-1.4	*0.2*	9.5	2.2	3.3	*7.3*	10.6	-3.5	-0.9	*6.3*	1.31	1.81	2.28
Class C	BasketballDrill	0.0	-0.1	0.1	*0.0*	1.7	1.5	2.0	*1.7*	3.3	2.0	3.1	*3.0*	1.16	1.45	1.91
	BQMall	0.2	-0.5	-0.2	*0.0*	1.1	0.9	1.0	*1.1*	2.4	0.0	-0.2	*1.6*	1.13	1.33	1.87
	PartyScene	0.0	0.8	0.1	*0.2*	0.5	2.0	0.9	*0.8*	1.5	0.8	-0.7	*1.0*	1.08	1.26	1.67
	RaceHorsesC	0.0	-0.1	-0.6	*-0.1*	0.6	0.9	0.4	*0.6*	1.3	1.1	0.8	*1.2*	1.08	1.26	1.59
Class D	BasketballPass	0.0	0.0	-0.2	*0.0*	1.3	0.5	1.0	*1.1*	2.2	1.4	1.6	*2.0*	1.09	1.33	1.65
	BQSquare	0.1	0.6	-2.1	*-0.2*	2.3	-1.9	-0.8	*1.1*	3.0	-0.4	-3.2	*1.4*	1.08	1.43	1.79
	BlowingBubbles	0.0	0.0	1.2	*0.2*	0.3	1.3	2.3	*0.8*	1.5	-0.3	0.2	*1.0*	1.06	1.22	1.64
	RaceHorses	0.1	0.1	0.0	*0.1*	0.5	0.3	0.6	*0.5*	1.3	0.7	0.8	*1.1*	1.06	1.23	1.51
Class E	FourPeople	1.7	-0.9	-0.6	*0.9*	7.2	2.3	2.9	*5.6*	7.5	1.8	2.0	*5.6*	1.94	2.57	3.36
	Johnny	5.6	2.2	1.9	*4.4*	13.4	4.7	3.3	*10.2*	11.9	3.9	2.0	*8.9*	1.64	3.50	4.30
	KristenAndSara	3.7	2.2	2.0	*3.2*	9.4	5.4	5.1	*8.0*	10.0	6.1	5.2	*8.6*	2.13	2.79	3.66
Average		0.8	0.4	0.1	*0.6*	3.9	1.9	2.0	*3.3*	4.7	1.1	1.0	*3.5*	1.29	1.73	2.24

TABLE VI
ACTUAL HIT RATE FOR EACH LEVEL OF THE PROPOSED AFQLD ALGORITHM

		Hit rates for RA configuration (%)			Hit rates for LB configuration (%)			Hit rates for LP configuration (%)		
		Level 0	Level 1	Level 2	Level 0	Level 1	Level 2	Level 0	Level 1	Level 2
Class A	Traffic	90.93	83.34	92.75	NA	NA	NA	NA	NA	NA
	PeopleOnStreet	96.12	88.02	88.67	NA	NA	NA	NA	NA	NA
	NebutaFestival	91.79	80.88	89.23	NA	NA	NA	NA	NA	NA
	SteamLocomotive	86.73	85.28	93.87	NA	NA	NA	NA	NA	NA
Class B	Kimono	89.26	83.84	95.78	85.34	79.73	94.09	87.93	78.53	93.73
	ParkScene	90.81	87.01	93.96	88.57	79.06	90.31	90.86	78.12	88.46
	Cactus	93.38	85.85	94.01	91.58	80.69	90.32	92.86	82.76	86.50
	BQTerrace	91.75	87.79	94.53	87.51	82.49	91.50	90.34	81.89	87.54
	BasketballDrive	89.77	83.14	92.85	86.41	79.52	89.71	91.08	84.09	92.59
Class C	RaceHorsesC	96.44	90.17	91.87	94.55	85.04	86.63	94.86	85.25	82.85
	BQMall	93.09	86.18	90.58	91.71	83.33	86.40	93.24	83.47	82.80
	PartyScene	94.48	88.64	91.39	93.61	84.21	87.29	95.25	86.09	82.59
	BasketballDrill	94.54	84.82	90.22	91.20	82.92	85.90	92.81	82.96	84.46
Class D	RaceHorses	97.13	91.32	91.86	97.68	89.44	84.70	98.08	90.30	84.56
	BQSquare	92.86	91.76	92.85	91.96	86.86	85.65	96.01	89.68	83.64
	BlowingBubbles	96.83	93.14	93.50	94.60	88.06	83.52	96.28	90.77	82.53
	BasketballPass	95.94	89.83	92.06	95.21	88.79	85.45	96.65	90.76	85.79
Class E	FourPeople	NA	NA	NA	91.25	81.47	92.43	94.89	82.63	90.48
	Johnny	NA	NA	NA	91.67	89.51	94.46	93.43	83.62	96.04
	KristenAndSara	NA	NA	NA	90.42	85.63	95.00	92.21	84.18	93.58
Average		93.09	87.12	92.35	91.45	84.17	88.96	93.55	84.69	87.38

to static backgrounds and few details), but when looking at the rest of the sequences in Tables III and IV, it can be seen that RA obtains better results than LB since the ME module is more complex for the RA configuration. Furthermore, it can be seen in Table IV that Class E has the worst BD rate, although this is due to the fact that videoconferencing achieves very good coding performance in the anchor transcoder, so increasing the bitrate in the same absolute quantity as in other classes (or even in a lower quantity) results in a higher relative increase, which is what the BD-rate measures. All these facts are also present in the LP configuration.

Finally, the results obtained prove a good tradeoff between the complexity reduction and the encoding performance provided by our proposal, which can be applied in a scalable way. The global experimental results confirm that the proposed algorithm can reduce the computational complexity of the H.264/HEVC transcoder by more than a half with a slight BD-rate increase, favoring the real-time software and hardware implementation.

C. Actual Hit Rate of Classifying Models

Another way to evaluate the algorithm is by looking at the actual hit rate (that is, the rate of correctly classified CU splittings) achieved by the algorithm. Table VI shows these hit rates. The hit rate for a sequence is shown as the average hit rate for the four QP values described above.

It can be seen that high hit rates are achieved on average, which demonstrates the accuracy of the classifying algorithm, which cannot be easily improved upon, since hit rates higher

TABLE VII
COMPARISON BETWEEN PTCM-LDF, FQLD, AND AFQLD ALGORITHMS USING LP CONFIGURATION WITH ONE REFERENCE FRAME

		AFQLD		PTCM-LDF [21]		FQLD [29]	
		BD-Rate (%)	Speed-up	BD-Rate (%)	Speed-up	BD-Rate (%)	Speed-up
Class B	Kimono	2.3	2.30	2.6	2.29	2.3	2.73
	ParkScene	1.8	2.22	3.7	2.68	0.9	2.53
	Cactus	3.4	2.03	5.0	2.38	6.1	2.20
	BasketballDrive	3.2	1.99	4.0	1.81	7.6	2.22
Class C	BasketballDrill	2.7	1.74	5.7	1.78	4.3	2.09
	BQMall	1.6	1.69	5.9	2.09	4.0	2.10
	PartyScene	0.9	1.52	3.4	1.96	2.2	2.04
	RaceHorsesC	0.9	1.50	5.1	1.96	3.7	1.94
Average		2.1	1.87	4.4	2.12	3.9	2.23

TABLE VIII
COMPARISON BETWEEN AFQLD, ECU, AFQLD + ESD + CFM, AND ECU + ESD + CFM

		AFQLD		ECU		AFQLD+ESD+CFM		ECU+ESD+CFM	
		BD-Rate (%)	Speed-up	BD-Rate (%)	Speed-up	BD-Rate (%)	Speed-up	BD-Rate (%)	Speed-up
Class A	Traffic	3.0	2.71	1.4	2.05	4.4	3.72	3.8	2.79
	PeopleOnStreet	1.6	1.73	0.9	1.24	2.6	2.08	2.8	1.52
	NebutaFestival	3.2	2.03	8.0	1.32	6.7	2.30	15.9	1.52
	SteamLocomotive	2.2	2.37	1.0	1.32	3.2	3.09	2.5	2.35
Class B	Kimono	2.3	2.42	1.2	1.74	3.4	3.08	3.0	2.14
	ParkScene	3.3	2.52	2.1	1.86	5.0	3.27	5.4	2.36
	Cactus	2.9	2.39	2.1	1.79	4.5	3.05	5.2	2.26
	BasketballDrive	4.7	2.26	1.3	1.62	6.3	2.78	3.7	2.00
	BQTerrace	4.3	2.74	4.4	2.05	7.4	3.68	10.5	2.69
Class C	BasketballDrill	3.7	2.10	1.6	1.52	5.3	2.59	4.4	1.87
	BQMall	2.9	2.22	1.5	1.68	4.1	2.86	4.2	2.14
	PartyScene	1.3	1.96	3.0	1.53	2.7	2.44	6.8	1.89
	RaceHorsesC	1.3	1.46	1.5	1.27	3.4	2.09	4.9	1.51
Class D	BasketballPass	4.5	1.86	1.7	1.34	6.4	2.22	4.8	1.63
	BQSquare	1.8	2.27	2.4	1.75	4.0	3.02	7.0	2.42
	BlowingBubbles	1.4	1.96	2.9	1.57	2.7	2.47	6.3	1.96
	RaceHorses	1.3	1.66	1.8	1.22	3.2	1.96	5.4	1.47
Average		2.7	2.16	2.3	1.61	4.4	2.75	5.7	2.03

than 95% are not usually achieved in real applications (and some of them are even higher than that value). In fact, it can be seen that all values are higher than 80%. In the case of Class E with LB and LP configurations, the values range from 81% to 96%, which strengthens the idea that the classifiers work equally well for Class E as for other classes even though the BD rate is higher (due to the good performance of the original encoder for these sequences in terms of RD, as stated above).

D. Comparison With Other Transcoding Optimizations

Table VII shows the comparison between the AFQLD, PTCM-LDF [21], and FQLD [29] algorithms. It should be noted that PTCM-LDF can be parametrized with the number of frames used to train and the length of the sequence. The shown results correspond to the average value of sequences of lengths of 2.5, 5, and 10 s and using 10 frames for the training (this way of showing the results is the same as Peixoto *et al.* [21] used to summarize them).

In this case, the same configuration as in [21] (LP configuration with only onereference frame) is used for AFQLD and FQLD in order to obtain comparable results to those presented. The results show the BD rate and the speedup for all the sequences that were used in [21]. It can be clearly seen that AFQLD outperforms both PTCM-LDF and FQLD in terms of BD rate. In the case of speedup, AFQLD results are a bit lower, but it should be noted that in terms of time reduction, AFQLD obtains 47% and FQLD obtains 55%, which are really close values, and the wide gain in BD rate makes AFQLD to perform better globally. This same fact happens when the comparison is between AFQLD and PTCM-LDF, but in this case, the speedups are even closer, since the time reduction of PTCM-LDF is 52%.

E. Comparison With Non-Transcoding-Specific Optimizations

Table VIII shows the comparison between the AFQLD algorithm and the nontranscoding-specific optimizations integrated in HM reference software, namely, ECU [16], ESD [14], and CFM [15], and described in Section II, using RA configuration. The idea is to compare our proposal (using level $=$ 2) with respect to another transcoder where the re-encoding stage enables the aforementioned techniques. A statistical comparison following the methodology presented in [42] has been carried out using data from Table VIII. First of all, a Friedman test for the BD-rates and the speedups has been used, with the result that all methods are not equivalent (in both, BD-rate and speedup terms). Then, a posthoc test using an Holm adjust method has been used in order to compare pairwise methods. These tests show that ECU

and AFQLD algorithms present comparable BD-rate values, while AFQLD outperforms ECU in terms of speedup. This result is also true when comparing AFQLD + ESD + CFM and ECU + ESD + CFM. Moreover, when comparing AFQLD and ECU + ESD + CFM, the tests show that the speedups are comparable, while AFQLD outperforms ECU + ESD + CFM in BD rate, showing the effectiveness of the proposed algorithm.

VI. Conclusion

In this paper, we have presented a new approach for accelerating the CU partitioning decision of an H.264/HEVC video transcoder, based on a statistical NB classifier algorithm, which decides on the most appropriate quadtree level by taking into account what has occurred in the H.264/AVC decoder, without the need for testing all the CUs/PUs. This allows an early classification of CUs/PUs, avoiding the exhaustive RDO evaluation of all the available CU sizes. Moreover, this algorithm is adaptive due to the fact that the algorithm can be dynamically adapted to the content of each sequence.

The simulation results obtained demonstrate a good tradeoff between the complexity reduction and the encoding performance provided by our proposal, which can be applied in an incremental way. The full implementation of the AFQLD algorithm for the three levels achieves a quantitative speedup of around 2.31× on average (with a time reduction of 56.7%) and a BD-rate penalty of around 3.4%, compared with the anchor transcoder, for a wide range of resolutions (from Classes A to E). For moderate complexity reduction, our AFQLD algorithm also allows the implementation of only the first depth level, obtaining a 26.7% time saving with a negligible rate penalty of around 0.8%. Furthermore, the AFQLD algorithm can achieve better BD rates than a state-of-the-art transcoding algorithm, such as PTCM-LDF [21] and FQLD [29]. It can also obtain better performance than non-transcoding specific algorithms, such as ECU, ESD, or CFM.

Finally, the proposed algorithm is suitable for H.264/HEVC hardware and software real-time transcoder realization, thanks to the noncomplex classifier implemented and a set of noncomplex computable features.

Acknowledgment

The authors would like to thank E. Peixoto for sharing the results of his experiments.

References

[1] *High Efficiency Video Coding*, document ITU-T Rec. H.265 and ISO/IEC 23008-2 (Version 1), Apr. 2013.

[2] *High Efficiency Video Coding*, document ITU-T Rec. H.265 and ISO/IEC 23008-2 (Version 2), Oct. 2014.

[3] *Advanced Video Coding for Generic Audiovisual Services*, document ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC) (Version 22), Feb. 2012.

[4] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—Including High Efficiency Video Coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012.

[5] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[6] C. C. Chi *et al.*, "Parallel scalability and efficiency of HEVC parallelization approaches," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1827–1838, Dec. 2012.

[7] G. Bjøntegaard, *Improvements of the BD-PSNR Model*, ITU-T SG16 Q6, document VCEG-AI11, Jul. 2008.

[8] W.-J. Han *et al.*, "Improved video compression efficiency through flexible unit representation and corresponding extension of coding tools," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 12, pp. 1709–1720, Dec. 2010.

[9] L. W. K. Panusopone and X. Fang, *Efficient Transform Unit Representation*, document JCTVC-D250, Jan. 2011.

[10] X. Li, M. Wien, and J.-R. Ohm, "Rate-complexity-distortion optimization for hybrid video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 957–970, Jul. 2011.

[11] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Performance and computational complexity assessment of High-Efficiency Video Encoders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1899–1909, Dec. 2012.

[12] F. Bossen, B. Bross, K. Sühring, and D. Flynn, "HEVC complexity and implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.

[13] *HM Reference Software*. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/, accessed Mar. 1, 2015.

[14] J. Yang, J. Kim, K. Won, H. Lee, and B. Jeon, *Early SKIP Detection for HEVC*, document JCTVC-G543, Nov. 2011.

[15] R. H. Gweon and Y.-L. Lee, *Early Termination of CU Encoding to Reduce HEVC Complexity*, document JCTVC-F045, Jul. 2011.

[16] K. Choi, S. Park, and E. S. Jang, *Coding Tree Pruning Base CU Early Termination*, document JCTVC-F902, Jul. 2011.

[17] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: An overview," *IEEE Signal Process. Mag.*, vol. 20, no. 2, pp. 18–29, Mar. 2003.

[18] D. Zhang, B. Li, J. Xu, and H. Li, "Fast transcoding from H.264 AVC to High Efficiency Video Coding," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Melbourne, Vic., Australia, Jul. 2012, pp. 651–656.

[19] E. Peixoto and E. Izquierdo, "A complexity-scalable transcoder from H.264/AVC to the new HEVC codec," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Orlando, FL, USA, Sep. 2012, pp. 737–740.

[20] E. Peixoto, B. Macchiavello, E. M. Hung, A. Zaghetto, T. Shanableh, and E. Izquierdo, "An H.264/AVC to HEVC video transcoder based on mode mapping," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Melbourne, Vic., Australia, Sep. 2013, pp. 1972–1976.

[21] E. Peixoto, T. Shanableh, and E. Izquierdo, "H.264/AVC to HEVC video transcoder based on dynamic thresholding and content modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 1, pp. 99–112, Jan. 2014.

[22] E. Peixoto, B. Macchiavello, E. M. Hung, and R. L. de Queiroz, "A fast HEVC transcoder based on content modeling and early termination," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Paris, France, Oct. 2014, pp. 2532–2536.

[23] P. Xing, Y. Tian, X. Zhang, Y. Wang, and T. Huang, "A coding unit classification based AVC-to-HEVC transcoding with background modeling for surveillance videos," in *Proc. Vis. Commun. Image Process. (VCIP)*, Nov. 2013, pp. 1–6.

[24] W. Jiang, Y. Chen, and X. Tian, "Fast transcoding from H.264 to HEVC based on region feature analysis," *Multimedia Tools Appl.*, vol. 73, no. 3, pp. 2179–2200, Dec. 2014.

[25] L. P. Van *et al.*, "Fast transrating for High Efficiency Video Coding based on machine learning," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Melbourne, Vic., Australia, Sep. 2013, pp. 1573–1577.

[26] J. De Praeter, J. De Cock, G. Van Wallendael, S. Van Leuven, P. Lambert, and R. Van de Walle, "Efficient transcoding for spatially misaligned compositions for HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Paris, France, Oct. 2014, pp. 2487–2491.

[27] T. Shanableh, E. Peixoto, and E. Izquierdo, "MPEG-2 to HEVC video transcoding with content-based modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1191–1196, Jul. 2013.

[28] A. J. Diaz-Honrubia, J. L. Martinez, and P. Cuenca, "Multiple reference frame transcoding from H.264/AVC to HEVC," in *Proc. Int. Conf. Multimedia Model. (MMM)*, Dublin, Ireland, Jan. 2014, pp. 593–604.

[29] A. J. Diaz-Honrubia, J. L. Martinez, J. M. Puerta, J. A. Gamez, J. de Cock, and P. Cuenca, "Fast quadtree level decision algorithm for H.264/HEVC transcoder," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Paris, France, Oct. 2014, pp. 2497–2501.

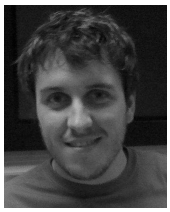
[30] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *SIGKDD Explorations Newslett.*, vol. 11, no. 1, pp. 10–18, 2009.

[31] F. Bossen, *Common HM Test Conditions and Software Reference Configurations*, document JCTVC-L1100, Jan. 2013.

[32] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "The KDD process for extracting useful knowledge from volumes of data," *Commun. ACM*, vol. 39, no. 11, pp. 27–34, Nov. 1996.
[33] G. Fernandez-Escribano, H. Kalva, P. Cuenca, L. Orozco-Barbosa, and A. Garrido, "A fast MB mode decision algorithm for MPEG-2 to H.264 P-frame transcoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 2, pp. 172–185, Feb. 2008.
[34] S. Patnaik and Y.-M. Yang, *Soft Computing Techniques in Vision Science*, vol. 395. New York, NY, USA: Springer-Verlag, 2012.
[35] *Subjective Video Quality Assessment Methods for Multimedia Applications*, document ITU-R P.910, 1999.
[36] U. Fayyad and K. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proc. Int. Joint Conf. Uncertainty AI*, Aug. 1993, pp. 1022–1027.
[37] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.
[38] M. Minsky, "Steps toward artificial intelligence," *Trans. Inst. Radio Eng.*, vol. 4, pp. 8–30, Jan. 1961.
[39] P. Domingos and M. Pazzani, "Beyond independence: Conditions for the optimality of the simple Bayesian classifier," in *Proc. Int. Conf. Mach. Learn.*, Jul. 1996, pp. 105–112.
[40] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques* (The Morgan Kaufmann Series in Data Management Systems). San Mateo, CA, USA: Morgan Kaufmann, 2011.
[41] *JM Reference Software, Version 18.4*. [Online]. Available: http://iphome.hhi.de/suehring/tml/, accessed May 14, 2014.
[42] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.

Antonio Jesús Díaz-Honrubia (S'14) received the bachelor's degree in computer science and engineering, and the M.Sc. degrees in advanced computing technologies and computer science and engineering, from University of Castilla-La Mancha, Albacete, Spain, in 2011 and 2012, respectively, where he is currently working toward the Ph.D. degree in advanced computing technologies.

He is with University of Castilla-La Mancha, where he carries out his research activities with the Computer Architecture and Technology Group, Albacete Research Institute of Informatics. His research interests include video transcoding from H.264/AVC to the High Efficiency Video Coding standard and 3D video coding.

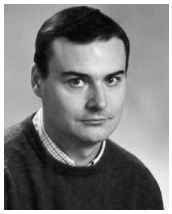


José Luis Martínez received the M.Sc. and Ph.D. degrees in computer science and engineering from University of Castilla-La Mancha, Albacete, Spain, in 2005 and 2009, respectively.

He was a Post-Doctoral Researcher with the Centre for Communication System Research, University of Surrey, Guildford, U.K. In 2011, he joined University of Castilla-La Mancha, where he is currently an Assistant Professor. He was a Visiting Researcher with Florida Atlantic University, Boca Raton, FL, USA, for nine months. He has 26 publications in these areas in international refereed journals and 60 conference proceedings. His research interests include video transcoding, scalable video coding, and video applications for multicore and GPU architectures.

Pedro Cuenca received the M.Sc. (Hons.) degree in physics with a minor in electronics and computer science from University of Valencia, Valencia, Spain, in 1994, and the Ph.D. degree in computer engineering from Polytechnic University of Valencia, Valencia, in 1999.

He joined the Department of Computer Engineering, University of Castilla-La Mancha, Albacete, Spain, in 1995, where he is currently a Full Professor. He has been a Visiting Researcher with Nottingham Trent University, Nottingham, U.K.; University of Ottawa, Ottawa, ON, Canada; and University of Surrey, Guildford, U.K. He has authored over 150 papers in international journals and conferences. His research interests include video compression, QoS video transmission, and video applications for multicore and GPU architectures.



José Antonio Gamez received the M.S. degree in computer science and the Ph.D. degree in computer science from University of Granada, Granada, Spain, in 1991 and 1998, respectively.

He joined the Department of Computer Science, University of Castilla-La Mancha, Albacete, Spain, in 1991, where he is currently a Full Professor. He has edited five books and five special issues of international journals. He has (co-)authored over 100 papers in journals, books, and refereed international conferences. His research interests include probabilistic reasoning, Bayesian networks, evolutionary algorithms, machine learning, and data mining.

José Miguel Puerta received the M.S. and Ph.D. degrees in computer science from University of Granada, Granada, Spain, in 1991 and 2001, respectively.

He joined the Department of Computer Systems, University of Castilla-La Mancha, Albacete, Spain, in 1991, where he is currently an Associate Professor. He has authored over 50 papers. His research interests include probabilistic graphical models, Bayesian networks, evolutionary algorithms, machine learning, and data mining.

2.2 CTU Splitting Algorithm for H.264/AVC and HEVC Simultaneous Encoding

- **Title:** CTU Splitting Algorithm for H.264/AVC and HEVC Simultaneous Encoding
- **Authors:** Antonio Jesús Díaz-Honrubia, Johan De Praeter, Glenn Van Wallendael, José Luis Martínez, Pedro Cuenca, José Miguel Puerta, and José Antonio Gámez
- **Type:** Journal
- **Journal:** The Journal of Supercomputing
- **Publisher:** Springer
- **ISSN:** 1573–0484
- **State:** Published online
- **Year:** 2016
- **DOI:** 10.1007/s11227-016-1683-1
- **URL:** http://link.springer.com/article/10.1007%2Fs11227-016-1683-1
- **Category**: Computer science, hardware and architecture
- **Impact Factor:** 1.088
- **JCR ranking:** Q2

J Supercomput
DOI 10.1007/s11227-016-1683-1

CTU splitting algorithm for H.264/AVC and HEVC simultaneous encoding

A. J. Diaz-Honrubia[1] · J. De Praeter[2] · G. Van Wallendael[2] · J. L. Martinez[1] · P. Cuenca[1] · J. M. Puerta[3] · J. A. Gamez[3]

Abstract *High-Efficiency Video Coding* (HEVC) was conceived by the *Joint Collaborative Team on Video Coding* as the natural successor of the H.264/AVC standard, which has been the most extended digital video standard in all segments of the domestic and professional markets for over 10 years. HEVC roughly doubles the compression

This work has been jointly supported by the MINECO and European Commission (FEDER funds) under the Projects TIN2012-38341-C04-04, TIN2013-46638-C3-3-P and TIN2015-66972-C5-2-R. This work has also been supported by the Spanish MECD under Grant FPU 12/00994. Finally, it is partly supported by the European Union FEDER (CAPAP-H5 network TIN2014-53522-REDT).

✉ A. J. Diaz-Honrubia
antonio.dhonrubia@uclm.es

J. De Praeter
Johan.DePraeter@ugent.be

G. Van Wallendael
Glenn.VanWallendael@ugent.be

J. L. Martinez
JoseLuis.Martinez@uclm.es

P. Cuenca
Pedro.Cuenca@uclm.es

J. M. Puerta
Jose.Puerta@uclm.es

J. A. Gamez
Jose.Gamez@uclm.es

1 High Performance Networks and Architectures Laboratory, University of Castilla-La Mancha, Albacete, Spain

2 Data Science Lab, Ghent University, Ledeberg-Ghent, Belgium

3 Intelligent Systems Laboratory, University of Castilla-La Mancha, Albacete, Spain

Published online: 27 February 2016

performance of H.264/AVC in Rate-Distortion terms at the expense of a high computational cost. Thus, most of the previous-generation devices can decode H.264/AVC streams but they cannot decode HEVC yet, while emerging devices are supposed to be able to decode both standards. Video providers should take advantage of bandwidth reduction using HEVC when possible, but they should also provide compatible streams to older devices, making it necessary to encode the same stream using both H.264/AVC and HEVC standards. This paper presents a coding tree unit splitting algorithm for a heterogeneous simultaneous encoding scenario which makes use of information from H.264/AVC encoder to make faster decisions in HEVC. Experimental results show that the proposed algorithm can achieve a good trade-off between coding efficiency and complexity.

Keywords HEVC · H.264/AVC · Simultaneous encoding · CTU splitting

1 Introduction

In the last years, H.264/Advance Video Coding (AVC) [14] has been the most widely used video compression standard for High-Definition (HD) video coding, but in April 2013, the *High-Efficiency Video Coding* (HEVC) standard [15] was established by the *Joint Collaborative Team on Video Coding* (JCT-VC). The main goal of HEVC was to achieve an efficient performance for delivering high-quality multimedia services over bandwidth-constrained networks, but also to give support to *Ultra-High Definition* (UHD), which demands a high bandwidth. In terms of Rate-Distortion (RD) performance, HEVC roughly doubles the compression performance of H.264/AVC but at a cost of extremely high computational requirements during encoding [17].

At the same time, new video-enabled devices have emerged in the market, such as smartphones or tablets. The requirements and constraints of these devices have changed, in turn, the way video streaming is demanded. In particular, the newest ones are able to decode HEVC streams, taking advantage of its superior compression performance. Older devices, however, are limited to decoding H.264/AVC streams.

In this context, content providers are required to support multiple formats and profiles. As an example, Adhikari et al. perform an analysis of the movie contribution, encoding and delivery system of a well-known streaming company [1], detailing how each of their movies is encoded at different bit rates and formats. When a user requests a movie, a manifest file is sent to the server indicating the formats that the device is able to decode. Then, the server decides the best format to send among those indicated. This process requires to encode the same movie with multiple bit rates and formats. However, this is too computationally expensive and, as shown in [1], to perform all these encodings, the company subcontracts cloud services due to the complexity of the tasks.

This goal could be achieved with the *Scalability Extension of HEVC* (SHVC) [5], which adds support for hybrid video coding, using H.264/AVC for the base layer and HEVC for the enhancement layers. Nevertheless, while this hybrid approach provides backward compatibility with H.264/AVC, it implies an overhead, since HEVC layer would be sent to H.264/AVC-capable devices (but not decoded) and HEVC-capable

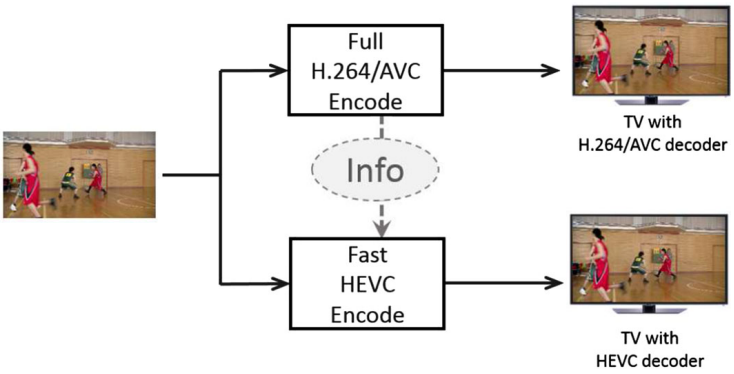


Fig. 1 Simultaneous encoding scenario

devices would receive the H.264/AVC layer, which implies a higher bit rate. Therefore, the use of hybrid SHVC would result in a bit rate overhead and in a higher complexity issue, since the enhancement layers use both regular and inter-layer predictions. Moreover, the experience with previous standards shows that scalable decoders are not usually implemented since, even scalable video suppose a bit rate saving, it also implies more expensive decoders, being harmful for the final user.

On the other hand, this paper presents a novel approach called heterogeneous simultaneous video encoding. In this approach, a video stream is encoded in both H.264/AVC and HEVC standards at the same time, so that an HEVC-capable device can take advantage of the bit rate reduction, while a compatible stream for H.264/AVC-capable devices is provided too, as shown in Fig. 1. While the H.264/AVC encoder remains unaltered, the HEVC encoder, which is substantially more complex, re-uses information calculated in the H.264/AVC stage. This solution provides support for a wide range of devices, while the overall complexity of the simultaneous encoder is reduced with the use of the proposed algorithm.

The proposal adapts a soft computing transcoding algorithm, called *Adaptive Fast Quadtree Level Decision* (AFQLD) [7], to this new scenario of collaborative coding between H.264/AVC and HEVC. AFQLD aims to exploit the information gathered in the H.264/AVC decoder to assist decisions on HEVC encoder using statistical *Naïve-Bayes* (NB) classifiers to avoid an exhaustive search. In an off-line data mining process, all the knowledge needed is fetched from the H.264/AVC decoder and is then converted into mathematical models, by means of *Machine Learning* (ML) techniques. However, the main novelty of this work with respect to [7] is that, in the current scenario, more information is present, since it can be gathered not only at the decoding stage, but also at the encoding one. Therefore, new and more accurate models have been trained using the information from the H.264/AVC encoder. Moreover, some details of the original algorithm has been adapted to the scenario, as it will be described at Sect. 4.

The remainder of this paper is organized as follows: Sect. 2 includes technical background to the new HEVC standard, while Sect. 3 details the related work. Section 4 describes the proposed modifications of the AFQLD algorithm. Experimental results are given in Sects. 5, and 6 concludes the paper.

2 Technical background

HEVC is based on a block-based hybrid approach used in all previous video compression standards. In addition, new tools have been introduced in HEVC that increase its coding efficiency compared with H.264/AVC [17], such as the new *Coding Unit* (CU) partitioning based on a hierarchical block structure, based on *Coding Tree Units* (CTUs), new transform sizes of 16×16 and 32×32, and a new tool in the decoding loop called *Sample Adaptive Offset* (SAO), which is applied to the reconstructed samples after the deblocking filter, with the goal of improving the perceived quality of the decoded sequence [21]. HEVC defines a new flexible CTU structure [12] with the aim of achieving an optimal adaptation of the CU to the content details. The CTU size can vary from 64×64 to 16×16 pixels, and each of them can be iteratively partitioned into four square sub-blocks, named CUs, with a minimum allowable size of 8×8 pixels. Therefore, a CTU can be further partitioned into four *Depth Levels*, from $d = 0$ (CU size of 64×64) $d = 3$ (CU size of 8×8), having 4^d CUs in each depth level. Thus, a CU in depth level d can be denoted as $\mathrm{CU}_{d,k}$ ($k = 0, 1, \ldots, 4^d - 1$).

HEVC increases flexibility, each $\mathrm{CU}_{d,k}$ becoming a root of two new trees containing two new unit types: the *Prediction Units* (PUs), and the *Transform Units* (TUs). The PU size can be one of the following: $2N \times 2N$, $2N \times N$, $N \times 2N$, $N \times N$, $2N \times \mathrm{nU}$, $2N \times \mathrm{nD}$, $\mathrm{nL} \times 2N$, $\mathrm{nR} \times 2N$). The prediction residue obtained in each of the PUs is transformed using a *Residual Quad Tree* structure, which supports various TU sizes from 32×32 to 4×4. In Fig. 2, an example of the partitioning is shown, depicting how a CTU is structured in a hierarchical tree where each CU branch ends in a leaf ($\mathrm{CU}_{d,k}$).

Furthermore, in the case of inter prediction, for each of these PU partitions a *Motion Estimation* (ME) algorithm is called, and in the case of intra-prediction, for each of these PU partitions 35 different coding modes can be evaluated. Therefore, the *Rate-Distortion Optimization* (RDO) model needs to evaluate each combination of CUs, PU partitioning modes, and TUs. As expected, the selection of the optimal partitioning for each of the three trees is an intensive time-consuming process due to the huge number of combinations that have to be evaluated [4].

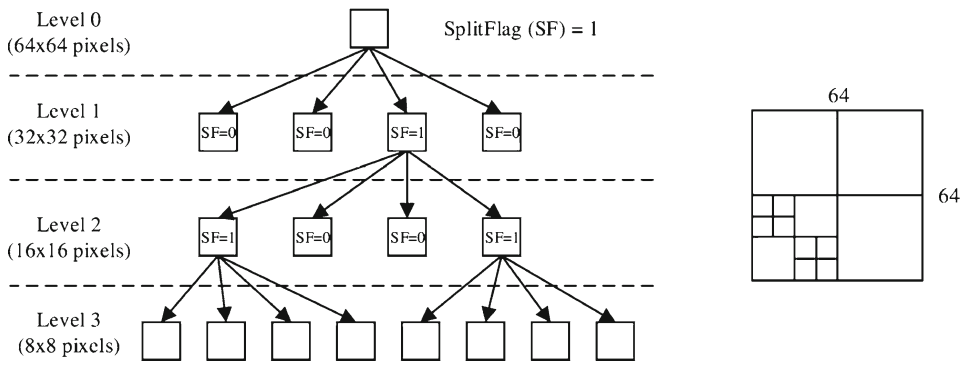


Fig. 2 Quadtree CTU splitting illustration

3 Related work

Related work on simultaneous encoding focuses on reducing the complexity of creating multiple bit rate representations of the same video. One approach that tries to exploit the redundancy between the different bit rate representations by performing ME only for the reference stream and re-using these decisions for the other streams is presented in [22]. A similar approach of re-using ME has also been used for simultaneous encoding in VP8 [8,10]. However, although the encoding complexity is reduced, the compression efficiency suffers significantly.

With the advent of HEVC, in which much of the complexity stems from determining the optimal structure out of many possible block structures, work on simultaneous encoding focuses on reducing the encoding complexity by limiting the amount of CUs and/or PUs to be evaluated in the simultaneously encoded streams based on the encoding decisions in the reference stream [6,19]. However, in this case, the reference stream is still encoded in HEVC. Using the less computationally complex H.264/AVC standard to encode the reference stream and using this information to accelerate the encoding of HEVC, the overall complexity of the simultaneous encoder can still be reduced.

To the knowledge of the authors, there are no other works at the moment of writing where the information fetched from an encoder is re-used to accelerate the encoding of another stream in a different standard. Furthermore, the presented heterogeneous approach can be considered more profitable than a homogeneous simultaneous video encoder because in this new case the reference stream (which is not accelerated) is less computationally complex than the accelerated stream (the first one is encoded with H.264/AVC and the second one with HEVC). Hence, the overall complexity of the simultaneous encoder is also reduced.

However, other scenarios are technically similar to the simultaneous encoding. For instance, video transcoding aims to convert a compressed video stream encoded with a given format and characteristics into another video stream encoded with a different codec or features. Several approaches have been developed for a H.264/AVC to HEVC video transcoder (see e.g., [7,18] or [20]) which could be extended for its use on the proposed scenario. Specifically, a previous work of the authors, [7], proposes a fast CTU splitting algorithm, so-called *Adaptive Fast Quadtree Level Decision* (AFQLD), for the aforementioned transcoder. This approach is described in the following subsection.

3.1 AFQLD algorithm

In [7], the CTU splitting problem is posed as a data classification problem where the information gathered in the H.264/AVC decoder assists decisions on CU splitting using a statistical *Naïve-Bayes* (NB) classifier to avoid the exhaustive RDO search over all possible CU sizes and its possible modes.

The algorithm, which is shown in Algorithm 1, tries to accelerate HEVC decisions by labeling the current CU as to split (C_S) or not to split (C_N) using the information fetched from H.264/AVC decoder. The decision at levels 0 and 1 (CU size of 64×64

Algorithm 1 AFQLD Algorithm [7]

```
Decode H.264/AVC sequence
Cost[1] = Cost[2] = Cost[3] = Cost[4] = 0
for all GOPs in sequence do
  for all frames in GOP do
    for all CTUs in frame do
      if First GOP and cost[currentFrameEnergy]==0 then
        cost[currentFrameEnergy] += SplittingError
      else
        for all Not finished CUs in CTU do
          if Level ∈ {0, 1, 2} then
            if Level == 2 then
              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
            else
              Classify as C_S or C_N according to the decision of the appropriate model M_l
            end if
            if Decision == C_S then
              if Level == 2 then
                Calculate all PUs
              else
                Calculate Skip and 2Nx2N PUs
              end if
              Split into 4 sub-CUs
            else
              Calculate all PUs
              Select the best CU and PU in RD terms and finish this CU
            end if
          else
            Calculate all PUs
            Select the best CU and PU in RD terms and finish this CU
          end if
        end for
      end if
    end for
  end for
end for
```

and 32 × 32 pixels, respectively) relies on NB classifiers, as they are the simplest representative of probabilistic classifiers. At level 2, the algorithm takes advantage of the fact that CU size at this level is the MB size in H.264/AVC too. This means that H.264/AVC has already performed an RDO search between the available modes and it chose the best one. Therefore, just copying the decision from H.264/AVC might be a good solution. However, given the new tools introduced by HEVC, inspecting

the adjacent MBs is also a good idea to know the complexity of the area. Thus, the decision at this level is made as shown in lines 12–22 of Algorithm 1.

AFQLD learns the NB classifiers using a dataset which contains 26 relevant variables fetched from H.264/AVC decoder (see [7] for details) from sequences of different resolutions and using different QP values. The features in this dataset are discretized and, then, a feature subset selection is applied to select the proper subset of features.

Furthermore, the models are calibrated online, taking into account the sequence which is being encoded. The basic classification rule [choose C_N if $P(C_N) > P(C_S)$] is replaced with a cost-based rule: choose $P(C_S)$ if $P(C_N) \times \text{Cost}_{NS} > P(C_S) \times \text{Cost}_{SN}$. These costs are estimated using the Lagrangian cost of splitting (L_S) and of not splitting (L_N) and the concept of absolute error:

$$\text{Cost}_{ij} = |\text{L}_j - \text{L}_i| \times \omega_{ij} \tag{1}$$

where $i, j \in \{S, N\}$, i being the predicted decision, j being the correct one, and ωij being an associated weight for each particular cost ($\omega_{NS} = 2.0$ and $\omega_{SN} = 1.0$).

To calculation these costs, the first frame of the current residual energy level is encoded by running the algorithm but letting the HEVC encoder make the correct decision so that it can calculate the actual and the predicted decisions.

As can be derived from the algorithm description, AFQLD could be straightly reused in the current scenario. However, we can take advantage of the fact that in the case of simultaneous encoding, the H.264/AVC encoding information is also present and the models can be improved with this new information.

4 AFQLD algorithm adaption for simultaneous encoding

As mentioned above, the HEVC encoding process is very computation complex. However, according to Sect. 3, some techniques which can reduce the encoding or transcoding complexity have been developed. However, in the scenario of this paper the information of the H.264/AVC encoder is also available and, then, we can use it to improve the models. Therefore, we have used the same algorithm structure as a transcoding algorithm (AFQLD in this case, see Algorithm 1), but replacing the H.264/AVC decoder with the encoder and improved the way that the bayesian models (which are used to make the decisions stated at line 24 of Algorithm 1) are learnt.

Thus, a new model $\mathcal{M}_l$ for making the decision at each level $l = 0, 1, 2$ of the quadtree is needed. As described in Sect. 3, we rely on a machine learning approach for levels 0 and 1, while at level 2 the decision is made in the same way as in the AFQLD algorithm, as can be seen in lines 12–22 of Algorithm 1. A different model is also needed depending on the frame residual energy, where four levels of energy have been considered (1, 2, 3 and 4) where 1 represents high residual energy and 4 represents low residual energy. For the machine learning models, NB classifiers have been selected again. They rely on a strong independence assumption: all features are conditionally independent given the class. This can be a considerable drawback in theory, but in practice NB shows a competitive performance with respect to more complex algorithms. Furthermore, this independence assumption brings the following

important advantages (e.g., NB is memory and space efficient both at learning and classification). In this scenario, the NB classifiers must decide whether the CU at the current level should be further split (C_S) or not (C_N), where C_S and C_N are the labels of the class variable to predict.

An NB classifier computes the posterior probability distribution of the class variable C given the values of the features $(f_1, \ldots, f_n)$, and it chooses the class $\{c_1, \ldots, c_K\}$ with the highest probability. Applying Bayes' theorem and realizing that the denominator is the same for all class labels, we get:

$$c^* = \underset{C \in \{c_1, \ldots, c_K\}}{\operatorname{argmax}} P(C|f_1, \ldots, f_n) = \underset{C \in \{c_1, \ldots, c_K\}}{\operatorname{argmax}} P(f_1, \ldots, f_n|C)P(C). \quad (2)$$

Obviously, managing the joint probability distribution of all the variables becomes unfeasible even for small datasets. However, thanks to the conditional independence assumption, the joint distribution is approximated by the multiplication of the probability of the class and of each feature given the class, and so the NB classifier can be formulated as:

$$c^* = \underset{C \in \{c_1, \ldots, c_K\}}{\operatorname{argmax}} P(C) \prod_{i=1}^{n} P(f_i|C).$$

Thus, if f_i is a discrete variable, its conditional probability is computed by counting relative frequencies, which would only take $\mathcal{O}(Nn)$ in learning phase, N being the number of instances and n the number of features in database. When f_i is numeric, the model assumes that its values follow a Gaussian distribution in the instances of each class. However, to avoid this assumption, the variables will be discretized. Moreover, since NB is linear in classification phase, its complexity is $\mathcal{O}(n)$.

On the one hand, as the classifier may misclassify partitions occasionally, in the levels predicted as C_S, $2N \times 2N$ and Skip PUs are anyway checked (and all PUs are checked at level 2 even though C_S is chosen). On the other hand, if the decision is C_N, then the current depth is considered as final and all PUs at this CU depth are evaluated and the algorithm for this CTU finishes.

For each model, a new dataset has been prepared with relevant variables from different sequences with different resolutions (*PeopleOnStreet, ParkScene, PartyScene* and *BQSquare*). For each sequence, the first 1000 CUs of each pair of frame energy and QP values {22, 27, 32, 37} with *Random Access* (RA) configuration were selected to be used for the learning process, which results in a total of 8 classifiers and 16,000 samples to learn each classifier. Regarding the variables, the initial set of features, **F**, contains a whole of 53 continuous variables. The first 26 are the same used in the original proposal of AFQLD (see [7] for details about them), while the remaining features represent the information which is available in the H.264/AVC encoder and not in the decoder:

- $w_{RDCostMode[i]}$ the RD cost of the i mode, where i is Skip, 16×16, 16×8, 8×16, 8×8 (the best RD cost of all possible 8×8 and smaller partitions), Intra 16×16, Intra 8×8, Intra 4×4, 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 16×16, 16×8, 8×16 and 8×8.
- $w_{VarMVx[i]}$, $w_{VarMVy[i]}$ variance of all x and y MV components, respectively, of each i mode, where i is 16×16, 16×8, 8×16 and 8×8.
- $w_{VarIntraDir8x8}$, $w_{VarIntraDir4x4}$ variance of all intra-directions of Intra 8×8 and Intra 4×4 modes.

Initially, the 53 features are of a numerical nature but they are discretized using the entropy-based Fayyad–Irani algorithm [9]. Then, a *feature subset selection* (*FSS*) is applied to select the proper subset of features [11]. We selected 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 FSS process finishes when the addition of features no longer improves the accuracy of the classifier.

Finally, the same calibrating model as described in Sect. 3.1 has been used, which takes into account the characteristics of the sequence which is being encoded. Thus, the classification rule is to choose $P(C_S)$ if $P(C_S) \times \text{Cost}_{SN} > P(C_N) \times \text{Cost}_{NS}$, calculating the costs as described in Eq. (1). However, it should be taken into account that in this scenario the Lagrangian costs are lower than in the transcoding scenario, since the sequence has not been encoded and decoded previously and, then, the differences between the original and the encoded sequences are lower. Moreover, the absolute error, which is a scale-sensitive metric, is used to calculate the costs. Thus, these two facts jointly produce that the costs calculated in the simultaneous encoding scenario are lower than in the transcoding scenario, which may cause misclassification. To solve this problem, the ω_{ij} values have been changed.

On the one hand, ω_{SN} has been set to 1.0, since this value simplifies the classification rule. On the other hand, after heuristically trying several weights, it has been concluded that the best weight for ω_{NS} is 6.0. This value is consistent, since a higher value than ω_{SN} should be used (the mistake of not splitting when the correct decision is to split is worse than the opposite, since no more PUs will be evaluated, while in the opposite mistake the tree is allowed to go back since $2N \times 2N$ and Skip PUs are checked). Moreover, in the transcoding scenario the best value is $\omega_{NS} = 2.0$ and, as the absolute error is used to calculate the costs, in this scenario this error will be smaller than in the transcoding scenario, since in the transcoding scenario the sequence which is being encoded with HEVC has been encoded and decoded with H.264/AVC before and the distortion will be higher.

5 Performance evaluation

This section aims to evaluate the proposed adaption of the AFQLD algorithm to the simultaneous encoding scenario. To ensure a common framework, JCT-VC defined a document [3] where test conditions are set out to homogenize comparisons. Therefore, this performance evaluation has been carried out in accordance with these guidelines, using QP values {22, 27, 32, 37} and random access (RA) configuration for both the

H.264/AVC and the HEVC streams. The results are grouped into *Classes* (A, B, C and D) (as described in [3]) according to their resolution:

- Class A (2560 × 1600 pixels): *Traffic*, *PeopleOnStreet*, *Nebuta*, *SteamLocomotive*.
- Class B (1920 × 1800 pixels): *Kimono*, *ParkScene*, *Cactus*, *BQTerrace*, *BasketballDrive*.
- Class C (832 × 480 pixels): *RaceHorsesC*, *BQMall*, *PartyScene*, *BasketballDrill*.
- Class D (416 × 240 pixels): *RaceHorses*, *BQSquare*, *BlowingBubbles*, *BasketballPass*.

The software used is JM 18.4 [16] for H.264/AVC and HM 16.6 [13] for HEVC. The remainder of the coding parameters not mentioned here is kept as default in the configuration file. Thus, the process to generate these results is the following:

1. Encode the YUV file with H.264/AVC reference software JM 18.4, producing all the information needed for the algorithm.
2. Encode the YUV file with HEVC reference software HM 16.6 (*anchor*).
3. Encode the YUV file with a modified version of HM 16.6 which includes the proposed simultaneous encoding algorithm (*proposed*).
4. Compare the *anchor* and *proposed* HEVC streams to obtain the metrics.

Measurements have been performed on a six-core Intel Core i7-3930K CPU running at 3.20 GHz. The results are presented in terms of *Time Reduction* (TR) and *BD-Rate* (BDR) [2] (which measures the increment in bit rate while maintaining the same objective quality), which are calculated as indicated in (3), where t_{anchor} is the execution time of the non-accelerated *anchor* HEVC encoder, and $t_{proposed}$ is the execution time of the *proposed* encoder. The global BDR is the weighted average of the Y, U and V components (given that the luminance is four times larger than the chrominances).

$$\begin{aligned} \text{Time reduction}\,(\%) &= (t_{anchor} - t_{proposed})/(t_{anchor}) \times 100 \\ \text{BD-rate}\,(\%) &= (4 \times \text{BD-rate}_{\text{Y}} + \text{BD-rate}_{\text{U}} + \text{BD-rate}_{\text{V}})/6 \end{aligned} \tag{3}$$

As an initial evaluation, Fig. 3 shows the differences between the CU splitting of the non-accelerated *anchor* encoder (left column), and the CU splitting of the *proposed*

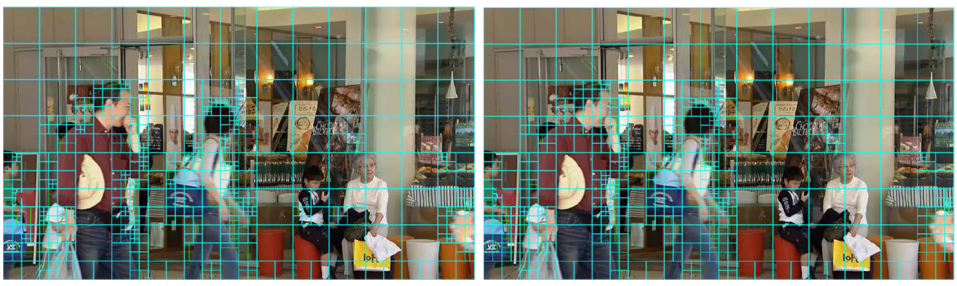


Fig. 3 CU splitting for original (*left*) and proposed (*right*) algorithms. 9th frame of *BQMall* sequence using RA configuration and QP = 27

Table 1 Coding efficiency and timing results of the proposed algorithm using RA configuration.

	Simultaneous encoding		Transcoding [7]	
	BDR (%)	TR (%)	BDR (%)	TR (%)
Class A				
Traffic	2.7	63.31	3.0	63.08
PeopleOnStreet(*)	2.4	44.26	1.6	42.23
NebutaFestival	1.3	51.31	3.2	50.80
SteamLocomotive	1.6	58.18	2.2	57.88
Class B				
Kimono	2.3	60.74	2.3	58.67
ParkScene(*)	2.2	62.48	3.3	60.29
Cactus	3.6	60.29	2.9	58.10
BasketballDrive	4.9	57.60	4.7	55.84
BQTerrace	2.3	65.49	4.3	63.44
Class C				
BasketballDrill	3.9	53.94	3.7	52.28
BQMall	3.1	56.40	2.9	55.05
PartyScene(*)	1.3	40.09	1.3	49.04
RaceHorsesC	2.5	42.36	1.3	31.60
Class D				
BasketballPass	2.6	42.17	4.5	46.27
BQSquare(*)	0.8	52.71	1.8	56.01
BlowingBubbles	1.4	45.35	1.4	49.05
RaceHorses	2.4	35.84	1.3	39.86
Average	2.4	52.50	2.8	52.99
Average without (*) sequences	2.6	53.31	2.9	52.46

encoder in the framework of the simultaneous encoding with the adapted AFQLD algorithm (right column). It can be seen that both CU splittings are quite similar.

Table 1 contains the results for RA configuration of the algorithm applied to the simultaneous scenario (with the improvements described in Sect. 4) and the original algorithm applied to the transcoding scenario [7]. Sequences marked with (*) are those used in the training stage and the average is presented for all the sequences and for all of them excluding the training ones.

It can be observed that the algorithm achieves a quantitative time reduction of around 52.50 % on average with a BD-rate penalty of 2.4 %, which proves the good trade-off between complexity reduction and encoding performance of the proposal. Moreover, the results show a similar time reduction compared to the transcoding scenario [7], while decreasing the BD-rate from 2.8 to 2.4 %, which represents a relative decrease of 14 %. Therefore, the global experimental results confirm that the proposed algorithm can reduce the computational complexity of the HEVC encoding stage by more than a half with a slight BD-rate increase, favoring the real-time software

and hardware implementation. It can also be noted that the results are similar for the training sequences and for those which were not used in the training stage.

Another way to evaluate the algorithm is by looking at the actual hit rate (that is, the rate of correctly classified CU splittings) achieved by the algorithm. The actual hit rate has been calculated for each QP and sequence, observing 93 % for CU size of 64×64 on average. The hit rate for CU size of 32×32 is 87 %, and for CU size of 16×16 is 92 %. The estimated hit rate using cross-validation on the models of CU sizes of 64×64 and 32×32 are 90 and 68 % on average, respectively, which indicates the benefit of evaluating some PUs even if the decision is to split.

6 Conclusions

This paper presents an approach for accelerating the CTU partitioning in the novel scenario of a heterogeneous simultaneous H.264/AVC and HEVC encoder which can be useful for video distribution companies. The splitting algorithm is a modification of the AFQLD algorithm, which is based on statistical NB classifiers. It decides on the most appropriate quadtree level by taking into account what has occurred in the H.264/AVC encoder, without the need for testing all the CUs/PUs. Obtained results show that HEVC encoding time is reduced more than a half with a negligible loss in BD-rate terms, which is even lower than the BD-rate obtained in the transcoding scenario as it was expected given the improvements presented in this paper.

References

1. Adhikari VK, Guo Y, Hao F, Varvello M, Hilt V, Steiner M, Zhang ZL (2012) Unreeling netflix: understanding and improving multi-CDN movie delivery. In: INFOCOM'12. IEEE, pp 1620–1628
2. Bjontegaard G (2008) Improvements of the BD-PSNR model. ITU-T SG16 Q 6:35
3. Bossen F (2013) Common HM test conditions and software reference configurations. In: Proc. 12th JCT-VC meeting, Doc. JCTVC-L1100
4. Bossen F, Bross B, Suhring K, Flynn D (2012) HEVC complexity and implementation analysis. IEEE Trans Circuits Syst Video Technol 22(12):1685–1696
5. Chen J, Boyce J, Yan Y, Hannuksela M, Sullivan G, Wang Y (2014) Scalable high efficiency video coding draft 7. In: JCTVC-R1008, 18th JCTVC meeting, Sapporo
6. De Praeter J, Diaz-Honrubia A, Van Kets N, Van Wallendael G, De Cock J, Van de Walle R (2015) Fast simultaneous video encoder for adaptive streaming. In: Proc. IEEE int. workshop multimedia signal process (MMSP), pp 1–6
7. Diaz-Honrubia AJ, Martinez JL, Cuenca P, Gamez JA, Puerta JM (2015) A statistical approach of a CTU splitting algorithm for a H.264/ AVC to HEVC video transcoder. In: 15th international conference on computational and mathematical methods in science and engineering, Rota
8. Espeland H, Stensland HK, Finstad DH, Halvorsen P (2012) Reducing processing demands for multi-rate video encoding: implementation and evaluation. Int J Multimedia Data Eng Manag (IJMDEM) 3(2):1–19
9. Fayyad UM, Irani KB (1993) Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the international joint conference on uncertainty in AI
10. Finstad D, Stensland H, Espeland H, Halvorsen P (2011) Improved multi-rate video encoding. In: Proc. IEEE int. symposium multimedia (ISM), pp 293–300
11. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
12. Han W, Min J, Kim I, Alshina E, Alshin A, Lee T, Chen J, Seregin V, Lee S, Hong Y, Cheon M, Shlyakhov N, McCann K, Davies T, Park J (2012) Improved video compression efficiency through

flexible unit representation and corresponding extension of coding tools. IEEE Trans Circuits Syst Video Technol 20(12):1899–1909
13. HM reference software. https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/. Accessed 1 Sept 2015
14. ITU-T Rec. H.264 and ISO/IEC 14496–10 (AVC) version 16: advanced video coding for generic audiovisual services (2012)
15. ITU-T Recommendation H.265 and ISO/IEC 23008-2 (Version 1): high efficiency video coding (2013)
16. Joint Collaborative Team on Video Coding: reference software to committee draft, version 18.4 (2012)
17. Ohm J, Sullivan G, Schwarz H, Tan TK, Wiegand T (2012) Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC). IEEE Trans Circuits Syst Video Technol 22(12):1669–1684
18. Peixoto E, Shanableh T, Izquierdo E (2014) H.264/AVC to HEVC video transcoder based on dynamic thresholding and content modeling. IEEE Trans Circuits Syst Video Technol 24(1):99–112
19. Schroeder D, Rehm P, Eckehard S (2015) Block structure reuse for multi-rate high efficiency video coding. In: Proc. IEEE int. conf. image process (ICIP), pp 3972–3976
20. Shen T, Lu Y, Wen Z, Zou L, Chen Y, Wen J (2013) Ultra fast H.264/AVC to HEVC transcoder. In: Data compression conference (DCC'13), Salt Lake City
21. Sullivan GJ, Ohm JR, Han WJ, Wiegand T (2012) Overview of the high efficiency video coding (HEVC) standard. IEEE Trans Circuits Syst Video Technol 22(12):1649–1668
22. Zaccarin A, Yeo BL (2002) Multi-rate encoding of a video sequence in the dct domain. In: Proc. IEEE int. symposium circuits syst. (ISCAS), vol 2, pp II-680–II-683

2.3 A Fast Hybrid Scalable H.264/AVC and HEVC Encoder

- **Title:** A Fast Hybrid Scalable H.264/AVC and HEVC Encoder
- **Authors:** Antonio Jesús Díaz-Honrubia, José Luis Martínez, and Pedro Cuenca
- **Type:** Journal
- **Journal:** The Journal of Supercomputing
- **Publisher:** Springer
- **ISSN:** 1573–0484
- **State:** Published online
- **Year:** 2016
- **DOI:** 10.1007/s11227-016-1802-z
- **URL:** http://link.springer.com/article/10.1007/s11227-016-1802-z
- **Category**: Computer science, hardware and architecture
- **Impact Factor:** 1.088
- **JCR ranking:** Q2

J Supercomput
DOI 10.1007/s11227-016-1802-z

A fast hybrid scalable H.264/AVC and HEVC encoder

A. J. Diaz-Honrubia[1] · **J. L. Martinez**[1] · **P. Cuenca**[1]

Abstract Over the past years, multimedia usage has changed dramatically, with networks and terminals of diverse bandwidths and capabilities coexisting, making an adaptability of the video stream necessary. By the use of video scalability schemes, video streams would be able to adapt to these heterogeneous networks and a wide range of terminals. Moreover, some devices may be able to decode a subset of all the available video standards, e.g. most devices can decode the well-known H.264/advanced video coding (AVC) standard, which has dominated the market for the past 10 years. However, more recent devices can take the advantage of more modern standards whose compression performance is much higher, such as high-efficiency video coding (HEVC). This problem can be solved by the use of hybrid scalability, which allows the use of H.264/AVC for the base layer and HEVC for the enhancement layers. However, scalable video coding is very computationally expensive, so acceleration techniques are of great help in this kind of encoders. This paper presents a fast inter prediction algorithm which makes use of information from H.264/AVC base layer encoding and

This work has been jointly supported by the MINECO and European Commission (FEDER funds) under the project TIN2015-66972-C5-2-R. This work has also been supported by the Spanish MECD under scholarship FPU 12/00994.

✉ A. J. Diaz-Honrubia
Antonio.DHonrubia@uclm.es

J. L. Martinez
JoseLuis.Martinez@uclm.es

P. Cuenca
Pedro.Cuenca@uclm.es

[1] High Performance Networks and Architectures Laboratory, University of Castilla-La Mancha, Albacete, Spain

Published online: 02 July 2016

uses it to make faster decisions in HEVC. Experimental results show that the proposed algorithm can achieve a good tradeoff between coding efficiency and complexity.

Keywords HEVC · H.264/AVC · Scalable Video · CTU Splitting

1 Introduction

The H.264/advance video coding (AVC) standard [14] has been the most widely used video compression standard over the past few years. However, in April 2013, the Joint Collaborative Team on Video Coding (JCT-VC) released the high-efficiency video coding (HEVC) standard [15]. The main goal of this new standard was not only to achieve an efficient performance for delivering high-quality multimedia services over bandwidth-constrained networks, but also to give support to ultra high definition (UHD), which demands a high bandwidth. In terms of rate-distortion (RD) performance, HEVC roughly doubles the compression performance of H.264/AVC, but at a cost of extremely high computational requirements during encoding [18].

On the one hand, new video-enabled devices (tablets, smartphones, …) have emerged in the market. The requirements and constraints of these newer devices have changed the way video streaming is demanded. In particular, the newest ones are able to decode HEVC streams, taking advantage of its superior compression performance. Older devices, however, are limited to decoding H.264/AVC streams.

On the other hand, taking into account the characteristics and the possible congestion of the networks, a media communication system should be able to adapt the media streams to the transmission constraints to ensure the continuity of acceptable quality images [24]. Scalable HEVC (SHVC) [6] is a powerful tool when dealing with either changing network conditions or devices with different capabilities since video sequences encoded with this extension of the HEVC standard can easily be adapted to the changing requirements. The previous scalable standard, i.e. H.264/scalable video coding (SVC) [14], already supported three different types of scalability: spatial, temporal and quality scalability. These scalabilities can be used for bit stream adaptation based on user and network capabilities. This is possible because the SVC and SHVC bit streams are organized in layers (one base layer and one or more enhancement layers). The base layer represents the lowest frame rate, resolution and/or quality and every enhancement layer increments frame rate, resolution and/or quality. By removing enhancement layers from a encoded bit stream, an adaptation to the channel bandwidth or to the device is achieved.

With the advent of SHVC a new type of scalability, so called hybrid scalability [5], was introduced. This new scalability allows the base and the enhancement layers to be encoded in different standards. This is usually implemented by combining an H.264/AVC stream for the base layer and HEVC streams for the other layers. Then, a stream can combine some of these types of scalability. In the specific framework of this paper, quality and hybrid scalabilities have been chosen. Thus, when the network is congested or when the receptor is a non-HEVC capable device, only the base layer, which is encoded with H.264/AVC using a higher quantization parameter (QP) (lower quality) than the other layers, is received and/or decoded. When neither the network

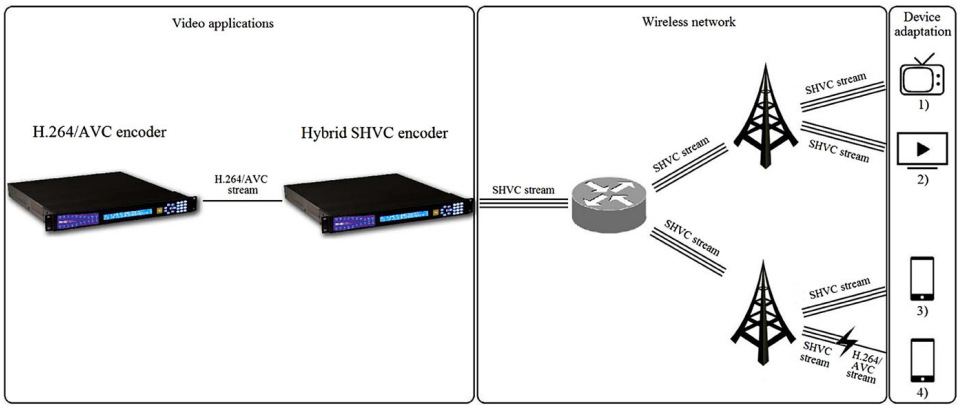


Fig. 1 Hybrid SHVC in a wireless network scenario

nor the receiving device is a constrain, a higher quality is received with the HEVC layer.

A lot of research has been done on scalable video transmission over wireless access networks, including broadband 4G networks, owing to its several advantages over conventional video coding. Scalable video in general, and SHVC in particular, avoids the problem of simulcasting fixed profile video streams at different bit rates, standards, resolutions or temporal profiles by embedding a base low-resolution stream in a hierarchical stream consisting of several differential enhancement layers. Another significant advantage of SHVC over conventional video coding is the graceful degradation of video quality in the event of packet drops due to network congestion. Figure 1 shows the work flow of the described scenario, where device 1) is an older device non-HEVC-capable device (but H.264/AVC-capable) and all the other devices are HEVC-capable devices. Then, devices 1) and 4) only show the H.264/AVC base layer (in the first case because of the decoding capability of the device, and in the second case because of network congestion), while devices 2) and 3) show the HEVC enhancement layer with higher quality.

Given the complexity increase of using HEVC compared to H.264/AVC, and the use of inter-layer references to reduce the bandwidth in SHVC, efficient encoding algorithms are required. This paper presents an algorithm which makes use of the information gathered during the encoding of the H.264/AVC base layer and reuses it in the HEVC layers; thus, while the H.264/AVC encoder for the base layer remains unaltered, the SHVC encoder for the enhancement layers, which is substantially more complex, is accelerated. The proposal uses a soft computing algorithm which makes use of Naïve-Bayes (NB) classifiers to avoid an exhaustive search over all the possible CU sizes of HEVC. In an off-line data mining process, all the knowledge needed is fetched from the H.264/AVC layer of some training sequences and is then converted into mathematical models by means of machine learning (ML) techniques.

This solution provides support for a wide range of devices, while the overall complexity of the SHVC encoder is reduced with the use of the proposed algorithm. This work can be considered as an extension of the paper presented in [8], since that paper proposed a fast intra algorithm for the H.264/AVC to HEVC transcoder, while this

paper also presents a fast algorithm changing the scenario from intra to inter prediction and from transcoding to hybrid SHVC.

The remainder of this paper is organized as follows: Sect. 2 includes technical background to the HEVC standard and the SHVC extension, while Sect. 3 details the related work. Section 4 describes the proposed algorithm. Experimental results are given in Sect. 5, and Sect. 6 concludes the paper.

2 Technical background

Even though HEVC is based on the same hybrid (spatio-temporal) redundancy block structure as previous video coding standards, such as H.264/AVC, there are important novelties and improvements with respect to it [18]. H.264/AVC was based on a MacroBlock (MB) structure, with all of these MBs having the same 16 × 16 pixels area. These MBs could be split into several sub-MBs, which was the basic unit to carry out the inter or intra prediction. Meanwhile, the transform could be applied to 16 × 16 or 8 × 8 blocks.

However, HEVC splits a frame in coding tree units (CTUs) [13], whose usual size is 64 × 64 (they can be smaller). Then, three new concepts are introduced: Coding Unit (CU), Prediction Unit (PU) and Transform Unit (TU). Every CTU is composed of one or more CUs, which can be recursively split into four sub-CUs using a quadtree structure (see Fig. 2). Thus, the CU size ranges from 64 × 64 to 8 × 8 pixels.

At the same time, each CU is composed of one or more PUs and one or more TUs, which are independent of each other. The PU is the unit in which the inter or intra prediction is performed and its size can be 2N × 2N, 2N × N, N × 2N, N × N, 2N × nU, 2N × nD, nL × 2N, nR × 2N. The last four PUs are called asymmetric PUs, which are a new type of prediction block not used in previous standards. Meanwhile, the transform is applied at TU level, whose size can vary from 32 × 32 to 4 × 4.

Therefore, the rate distortion optimization (RDO) model is much more complex than the RDO model in previous standards due to the high number of CU, PU and TU combinations [4]. Moreover, HEVC also introduces other new tools. e.g., the sample

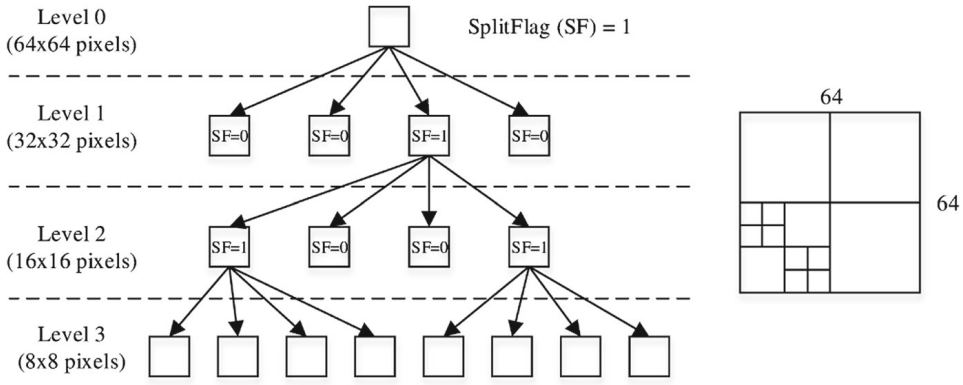


Fig. 2 Quadtree CTU splitting illustration

adaptive offset (SAO) is applied to the reconstructed frames after the deblocking filter, aiming to improve the perceptual quality [22].

The main objective of SHVC is not to create a different bit stream for each type of device (TVs, tablets, smartphones, …). In our scenario, first a base layer is encoded in H.264/AVC that consists of a low-quality version of the input video stream, and then, on top of this base layer, HEVC enhancement layers are encoded. These enhancement layers allow those devices that receive and are able to decode more than the bare minimum to improve video quality, in accordance with their capabilities. Four types of scalability are supported (namely spatial, temporal, quality and hybrid). On the one hand, hybrid scalability allows the video stream to be backward compatible with older devices by the use of different standards for the base and the enhancement layers.

On the other hand, quality scalability makes it possible to adapt the quality of the stream by choosing a different QP for each layer. Then, frames in the base layer are used as reference frames for frames in the enhancement layers. Given that the frame in all layers are the same frame with different quality, there is a high probability of finding a good matching during the motion estimation stage. Moreover, enhancement layers do not contain I frames, since even though frames which are co-located to an I frame in the base layer will be P frames in the enhancement layers. These facts cause a large decrease of the bit rate in the enhancement layers. For a comprehensive overview of SHVC, the reader is referred to [6].

3 Related work

As far as the authors of this paper know, there are not many previous works which deal with the acceleration of SHVC. In particular, no work which deals with the acceleration of hybrid SHVC. In [1], the authors propose a fast mode decision for quality scalability in SHVC based on the observations made when they compare the base and the enhancement layers. Specifically, they observe that a 31.7 % of CUs in the enhancement layers have the same depth as the co-located CUs in the base layer, that intra PUs are not usually chosen in the enhancement layers and that irrespective of whether a vertical PU was chosen in the base layer or not, it is very likely to choose a vertical PU in the co-located block in the enhancement layers too, and vice versa. The results show that on average for the used sequences, the authors obtain a 47 % time reduction in the whole SHVC encoder with a 0.9 % BD-rate [2], which measures the increment of bit rate while maintaining the same objective quality.

In [23], a similar approach is followed but in the intra prediction case. The authors skip some depths of the quadtree according to the observations of the behaviour of the same depth CUs in the enhancement layer given the depth of the base layer. Moreover, they choose an intra prediction directional mode according to neighbour PUs. The results show that the authors obtain a 61 % time reduction with a 0.3 % BD-rate. However, they do not detail if the time reduction is measured only for the enhancement layer or for the whole SHVC encoder and they do not specify the separate gain of the quadtree limitation and the intra direction selection.

However, it must be taken into account that the observations made in these two papers cannot be used since in our scenario, since the base layer is not encoded

using HEVC, but H.264/AVC, making it more difficult to make a prediction, since the block sizes are not the same and there are too many significant changes between the standards. Moreover, the proposal of [23] combines the quadtree limitation with the intra direction selection, which cannot be compared with an inter proposal such as the one presented in this paper.

There are several works which deal with the objective of accelerating HEVC according to information fetched from H.264/AVC. It can be interesting to take these proposals into account, since even though they cannot be compared with the scenario of this paper (they are not designed for scalable video), some ideas can be used. In [8], the same authors of this paper propose a fast intra mode decision based on the information of the intra modes of H.264/AVC. However, as the block sizes are very different, they follow a MacroBlock (MB) clustering approach according to CU sizes. This clustering approach will also be the strategy followed in this paper to accommodate the difference between block sizes.

In [9] a CU splitting algorithm for an H.264/AVC to HEVC video transcoder is proposed. This work is based on the use of *Bayesian classifiers* which are trained using information from H.264/AVC as features and the HEVC CU splitting decisions for each CU depth as class variables. This kind of classifiers will be used in this paper to predict the CU depth too. Since the classifiers used in that work only use information from the decoder, the work in [7] extends the model in order to use information from encoder specific information, since the simultaneous encoding framework of this paper allows it, improving the results with respect to using only decoding information. Thus, as the H.264/AVC encoding information is also available in the hybrid SHVC scenario, this advantage will also be used in this work.

A last type of scenario is where H.264/AVC is not used at all and the objective is to accelerate the HEVC encoding with information gathered during the encoding itself or from observations made from the experience of previously encoded sequences. Based on the analysis of conditional probability values for Skip and Merge modes, a fast CU determination algorithm is proposed in [17], which obtains a 35 % time reduction with 0.8 % BD-rate.

4 Proposed fast hybrid SHVC algorithm

As discussed in Sect. 1, this work proposes an algorithm which aims to reduce hybrid SHVC computational complexity by trying to guess the quadtree depth for each CU in the HEVC enhancement layers. The algorithm can be applied to a different number of quadtree depths, since for each depth it decides if it is more likely to split the current CU (C_S), going a level deeper in the quadtree, or not (C_N), choosing the current depth as the maximum allowed. Thus, this problem is a binary classification problem.

Taking into account that a prediction may be wrong, when the decision is C_S at levels 0 or 1, Skip and 2N $\times$ 2N PUs are anyway checked. This checking lets the quadtree to go back if the lagrangian costs at deeper levels are worse than in upper levels. Moreover, all PUs are checked at level 2 since smaller CUs are more difficult to classify and this might cause a big RD performance drop. Otherwise, if the decision is C_N, the quadtree is not allowed to go deeper and all the PUs at this CU depth are

evaluated and the algorithm for this CU finishes. This process is schematically shown in Algorithm 1.

Algorithm 1 Proposed fact CTU splitting algorithm for SHVC.

```
if depth==3 then
  Try all possible PU sizes
  return best CU and PU combination in RD performance terms
else
  Choose C_S or C_N for this CU using the appropriate M_i model.
  if C_S then
    if depth==2 then
      Try all possible PU sizes
    else
      Try only Skip and 2N × 2N PU sizes
    end if
    Split the current CU into four sub-CU
    Recursively apply the algorithm for each sub-CUs
  else
    Try all possible PU sizes
    return best CU and PU combination in RD performance terms
  end if
end if
```

Thus, the different models described in Algorithm 1, $\mathcal{M}_l$, which make the splitting decisions at depth levels 0, 1 and 2, must be learnt. For that purpose, we have followed a data mining approach for bigger CU sizes (depth levels 0 and 1, what represents 64×64 and 32×32 CUs, respectively). Meanwhile, a straighter approach has been used at level 2 , since CU size at this level (16×16 pixels) is also the MB size in H.264/AVC. Thus, the proposed algorithm imitates H.264/AVC as described in Algorithm 2 by taking into consideration neighbouring blocks in some cases.

Algorithm 2 Splitting algorithm for quadtree level 2.

```
if H.264/AVC MB was Skip or 16×16 then
  Choose C_N
else if H.264/AVC MB was 16×8 or 8×16 then
  if All neighbouring H.264/AVC MB were Skip, 16×16, 16×8 or 8×16 then
    Choose C_N
  else
    Choose C_S
  end if
else
  Choose C_S
end if
```

Therefore, for CU sizes of 64×64 and 32×32 (depth levels 0 and 1), we must train classifiers using supervised learning. More specifically, a probabilistic approach has been chosen and the simplest bayesian classifier, which is the Naïve-Bayes classifier, has been selected. It calculates the posterior probability of each class C_i given an input

set of features $\mathbf{F} = \{w_1, \ldots, w_n\}$: $P(C_i|\mathbf{F}) \propto P(C_i) \prod_{j=1}^{n} P(w_j|C_j)$. Finally, the classifier selects the output class with the highest probability.

Moreover, the difference of QP between the layers is taken into account. Higher QPs tend to produce larger CU sizes, since when the quantization is too aggressive, the area looks smoother. Because of this, the splitting probability is scaled by a factor: $\tilde{P}(C_S|\mathbf{F}) = P(C_S|\mathbf{F}) \times QP_{EL}/QP_{BL}$, where QP_{BL} and QP_{EL} are the QP values in the base and the enhancement layers, respectively. Thus, if QP_{BL} is higher than QP_{EL}, then the scaled probability of splitting in the enhancement layer, $\tilde{P}(C_S|\mathbf{F})$, will be lower than $P(C_S|\mathbf{F})$.[1] Then, the scaled probability of not to split is calculated as $\tilde{P}(C_N|\mathbf{F}) = 1 - \tilde{P}(C_S|\mathbf{F})$.

Then, the first thing that must be done is to learn the classifiers for the different models. A total of 8 models have been considered according to the CU depth level (0 or 1) and the energy that the residue of the frame will have. For this energy, 4 levels have been considered (1, 2, 3 and 4), where 1 is the highest energy and 4 is the lowest energy. Therefore, a frame which is being encoded with the random access (RA) configuration can be assigned to a different energy level according to the hierarchical layer in the group of pictures (GOP) structure.

Each classifier has been learnt using *PeopleOnStreet*, *ParkScene*, *PartyScene* and *BQSquare* sequences. This represents one sequence per class (A, B, C and D), so that they are representative of a wide range of resolutions. Furthermore, 4 different QP values (22, 27, 32, 37) have been used. The initial 1000 CUs of each QP-sequence pair have been selected to learn the classifiers. The set of features, $\mathbf{F}$, have been extracted during the H.264/AVC base layer encoding, and these are calculated for the area covered (in MBs) by the current CU in the HEVC layers.

According to the previous experience of the authors, the following sets of features might be considered good predictors: (1) features which model the spatio-temporal complexity of a frame; (2) information from the encoding of the H.264/AVC base layer is available and it can save some calculations on HEVC layers; (3) the mean and the variance of the residue [11], as well as other statistical information, have been used in previous works; and (4) information calculated in a dynamical way in the HEVC layers can also be used.

According to the previous list, the initial set of features, $\mathbf{F}$, is composed of 53 variables. All the features can be fetched from the H.264/AVC base layer encoding (moreover the first 24 features could even be extracted from a H.264/AVC decoder). Nevertheless, the last 2 features are dynamically calculated during the encoding stage of the HEVC layers, where

- w_{QP}: QP value used to encode the base layer.
- w_{bits}: number of bits occupied by the encoded H.264/AVC MBs for the current CU area.binary arithmetic coding (CABAC) operation.
- w_{intra}, w_{skip}, w_{16}, w_4, w_{inter}: number of Intra, Skip, Inter 16×16, Inter 4×4 and other Inter MBs, respectively.
- w_{DCTno0}: number of DCT coefficients which are different of zero.
- w_{width}, w_{height}: width and height of the frame, respectively.

[1] If $\tilde{P}(C_S|\mathbf{F}) > 1.0$, then $\tilde{P}(C_S|\mathbf{F}) = 1.0$. If $\tilde{P}(C_S|\mathbf{F}) < 0.0$, then $\tilde{P}(C_S|\mathbf{F}) = 0.0$.

- w_{MVsum}: sum of all the x and y MV components in the frame.
- $w_{\text{MV}x\text{Mean}}$, $w_{\text{MV}y\text{Mean}}$, $w_{\text{MV}x\text{Var}}$, $w_{\text{MV}y\text{Var}}$: mean and variance of x and y MVs components, respectively, for the area covered.
- w_{resMean}, w_{resVar}: mean and variance of the residue, respectively.
- $w_{\text{resMeanSubCU1}}$, $w_{\text{resMeanSubCU2}}$, $w_{\text{resMeanSubCU3}}$, $w_{\text{resMeanSubCU4}}$: mean of the residue for each sub-CU (1, 2, 3 and 4).
- $w_{\text{resVarSubCUs}}$: variance of the 4 previous means.
- w_{sobelH}, w_{sobelV}: sum of the horizontal and vertical Sobel operators [19], respectively.
- $w_{\text{RDCostMode}[i]}$: RD cost of the i mode in H.264/AVC, where $i \in \{\text{Skip}, 16 \times 16, 16 \times 8, 8 \times 16, 8 \times 8, \text{Intra } 16 \times 16, \text{Intra } 8 \times 8, \text{Intra } 4 \times 4, \text{IntraPCM}\}$.
- $w_{\text{VarIntraDir8x8}}$, $w_{\text{VarIntraDir4x4}}$: variance of all Intra directions of Intra 8×8 and Intra 4×4 modes.
- $w_{\text{MeanMV}x[i]}$, $w_{\text{MeanMV}y[i]}$: mean of all x and y MV components, respectively, of each i mode, where $i \in \{16 \times 16, 16 \times 8, 8 \times 16 \text{ and } 8 \times 8\}$.
- $w_{\text{VarMV}x[i]}$, $w_{\text{VarMV}y[i]}$: variance of all x and y MV components, respectively, of each i mode, where $i \in \{16 \times 16, 16 \times 8, 8 \times 16 \text{ and } 8 \times 8\}$.
- w_{SkipCost}, $w_{\text{2Nx2NCost}}$: the HEVC Lagrangian cost of choosing Skip and 2N × 2N, respectively.

4.1 Data preprocessing

After the previous features have been calculated, the datasets should be subjected to a clean-up process to avoid some assumptions or overfitting. First of all, all the features are of a numerical nature but, to avoid the assumption that the probability of a feature given the class follow a normal distribution (or any other distribution), they have been discretized with the Fayyad–Irani algorithm [10], in which the intervals are chosen using the entropy: the resulting intervals are those which maximize the discriminative power regarding the class.

After that, a feature subset selection (FSS) algorithm [12] is used to remove those features which are redundant or those which do not model the problem correctly. Specifically, a greedy strategy with wrapper evaluation has been chosen, starting the process with an empty set and incorporating the best remaining feature iteratively. In the wrapper case, the best feature is considered to be the one that, when incorporated to the current subset, induces the classifier with the maximum accuracy.

The FSS process should use (if possible) the same algorithm that will be used to train the classifiers; in this case the NB algorithm has been used to evaluate the goodness of each subset. The FSS process finishes when the addition of a new feature does not improve the accuracy of the classifier.

Once the discretization and FSS have been applied, the 8 classifiers, for 2 CU sizes (i.e. 64×64 and 32×32) and 4 levels of energy, are learnt by the use of NB classifiers, finishing the offline training stage.

4.2 Online stage of the algorithm

After the 8 classifiers have been trained; an online stage of the algorithm is carried out for each HEVC layer during encoding time, in which a classifying threshold is learnt. The basic classification rule in a binary NB classifier is to choose C_S if $\tilde{P}(C_S) > \tilde{P}(C_N)$. However, the cost of making the error of not splitting when the standard HEVC decides the opposite should be more costly because, in other case, the speed drops but the quality of the image does not.

To take this fact into consideration, the classification rule is modified by adding misclassifying costs, i.e. choose C_S if $\tilde{P}(C_S) \times \text{Cost}_{SN} > \tilde{P}(C_N) \times \text{Cost}_{NS}$, where Cost_{SN} represents the cost of wrongly choosing C_S 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 with each particular cost.

$$\text{Cost}_{ij} = \left| L_j - L_i \right| \times \omega_{ij}, \tag{1}$$

In the transcoding scenario of [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 SHVC 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 $\omega_{NS} = 6.0$ and $\omega_{SN} = 1.0$.

5 Performance evaluation

In this section an evaluation of the proposed algorithm for accelerating the hybrid SHVC with quality scalability is performed. In order to ensure homogeneity among experiments, the common test conditions defined by the JCT-VC have been used [3]. The number of layers has been set out to two: the H.264/AVC base layer and one HEVC enhancement layer, using Random Access (RA) configuration for both of them (and enabling inter-layer references). It has been chosen to use only two layers since this is the most practical scenario, where some devices receive a lower quality stream and other devices receive a higher quality stream, since including more layers would result in a much higher overhead for those who will receive the best quality.

The QP values used were {22, 27, 32, 37} for the H.264/AVC base layer and {20, 25, 30, 35}, respectively, for the HEVC enhancement layer. The results are grouped according to their resolution in *Classes* from A to D (as shown in [3]):

- Class A (2560 × 1600 pixels): *Traffic*, *PeopleOnStreet*, *Nebuta*, *SteamLocomotive*.
- Class B (1920 × 1800 pixels): *Kimono*, *ParkScene*, *Cactus*, *BQTerrace*, *BasketballDrive*.
- Class C (832 × 480 pixels): *RaceHorsesC*, *BQMall*, *PartyScene*, *BasketballDrill*.
- Class D (416 × 240 pixels): *RaceHorses*, *BQSquare*, *BlowingBubbles*, *BasketballPass*.

The software used to encode the H.264/AVC base layer was JM 18.4 [16] and the software used to encode the HEVC enhancement layer SHM 6.1 was [20]. The parameters of the encoders which have not been mentioned here are kept as default in the configuration files. Thus, the process to obtain the results is the following:

1. First of all, encode the YUV file of the base layer with H.264/AVC reference software JM 18.4 using the aforementioned QP values, fetching all the features needed by the algorithm.
2. Second, encode the YUV file of the enhancement layer with SHVC reference software SHM 6.1 using the aforementioned lower QP values (*anchor*).
3. Then, encode the same YUV file of the enhancement layer with a modified version of SHM 6.1 which includes the proposed fast algorithm using the aforementioned QP values (*proposed*).
4. Finally, compare the *anchor* and *proposed* SHVC streams and get the time saving and the RD performance comparison (see bellow).

Measurements have been performed using a six-core Intel Core i7-3930K CPU running at 3.20 GHz. The results are shown using the *Time Reduction* and *BD-Rate* [2] (which measures the increment in bit rate while maintaining the same objective quality). These metrics are calculated as shown in (2), where t_{anchor} represents the execution time of the non-accelerated *anchor* SHVC encoder, and t_{proposed} is the execution time of the *proposed* SHVC encoder. The mean BD-rate is a weighted average of the luminance (Y) and chrominances (U and V) according to the guidelines given by the JCT-CV [21] and they are considered a good representation of the human visual system.

$$\begin{aligned} \text{Time reduction } (\%) &= (t_{\text{anchor}} - t_{\text{proposed}})/(t_{\text{anchor}}) \times 100 \\ \text{BD-rate } (\%) &= (6 \times \text{BD-rate}_{\text{Y}} + \text{BD-rate}_{\text{U}} + \text{BD-rate}_{\text{V}})/8 \end{aligned} \tag{2}$$

Table 1 shows the results of the proposed algorithm applied to the hybrid SHVC scenario using RA configuration. Those sequences which are marked with an asterisk $^{(*)}$ are the sequences which were used in the training stage. Thus, the average results are presented for all the sequences and for all the non-training sequences. As the final stream is composed of two layers and the objective is to evaluate the performance of those users who receive all the layers, the BD-rate has been calculated using the sum of the bit rates of both layers, while the PSNR has been taken from the HEVC layer, since that will be the layer shown to the user when the device allows it.

Table 1 Time reduction and BD-rate of the proposed algorithm for hybrid SHVC using RA configuration

	BD-rate (%)	Time reduction (%)
Class A		
Traffic	2.6	63.31
PeopleOnStreet(*)	1.2	44.26
NebutaFestival	1.2	51.31
SteamLocomotive	5.2	58.18
Class B		
Kimono	5.1	60.74
ParkScene(*)	2.5	62.48
Cactus	2.8	60.29
BasketballDrive	4.8	57.60
BQTerrace	2.4	65.49
Class C		
BasketballDrill	3.1	53.94
BQMall	2.7	56.40
PartyScene(*)	1.7	40.09
RaceHorsesC	2.7	42.36
Class D		
BasketballPass	2.5	42.17
BQSquare(*)	1.1	52.71
BlowingBubbles	1.7	45.35
RaceHorses	2.2	35.84
Average	2.6	60.11
Average without (*) sequences	3.0	60.20

The time reduction is calculated using the acceleration of the HEVC layer, which is the focus of the algorithm.

As can be observed, the proposed fast hybrid SHVC algorithm achieves a time reduction of around 60 % on average with a BD-rate loss of only 2.6 % (3.0 % when excluding the training sequences, but it still represents a low penalty for such a high time reduction). This fact confirms the good trade-off between acceleration and RD performance of the proposed fast encoding algorithm.

Hence, the global time reduction and BD-rate confirm that the proposed fast SHVC encoding algorithm reduces the computational complexity of the HEVC layers without a significant RD performance change, making it easier to implement a real-time software and hardware SHVC encoder. It can be seen in Table 1 that the results for the training sequences are similar than those for all the other sequences, except 3 sequences which make the results to worsen (*SteamLocomotive*, *Kimono* and *BasketballDrive*).

6 Conclusions

A fast hybrid SHVC with quality scalability has been presented in this paper. The stream is composed of an H.264/AVC base layers and one or more HEVC enhancement layers (in real scenarios it is usual that only one enhancement layer is used for quality scalability). This scenario is very useful in wireless networks and when there exists a wide range of different devices among which are included older devices.

The algorithm aims to stop the quadtree branching by the use of statistical NB classifiers based on the observations of some features fetched during the H.264/AVC base layer encoding or during the HEVC enhancement layers encoding. The results show that the proposed algorithm obtains a time reduction of around 60 % with a low degradation in RD performance terms, which has been measured by means of the BD-rate.

References

1. Bailleul R, Cock JD, Walle RVD (2014) Fast mode decision for snr scalability in SHVC digest of technical papers. In: IEEE International Conference on Consumer Electronics (ICCE), pp 193–194. doi:10.1109/ICCE.2014.6775968
2. Bjontegaard G (2008) Improvements of the BD-PSNR model, vol 6. ITU-T SG16 Q, p 35
3. Bossen F (2013) Common HM test conditions and software reference configurations. In: Proc. 12th JCT-VC Meeting, Doc. JCTVC-L1100, Geneve
4. Bossen F, Bross B, Suhring K, Flynn D (2012) HEVC complexity and implementation analysis. IEEE Trans Circuits Syst Video Technol 22(12):1685–1696
5. Boyce JM, Hong D, Jang W, Wenger S (2012) VPS support for out-of-band signaling and hybrid codec scalability. In: Proc. 11th JCT-VC Meeting, Doc. JCTVC-K0206
6. Boyce JM, Ye Y, Chen J, Ramasubramonian AK (2016) Overview of SHVC: scalable extensions of the high efficiency video coding standard. IEEE Trans Circuits Syst Video Technol 26(1):20–34. doi:10.1109/TCSVT.2015.2461951
7. Diaz-Honrubia AJ, De Praeter J, Van Wallendael G, Martinez JL, Cuenca P, Puerta JM, Gamez JA (2016) CTU splitting algorithm for H.264/AVC and HEVC simultaneous encoding. J Supercomput 1–13. doi:10.1007/s11227-016-1683-1
8. Diaz-Honrubia AJ, Martinez JL, Cuenca P (2016) Fast intra mode decision for an H.264/AVC to HEVC video transcoder. In: 16th International Conference on Computational and Mathematical Methods in Science and Engineering, Rota, Cadiz, Spain
9. Diaz-Honrubia AJ, Martinez JL, Cuenca P, Gamez JA, Puerta JM (2016) Adaptive fast quadtree level decision algorithm for H.264/HEVC video transcoding. IEEE Trans Circuits Syst Video Technol 26(1):154–168. doi:10.1109/TCSVT.2015.2473299
10. Fayyad UM, Irani KB (1993) Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the International Joint Conference on Uncertainty in AI
11. Fernandez-Escribano G, Kalva H, Cuenca P, Orozco-Barbosa L, Garrido A (2008) A fast MB mode decision algorithm for MPEG-2 to H.264 P-frame transcoding. IEEE Trans Circuits Syst Video Technol 18(2):172–185
12. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
13. Han W, Min J, Kim I, Alshina E, Alshin A, Lee T, Chen J, Seregin V, Lee S, Hong Y, Cheon M, Shlyakhov N, McCann K, Davies T, Park J (2012) Improved video compression efficiency through flexible unit representation and corresponding extension of coding tools. IEEE Trans Circuits Syst Video Technol 20(12):1899–1909
14. ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC) version 16 (2012) Advanced video coding for generic audiovisual services
15. ITU-T Recommendation H.265 and ISO/IEC 23008-2 (Version 1) (2013) High efficiency video coding

16. Joint Collaborative Team on Video Coding (2012) Reference Software to Committee Draft, version 18.4
17. Kim BG (2016) Fast coding unit (cu) determination algorithm for high-efficiency video coding (hevc) in smart surveillance application. J Supercomput 1–22. doi:10.1007/s11227-016-1730-y
18. Ohm J, Sullivan G, Schwarz H, Tan TK, Wiegand T (2012) Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC). IEEE Trans Circuits Syst Video Technol 22(12):1669–1684
19. Patnaik S, Yang YM (2012) Soft computing techniques in vision science, vol 395. Springer, Berlin
20. SHM 6.1 reference Software. https://hevc.hhi.fraunhofer.de/svn/svn_SHVCSoftware/tags/SHM-6.1/
21. Sullivan G, Minoo K (2013) Objective quality metric and alternative methods for measuring coding efficiency . In: Proc. 8th JCT-VC Meeting, San Jose, USA, JCTVC-H0012
22. Sullivan GJ, Ohm JR, Han WJ, Wiegand T (2012) Overview of the high efficiency video coding (HEVC) standard. IEEE Trans Circuits Syst Video Technol 22(12):1649–1668
23. Wang D, Yuan C, Sun Y, Zhang J, Zhou H (2014) Fast mode and depth decision algorithm for intra prediction of quality SHVC. In: Intelligent Computing Theory: 10th International Conference, ICIC 2014, pp 693–699. doi:10.1007/978-3-319-09333-8_75
24. Zhang H, Nguyen H, Martínez-Graciá E, Tudela-Solano PA, Zhang D, Crespi N, Guo B (2013) Scalable multimedia delivery with QoS management in pervasive computing environment. J Supercomput 65(1):317–335. doi:10.1007/s11227-011-0581-9

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

- **Title:** Reducing the Complexity of a Multiview H.264/AVC and HEVC Hybrid Architecture
- **Authors:** Antonio Jesús Díaz-Honrubia, Johan De Praeter, José Luis Martínez, Pedro Cuenca, and Glenn Van Wallendael
- **Type:** Journal
- **Journal:** Journal of Signal Processing Systems for Signal, Image, and Video Technology
- **Publisher:** Springer
- **ISSN:** 1939–8018
- **State:** Published online
- **Year:** 2016
- **DOI:** 10.1007/s11265-016-1156-z
- **URL:** http://link.springer.com/article/10.1007/s11265-016-1156-z
- **Category**: Engineering, Electrical & Electronic
- **Impact Factor:** 0.508
- **JCR ranking:** Q4

J Sign Process Syst
DOI 10.1007/s11265-016-1156-z

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

A. J. Díaz-Honrubia[1] · J. De Praeter[2] · J. L. Martínez[1] · P. Cuenca[1] · G. Van Wallendael[2]

Received: 30 December 2015 / Revised: 25 April 2016 / Accepted: 17 June 2016

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

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*

✉ A. J. Díaz-Honrubia
antonio.dhonrubia@uclm.es

J. De Praeter
Johan.DePraeter@ugent.be

J. L. Martínez
JoseLuis.Martinez@uclm.es

P. Cuenca
Pedro.Cuenca@uclm.es

G. Van Wallendael
Glenn.VanWallendael@ugent.be

[1] Albacete Research Institute of Informatics (I3A), University of Castilla-La Mancha, Albacete, Spain

[2] Ghent University - Data Science Lab, Ghent, Belgium

Published online: 30 June 2016

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

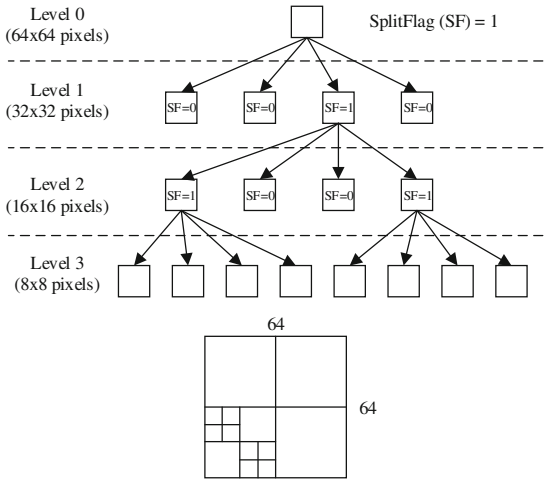


Figure 1 CTU splitting illustration.

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 (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.

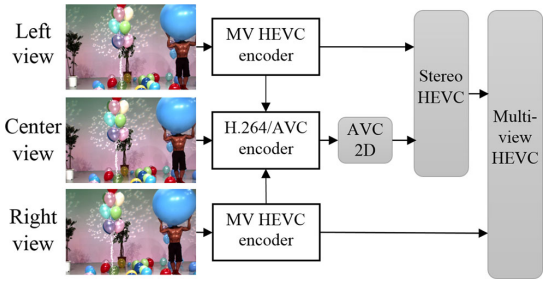


Figure 2 Hybrid multiview architecture.

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.

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
```

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 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 Eq. 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.

$$Cost_{ij} = \left|L_j - L_i\right| \times \omega_{ij}, \tag{1}$$

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 $\omega_{NS} = 6.0$ 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 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 Eq. 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).

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

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

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

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

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.

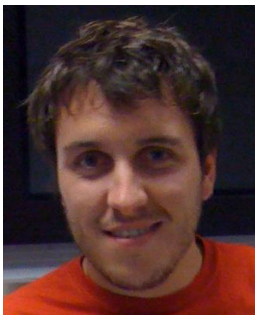
Acknowledgments This work has been jointly supported by the MINECO and European Commission (FEDER funds) under the projects TIN2012-38341-C04-04 and TIN2015-66972-C5-2-R. Likewise, this work has also been supported by the Spanish Education, Culture and Sports Ministry under grant FPU12/00994.

References

1. Aimar, L., Merritt, L., Glaser, F., Mitrofanov, A., & Gramner, H. (2013). x264 software. http://www.videolan.org/developers/x264.html.
2. Bjontegaard, G. (2008). Improvements of the BD-PSNR model. *ITU-T SG16 Q*, *6*, 35.
3. Bossen, F. (2013). Common HM test conditions and software reference configurations. In *Proc. 12th JCT-VC Meeting, Doc. JCTVC-L1100*.
4. Chen, J., Boyce, J., Yan, Y., Hannuksela, M., Sullivan, G., & Wang, Y. (2014). Scalable high efficiency video coding draft 7. In *JCTVC-R1008, 18th JCTVC Meeting*. Sapporo.
5. Chen, J., Boyce, J., Ye, Y., & Hannuksela, M.M. (2014). Scalable HEVC (SHVC) test model 6 (SHM 6). In *Proc. 17th JCT-VC meeting, Valencia, Spain, No. JCTVC-Q1007*.
6. Choi, K., Park, S.H., & Jang, E.S. (2011). CodingTtree pruning base CU early termination. In *JCT-VC document, JCTVC-F902*.
7. Diaz-Honrubia, A.J., Cebrian-Marquez, G., Martinez, J.L., Cuenca, P., Puerta, J.M., & Gamez, J.A. (2014). Low-complexity heterogeneous architecture for H.264/HEVC video transcoding. *Journal of Real-Time Image Processing*, 1–17. 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. (2015). 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). doi:10.1109/MLSP.2015.7324366.
9. Diaz-Honrubia, A.J., Martinez, J.L., Cuenca, P., Gamez, J.A., & Puerta, J.M. (2016). Adaptive fast quadtree level decision algorithm for h.264/hevc video transcoding. *IEEE Transactions on Circuits and Systems for Video Technology*, *26*(1), 154–168. doi:10.1109/TCSVT.2015.2473299.
10. Fayyad, U.M., & Irani, K.B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the international joint conference on uncertainty in AI* (pp. 1022–1027).
11. Fernandez-Escribano, G., Kalva, H., Cuenca, P., Orozco-Barbosa, L., & Garrido, A. (2008). 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.
12. Flores, J., Gámez, J.A., & Martínez, A.M. (2012). Supervised classification with Bayesian networks. In *Intelligent data analysis for real-life applications: theory and practice* (pp. 72–102).
13. Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, *3*, 1157–1182.
14. H.264, & I...R.I.T. (2009). Advanced video coding for generic audiovisual services. Annex H: Multiview Video Coding (MVC).
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 (2012). Reference software to committee draft, version 18. 4.
18. Muller, K., & Vetro, A. (2014). Common test conditions of 3DV core experiments. In *Proc. 7th JCT3V Meeting, Doc. JCT3V-G1100*.
19. Ohm, J., Sullivan, G., Schwarz, H., Tan, T.K., & Wiegand, T. (2012). 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.
20. Patnaik, S., & Yang, Y.M. (2012). *Soft computing techniques in vision science* (Vol. 395). Springer.
21. Sullivan, G.J., Ohm, J.R., Han, W.J., & Wiegand, T. (2012). Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *22*(12), 1649–1668.
22. Tech, G., Wegner, K., Chen, Y., Hannuksela, M., & Boyce, J. (2014). MV-HEVC draft text 9. In *JCT3V-I1002, 9th JCT3V Meeting*. Sapporo.
23. Van Leuven, S., Bruls, F., Van Wallendael, G., De Cock, J., & Van de Walle, R. (2012). Doc.MPEG-M23669: Hybrid 3D Video Coding. ISO/IEC JTC1/SC29/WG11 (MPEG).
24. Van Leuven, S., Van Wallendael, G., De Cock, J., Bruls, F., Luthra, A., & Van de Walle, R. (2012). Doc.MPEG-M24968: Overview of the coding performance of 3D video architectures. ISO/IEC JTC1/SC29/WG11 (MPEG).

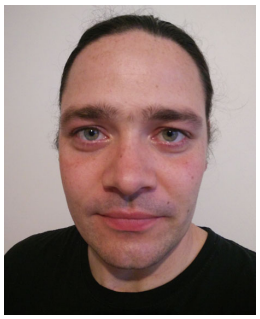
Antonio Jesús Díaz-Honrubia received the Bachelor's degree in Computer Science and Engineering, and the M.Sc. degrees in Advanced Computing Technologies and Computer Science and Engineering, from University of Castilla-La Mancha, Albacete, Spain, in 2011 and 2012, respectively, where he is currently working toward the Ph.D. degree in Advanced Computing Technologies. He is with University of Castilla-La Mancha, where he carries out his research activities with the Computer Architecture and Technology Group, Albacete Research Institute of Informatics. His research interests include video transcoding from H.264/AVC to the High Efficiency Video Coding standard and 3D video coding.

Johan De Praeter received the M.Sc. degree in Computer Science Engineering from Ghent University, Belgium, in 2013. Since then he joined Data Science Lab (formerly Multimedia Lab), Ghent University, which is also part of the independent research institute iMinds founded by the Flemish government. At Data Science Lab he is working towards a Ph.D. His research interests include video compression, High Efficiency Video Coding, machine learning, transcoding, and simultaneous encoding.



Jose Luis Martinez received his M.Sc. and Ph.D. degrees in Computer Science and Engineering from the University of Castilla-La Mancha, Albacete, Spain, in 2005 and 2009, respectively. After completing his Ph.D, he was postdoc researcher at the Centre for Communication System Research (CCSR), at the University of Surrey, (UK). In 2011, he joined the University of Castilla-La Mancha, where he is currently assistant professor. His research interests include video transcoding, scalable video coding and video applications for multicore and GPU architectures. He has also been a visiting researcher at the Florida Atlantic University, Boca Raton (USA) for 9 months. He has 26 publications in these areas in international refereed journals and 60 conference proceedings.

Pedro Cuenca received his M.Sc. degree in Physics (Electronics and Computer Science, award extraordinary) from the University of Valencia in 1994. He received his Ph.D. degree in Computer Engineering in 1999 from the Polytechnic University of Valencia. In 1995 he joined the Department of Computer Engineering at the University of Castilla-La Mancha (UCLM), where he is currently a Full Professor. He has also been a visiting researcher at Nottingham Trent University, University of Ottawa and University of Surrey. His research topics are centered in the area of video compression, QoS video transmission, and video applications for multicore and GPU architectures. He has published over 150 papers in international journals and conferences.



Glenn Van Wallendael obtained the M.Sc. degree in Engineering from Ghent University, Belgium in 2008. Afterwards, he worked towards a Ph.D. at Data Science Lab (formerly Multimedia Lab), Ghent University, with the financial support of the Agency for Innovation by Science and Technology (IWT). Currently, he continues working in the same group as a post-doctoral researcher. His main topics of interest are video compression including scalable video compression, transcoding, and Quality of Experience(QoE).

Chapter 3

Conclusions and Future Work

This chapter concludes the Thesis by presenting the conclusions that may be drawn from the works presented. Finally, some research lines to be explored in the future regarding the topic of the Thesis are discussed.

3.1 Conclusions

As was mentioned in Chapter 1, the new HEVC standard can be of great help in compressing video streams with demanding requirements, such as high resolution or frame rate. Furthermore, there are huge numbers of legacy sequences encoded with H.264/AVC, as well as legacy devices that cannot decode HEVC. Therefore, during the migration from one to the other standard, it can be very useful for HEVC-capable devices to take advantage of the superior compression performance of HEVC, while legacy devices should still be able to decode the stream (in H.264/AVC format).

Therefore, the *main conclusion* of the Thesis is that a fast H.264/AVC and HEVC collaborative video coding algorithm has been designed by accelerating the HEVC encoder in four different scenarios by means of the use of data fetched from the H.264/AVC stream (or during its encoding stage) and machine learning to make fast decisions during the HEVC encoding stage.

In order to accomplish this objective, the five goals which were set at the beginning of the Thesis have been achieved by performing a set of tasks for each of them. Firstly, a theoretical study of H.264/AVC and HEVC and their similarities and differences was carried out. In particular, a comparative analysis of the coding efficiency and encoding time was carried out. Furthermore, a study of different machine learning algorithms was carried out in order to choose the most appropriate one for our purpose.

After that, an algorithm for a fast H.264/AVC to HEVC heterogeneous transcoder, the so called AFQLD, was designed. This algorithm is based on deciding whether a node of the HEVC quadtree should be further split or not. In the case that it is split, all the PUs are checked at that level. Otherwise, most of the PUs are skipped in order to achieve a high acceleration. In order to take a decision, Naïve-Bayes classifiers were used at levels 0 and 1, while a mode copy strategy was followed at level 2.

This algorithm was designed in an incremental way, starting from a simpler algorithm, and improving it in several steps by the use of new information or more advanced techniques, such as a dynamic threshold in the classification rule. Moreover, the algorithm was designed so that the tradeoff between acceleration and coding performance can be chosen by the user, since it can be limited for application to a reduced number of depth levels of the quadtree.

The results of the final version of the AFQLD algorithm show that it is able to reduce the HEVC encoding time by 53.7% with a BD-rate of only 2.7% when using the *Random Access* configuration. However, if a lower bit rate is needed at the same objective quality, the user can choose to apply the algorithm only to depth levels 0 and 1, in which case the BD-rate decreases to 2.4%, achieving an acceleration of 41.2%. And if the applications requires even lower bit rate, applying the algorithm only at level 0 produces a BD-rate of only 0.5%, while the acceleration is 21.9%.

The high accuracy of this transcoder can be seen since, once more for the *Random Access* configuration, the hit rates at depth level 0, 1, and 2 are 93%, 87% and 92%, respectively. Moreover, when it is compared with ECU, it has been demonstrated that it is able to obtain a higher acceleration with a lower BD-rate.

Regarding the third goal of the Thesis, a fast simultaneous encoder has been designed by adapting the AFQLD algorithm to this scenario. For this purpose information from the H.264/AVC encoder has also been used. Moreover, the costs of the classification rule were adjusted. In this scenario, the acceleration of the HEVC encoder is the same as in the transcoding case, but with a lower BD-rate. For instance, when applying the algorithm to all depth levels and with the *Random Access* configuration, the BD-rate in this case is only 2.4%, that is, 0.3% less than in the transcoder in absolute terms (which equates to 12.5% in relative terms). The BD-rate reduction is due to the fact that more information is present in this scenario.

The fourth goal was achieved by the adaptation of the AFQLD algorithm to the hybrid SHVC scenario with quality scalability. Starting from the previous case, in which H.264/AVC was already present, the classification rule was adapted once more so that it takes into account the different QP values on each layer. More specifically, a weighting of the decision was performed, multiplying the probability of choosing to split by QP_{EL}/QP_{BL}, where QP_{EL} and QP_{BL} are the QPs of the enhancement and the base layers, respectively. The results show that an acceleration of more than 60% is achieved with a negligible penalty of only 2.6% in BD-rate terms.

Finally, the last secondary conclusion, associated with the fifth goal, is that the AFQLD algorithm can also be adapted to the hybrid MV-HEVC scenario. However, in this case the displacement between views must be considered when assigning an MB to a CU. In order to compensate for this difference, a fast displacement calculation is performed when assigning an MB to its collocated CU in HEVC. In this case the BD-rate is a bit higher: 4.8% in the 2-view case and 5.9% in the 3-view case. This increment is due to the displacement between

views, and that the two views might have different lighting and/or noise. However, the best acceleration is achieved in this case, reaching more than 70% on average with both 2 and 3-view cases.

3.2 Future Work

There are several ways in which the work presented in the Thesis can be extended. In the short term, we intend to implement a hybrid SHVC encoder with temporal scalability, using H.264/AVC for the base layer and HEVC for the remaining layers. As is usual in temporal scalability, and opposite to the case of quality scalability in which the same stream was encoded with different QPs in each layer, the different layers will contain different frames of the sequence, so that each layer may improve the overall frame rate, e.g. the base layer may be independently decoded with a frame rate of 15 *frames per second* (fps), the second layer may increase this by another 15 fps, making a total of 30 fps, and so on.

The problem in this case is that the frames in different layers are not the same, a fact that may produce errors when assigning an MB to a CU in HEVC due to movements. Furthermore, this case is even more complicated than the case of the hybrid MV-HEVC, since the displacements are not necessarily only in the horizontal direction, as happens in that case, in which the displacement is only caused by the difference between the position of the cameras. Conversely, in this scenario, the displacements are caused by objects or the camera moving between frames.

Another scenario that might be considered in the future is the last type of scalability, namely spatial scalability (different layers improve the resolution of the video). The problem in this case is that the size of the frames in each layer differs from the size in other layers. In this case, the MBs should be scaled properly, a fact that might even make it a simpler problem in some cases, since the area covered by an MB in the base layer might coincide with the area covered by a CU in HEVC (or, at least, the areas covered by them will be more equal). On the one hand, this makes the predictions easier for bigger CUs. However, on the other hand, they will be more difficult for smaller CUs.

Furthermore, in the case of extending the multiview scenario to 3D video, the fact that each view is accompanied by a depth map should be considered. In this case the particularities of the depth maps should be taken into account, these maps being very different from typical views, since the edges in the depth maps may mean a big change in depth, producing strange visual artifacts if these edges are distorted during encoding in the same way as in texture images.

Also in the framework of collaboration between H.264/AVC and HEVC, but not with the objective of accelerating an encoder, but to further compress the HEVC-encoded sequences, a perceptual transcoder might be considered. Perceptual video coding consists in taking into account not only the objective quality metrics, but also the subjective quality perceived by the

viewer. The main idea consists in considering that the viewer will not pay attention to some areas of the image, such as background. Therefore, the QP can be dynamically adjusted so that the background areas increase their QP value.

Thus, in a transcoding scenario, for example, where the objective is to decrease the bit rate of the sequence by migrating the sequence to a standard with a greater compression performance, it might be useful to employ perceptual transcoding, since the bit rate can be further reduced by the use of these techniques. As one of the problems in the said scenario is background identification, the already encoded sequence might provide some information, since, background areas tend to have smaller residue values or produce fewer DCT coefficients which are not zero.

Finally, the range can be extended by including other standards. For instance, the collaborative video coding in scenarios such as transcoding or simultaneous encoding can be considered between HEVC and VP9 [41] or the *Thor Video Codec* [4]. Moreover, the range can be extended even further if standards or codecs under development are considered, such as VP10 [14] or the standard which is thought to be the successor of HEVC, which is now in its firsts stages of development by the *Joint Video Exploration Team (on Future Video coding)* (JVET) [9].

Bibliography

[1] A. Aaron, R. Zhang, and B. Girod. Wyner-Ziv Coding for Motion Video. In *Proceedings of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, Pacific Grove, California, USA*, volume 1, pages 240–244, Nov 2002.

[2] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z.-L. Zhang. Unreeling Netflix: Understanding and Improving Multi-CDN Movie Delivery. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM), Orlando, Florida, USA*, pages 1620–1628, Mar 2012.

[3] G. Bjontegaard. Improvements of the BD-PSNR model. In *Proceedings of the VCEG-AI11, Doc. ITU-T SG16 Q.6, Berlin, Germany*, Jul 2008.

[4] G. Bjontegaard, T. Davies, A. Fuldseth, and S. Midtskogen. The Thor Video Codec. In *Data Compression Conference (DCC), Snowbird, Utah, USA*, Apr 2016.

[5] F. Bossen. Common HM Test Conditions and Software Reference Configurations. In *Proceedings of the 12th JCT-VC Meeting, Doc. JCTVC-L1100, Geneva, Switzerland*, Jan 2013.

[6] J. M. Boyce, D. Hong, W. Jang, and S. Wenger. VPS Support for out-of-Band Signaling and Hybrid Codec Scalability. In *Proceedings of the 11th JCT-VC Meeting, Doc. JCTVC-K0206, Shanghai, China*, Oct 2012.

[7] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

[8] G. Cebrian-Marquez, A. J. Diaz-Honrubia, J. De Praeter, G. Van Wallendael, J. L. Martinez, and P. Cuenca. A Motion Vector Re-Use Algorithm for H.264/AVC and HEVC Simultaneous Video Encoding. In *Proceedings of the 13th International Conference on Advances in Mobile Computing and Multimedia (MoMM), Brussels, Belgium*, pages 241–245, 2015.

[9] J. Chen, E. Alshina, G. Sullivan, J.-R. Ohm, and J. Boyce. Algorithm Description of Joint Exploration Test Model 3 (JEM3). In *Proceedings of the 3th JVET Meeting, Doc. JVET-C1001, Geneva, Switzerland*, May 2016.

[10] J. Chen, J. Boyce, Y. Yan, M. Hannuksela, G. J. Sullivan, and Y. K. Wang. Scalable High Efficiency Video Coding Draft 7. In *Proceedings of the 18th JCT-VC Meeting, Doc. JCTVC-R1008, Sapporo, Japan*, Jul 2014.

[11] J. Chen, B. J., Y. Ye, and H. M. M. SHVC Test Model 6 (SHM 6). In *Proc. 17th JCT-VC Meeting, Valencia, Spain, No. JCTVC-Q1007*, Mar 2014.

[12] K. Choi, S. Park, and E. S. Jang. Coding Tree Pruning base CU Early Termination. In *Proceedings of the 6th JCT-VC Meeting, Doc. JCTVC-F902, Torino, Italy*, Jul 2011.

[13] Cisco. Visual Networking Index: Forecast and Methodology, 2015–2020. Jun 2016.

[14] A. Converse, J. Bankoski, J. Han, Z. Liu, D. Mukherjee, H. Su, and Y. Xu. New Video Coding Tools Under Consideration for VP10. In *Proceedings of the VideoLAN Dev Days, Paris, France*, Sep 2015.

[15] A. Corrales-Garcia, J. L. Martinez, G. Fernandez-Escribano, and F. J. Quiles. Variable and Constant Bitrate in a DVC to H.264/AVC Transcoder. *Signal Processing: Image Communication*, 26(6):310–323, 2011.

[16] E. de la Torre, R. Rodriguez-Sanchez, and J. L. Martinez. Fast Video Transcoding from HEVC to VP9. *IEEE Transactions on Consumer Electronics*, 61(3):336–343, Aug 2015.

[17] J. De Praeter, A. J. Diaz-Honrubia, P. T., G. Van Wallendael, and L. P. Simultaneous Encoder for High-Dynamic-Range and Low-Dynamic-Range Video. *Under review for publication in IEEE Transactions on Consumer Electronics*, 2016.

[18] J. De Praeter, A. J. Diaz-Honrubia, N. Van Kets, G. Van Wallendael, J. De Cock, and R. Van de Walle. Fast Simultaneous Video Encoder for Adaptive Streaming. In *Proceeding of the 2015 IEEE International Workshop Multimedia Signal Processing (MMSP), Xiamen, China*, pages 1–6, Oct 2015.

[19] A. J. Diaz-Honrubia, G. Cebrian-Marquez, J. L. Martinez, P. Cuenca, J. M. Puerta, and J. A. Gamez. Low-complexity heterogeneous architecture for H.264/HEVC video transcoding. *Journal of Real-Time Image Processing*, 12(2):311–327, 2016.

[20] A. J. Diaz-Honrubia, J. De Praeter, J. L. Martinez, P. Cuenca, and G. Van Wallendael. Reducing the Complexity of a Multiview H.264/AVC and HEVC Hybrid Architecture. *Journal of Signal Processing Systems*, pages 1–10, 2016.

[21] A. J. Diaz-Honrubia, J. De Praeter, S. Van Leuven, J. De Cock, J. L. Martinez, and P. Cuenca. Using Bayesian Classifiers for Low Complexity Multiview H.264/AVC and HEVC Hybrid Architecture. In *Proceedings of the IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), Boston, Massachusetts, USA*, pages 1–6, Sep 2015.

[22] A. J. Diaz-Honrubia, J. De Praeter, G. Van Wallendael, J. L. Martinez, P. Cuenca, J. M. Puerta, and J. A. Gamez. CTU Splitting Algorithm for H.264/AVC and HEVC Simultaneous Encoding. *The Journal of Supercomputing*, pages 1–13, Feb 2016.

[23] A. J. Diaz-Honrubia, J. L. Martinez, and P. Cuenca. HEVC: A Review, Trends and Challenges. In *II Workshop on Multimedia Data Coding and Transmission, Elche, Spain*, Sep 2012.

[24] A. J. Diaz-Honrubia, J. L. Martinez, and P. Cuenca. Multiple Reference Frame Transcoding from H.264/AVC to HEVC. In *International Conference on Multimedia Modeling (MMM), Dublin, Ireland*, Jan 2014.

[25] A. J. Diaz-Honrubia, J. L. Martinez, and P. Cuenca. A Fast Hybrid Scalable H.264/AVC and HEVC Encoder. *The Journal of Supercomputing*, pages 1–14, 2016.

[26] A. J. Diaz-Honrubia, J. L. Martinez, and P. Cuenca. A Fast Intra H.264/AVC to HEVC Transcoding System. *Under review for publication in Multimedia Tools and Applications*, 2016.

[27] A. J. Diaz-Honrubia, J. L. Martinez, P. Cuenca, J. A. Gamez, and J. M. Puerta. A Data-Driven Probabilistic CTU Splitting Alogorithm for Fast H.264/HEVC Video Transcoding. In *Data Compression Conference (DCC), Snowbird, Utah, USA*, pages 449–449, Apr 2015.

[28] A. J. Diaz-Honrubia, J. L. Martinez, P. Cuenca, J. A. Gamez, and J. M. Puerta. A Statistical Approach of a CTU Splitting Algorithm for a H.264/AVC to HEVC Video Transcoder. In *15th International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE), Rota, Cadiz, Spain*, pages 449–460, Jul 2015.

[29] A. J. Diaz-Honrubia, J. L. Martinez, P. Cuenca, J. A. Gamez, and J. M. Puerta. 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, Jan 2016.

[30] A. J. Diaz-Honrubia, J. L. Martinez, P. Cuenca, and H. Kalva. A Fast Splitting Algorithm for an H.264/AVC to HEVC Intra Video Transcoder. In *Data Compression Conference (DCC), Snowbird, Utah, USA*, Apr 2016.

[31] A. J. Diaz-Honrubia, J. L. Martinez, J. M. Puerta, J. A. Gamez, J. de Cock, and P. Cuenca. Fast Quadtree Level Decision Algorithm for H.264/HEVC Transcoder. In *Proceedings of the IEEE International Conference on Image Processing (ICIP), Paris, France*, pages 2497–2501, Oct 2014.

[32] H. Espeland, H. K. Stensland, D. H. Finstad, and P. Halvorsen. Reducing processing demands for multi-rate video encoding: Implementation and evaluation. *International*

Journal of Multimedia Data Engineering and Management (IJMDEM), 3(2):1–19, 2012.

[33] U. M. Fayyad and K. B. Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *Proceedings of the International Joint Conference on Uncertainty in AI, Chambery, France*, Aug 1993.

[34] G. Fernandez-Escribano, J. Bialkowski, J. Gamez, H. Kalva, P. Cuenca, L. Orozco-Barbosa, and A. Kaup. Low-Complexity Heterogeneous Video Transcoding Using Data Mining. *IEEE Transactions on Multimedia*, 10(2):286–299, Feb 2008.

[35] G. Fernandez-Escribano, H. Kalva, P. Cuenca, L. Orozco-Barbosa, and A. Garrido. 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, Feb 2008.

[36] D. Finstad, H. Stensland, H. Espeland, and P. Halvorsen. Improved Multi-Rate Video Encoding. In *Proceedings of the IEEE International Symposium on Multimedia (ISM), Dana Point, California, USA*, pages 293–300, Dec 2011.

[37] J. Flores, J. A. Gámez, and A. M. Martínez. Supervised Classification with Bayesian Networks: a Review on Models and Applications. *Intelligent Data Analysis for Real-Life Applications: Theory and Practice*, pages 72–102, 2012.

[38] R. Garrido-Cantos, J. De Cock, J. L. Martinez, S. Van Leuven, and P. Cuenca. Motion-Based Temporal Transcoding from H.264/AVC-to-SVC in Baseline Profile. *IEEE Transactions on Consumer Electronics*, 57(1):239–246, Feb 2011.

[39] R. Garrido-Cantos, J. De Cock, J. L. Martínez, S. Van Leuven, P. Cuenca, and A. Garrido. H.264/AVC-to-SVC Temporal Video Transcoder for Video Broadcasting in Wireless Networks. *Multimedia Tools and Applications*, 75(1):497–525, 2016.

[40] Q. Ge and D. Hu. Fast Encoding Method Using CU Depth for Quality Scalable HEVC. In *Transactions of the IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), Ottawa, Ontario, Canada*, pages 1366–1370, Sep 2014.

[41] A. Grange, P. de Rivaz, and J. Hunt. *VP9 Bitstream and Decoding Process Specification*. Google, May 2016.

[42] I. Guyon and A. Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[43] R. H. Gweon and Y.-L. Lee. Early Termination of CU Encoding to Reduce HEVC Complexity. In *Proceedings of the 6th JCT-VC Meeting, Doc. JCTVC-F045, Torino, Italy*, Jul 2011.

[44] D. Heckerman. Bayesian Networks for Data Mining. *Data Mining and Knowledge Discovery*, 1(1):79–119, 1997.

[45] ISO/IEC JTC1/SC29/WG11 (MPEG). *MPEG-2: Generic Coding of Moving Pictures and Associated Audio Information. Document 13818-2*, Mar 1994.

[46] ITU-T Recommendation H.264 and ISO/IEC 14496-10 (version 10). *Advanced Video Coding for Generic Audiovisual Services*, Feb 2016.

[47] ITU-T Recommendation H.265 and ISO/IEC 23008-1 (Version 1). *High Efficiency Video Coding*, Apr 2013.

[48] ITU-T Recommendation H.265 and ISO/IEC 23008-2 (Version 2). *High Efficiency Video Coding*, Oct 2014.

[49] T. Katayama, W. Shi, T. Song, and T. Shimamoto. Low-Complexity Intra Coding Algorithm in Enhancement Layer for SHVC. In *Proceedings of the IEEE International Conference on Consumer Electronics (ICCE), Antalya, Turkey*, pages 419–422, Jan 2016.

[50] J. L. Martinez, G. Fernandez-Escribano, H. Kalva, W. A. C. Fernando, and P. Cuenca. Wyner-Ziv to H.264 Video Transcoder for Low Cost Video Encoding. *IEEE Transactions on Consumer Electronics*, 55(3):1453–1461, Aug 2009.

[51] K. McCann, C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. J. Sullivan. High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Encoder Description. In *Proc. 18th JCT-VC Meeting, Sapporo, Japan, No. JCTVC-R1002*, Jul 2014.

[52] J. Ohm, G. Sullivan, H. Schwarz, T.-K. Tan, and T. Wiegand. 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, Dec 2012.

[53] L. Pham Van, J. De Cock, A. Diaz-Honrubia, G. Van Wallendael, S. Van Leuven, and R. Van de Walle. Fast Motion Estimation for Closed-Loop HEVC Transrating. In *Proceedings of the IEEE International Conference on Image Processing (ICIP), Paris, France*, Oct 2014.

[54] Y. Piao, J. Min, and J. Chen. Encoder Improvement of Unified Intra Prediction. In *Proceedings of the 3th JCT-VC Meeting, Doc. JCTVC-C207, Guangzhou, China*, Oct 2010.

[55] S. L. Salzberg. C4.5: Programs for Machine Learning by J. Ross Quinlan. *Machine Learning*, 16(3):235–240, 1994.

[56] D. Schroeder, P. Rehm, and S. Eckehard. Block Structure Reuse for Multi-Rate High Efficiency Video Coding. In *Proceedings of the IEEE International Conference on Image Processing (ICIP), Quebec City, Quebec, Canada*, pages 3972–3976, Sep 2015.

[57] H. Schwarz, D. Marpe, and T. Wiegand. Analysis of Hierarchical B Pictures and MCTF. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), Toronto, Ontario, Canada*, pages 1929–1932, Jul 2006.

[58] T. Shanableh, E. Peixoto, and E. Izquierdo. MPEG-2 to HEVC Video Transcoding With Content-Based Modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(7):1191–1196, Jul 2013.

[59] J.-R. Sullivan, G. J. Ohm, W.-J. Han, and T. Wiegand. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, Dec 2012.

[60] T. K. Tan, R. Weerakkody, M. Mrak, N. Ramzan, V. Baroncini, J. R. Ohm, and G. J. Sullivan. Video Quality Evaluation Methodology and Verification Testing of HEVC Compression Performance. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(1):76–90, Jan 2016.

[61] G. Tech, Y. Chen, K. Müller, J. R. Ohm, A. Vetro, and Y. K. Wang. Overview of the Multiview and 3D Extensions of High Efficiency Video Coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(1):35–49, Jan 2016.

[62] A. M. Tourapis, A. Leontaris, K. Sühring, and G. J. Sullivan. *H.264/14496-10 AVC Reference Software Manual, version 18.4*. Joint Video Team, Jun 2012.

[63] C.-Y. Tsai, C.-Y. Chen, and Y.-W. Huang. SAO software cleanup and non-normative encoder only bug fixes. In *Proceedings of the 14th JCT-VC Meeting, Doc. JCTVC-N0230, Vienna, Austria*, Jul 2013.

[64] S. Van Leuven, F. Bruls, G. Van Wallendael, J. De Cock, and R. Van de Walle. Hybrid 3D Video Coding. In *Proceedings of the 99th MPEG Meeting, Doc. MPEG-M23669, San Jose, California, USA*, Feb 2012.

[65] A. Vetro, C. Christopoulos, and H. Sun. Video Transcoding Architectures and Techniques: an Overview. *IEEE Signal Processing Magazine*, 20(2):18–29, Mar 2003.

[66] J. Yang, J. Kim, K. Won, H. Lee, and B. Jeon. Early SKIP Detection for HEVC. In *Proceedings of the 7th JCT-VC Meeting, Doc. JCTVC-G543, Geneva, Switzerland*, Nov 2011.