UNIVERSITY OF CASTILLA-LA MANCHA

Computing Systems Department

Towards Fast HEVC Encoding Using Heterogeneous Architectures

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: Gabriel Cebrián Márquez
Advisors: Dr. Pedro Ángel Cuenca Castillo
Dr. José Luis Martínez Martínez

Albacete, March 2018

UNIVERSIDAD DE CASTILLA-LA MANCHA

Departamento de Sistemas Informáticos

Towards Fast HEVC Encoding Using Heterogeneous Architectures

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: Gabriel Cebrián Márquez
Directores: Dr. Pedro Ángel Cuenca Castillo
Dr. José Luis Martínez Martínez

Albacete, marzo de 2018

A Estefanía,
por compartir una vida juntos.

A mi familia,
por su inestimable cariño.

Acknowledgments

It has been a long journey. An adventure full of significant moments. A path that ends here, at the beginning of new experiences… but I would have been unable to come all this way without all the people who have supported me during these years. These lines are dedicated to all of them.

In the first place, I would like to thank Pedro and José Luis for the role they have played as advisors. Without them, I would have possibly never got to know the world of research. I am indebted to them for their guidance during all this time, and for having transmitted to me the values that characterize a good researcher. Even more than that, they have also been the people on whom I have been able to count when it came to taking decisions that involved academic and professional matters. They provided me with the best of advice at those moments when I was filled with doubts. I hope we can maintain close contact in the future.

It is also necessary to highlight the work of the teaching staff of the Faculty of Computer Science Engineering and the Albacete Research Institute of Informatics, who provided me with the knowledge that one day became the basis for this Doctoral Thesis.

Naturally, I would like to thank all my colleagues in the High-Performance Networks and Architectures group for being much more than that, for becoming the people with whom to drown my troubles, laugh and enjoy every single second inside and outside the laboratory. It has been a long period of time side by side, and I know that I will never forget these good moments. As often occurs in these cases, I will take the liberty of not mentioning any of you in particular, while not diminishing the importance of these words.

Similarly, I would also like to thank all the people who welcomed me in my research internships in Berlin and Ghent. The time I spent working in such beautiful places allowed me to meet not only excellent researchers, but also the wonderful people behind them. It was a privilege to exchange ideas, to contrast opinions and to discover new points of view. I sincerely hope we can meet again in the future.

The last words of these acknowledgments are reserved for the people closest to me; those people who stayed by me unconditionally. I shall begin by thanking my family. My brother, José Luis, for being a mentor to me. We may be as different as chalk and cheese, but I have always admired your determination and courage facing the world. My mother, Llanos,

for taking care of us, and for your unwavering love. We will never be able to thank you for everything you have given to us. Lastly, my father, José Luis, for making me someone better, although not even a shade of that wonderful person you were. You left a deep mark on all of us. But for me, you were always there.

I must not forget those people who, although not linked by blood ties, have treated me like part of their own family. Thank you, Antonio and Mª del Pilar, for having welcomed me into your home and treating me almost like a son.

And finally, I would like to dedicate these words to you, Estefanía. How could I describe in only these few words all that you have been to me in the past few years? I simply could not. For me, you have been the shoulder to lean on when I collapsed; the light in the shadows. You made me see that it was worth fighting when all seemed impossible. And, without even noticing, the two of us are about to become Doctors and to begin a new adventure… together, you and me. Thank you for letting me be part of your life.

This work was jointly supported by the Spanish Ministry of Economy and Competitiveness and the European Commission (MINECO/FEDER funds) under the projects TIN2012-38341-C04-04 and TIN2015-66972-C5-2-R, and by the Spanish Ministry of Education, Culture and Sports under the grant FPU13/04601.

Agradecimientos

Ha sido una larga aventura. Una travesía repleta de momentos. Un camino que finaliza aquí para dar comienzo a nuevas experiencias… pero no me hubiera sido posible llegar hasta este punto de no haber sido por todas las personas que me han apoyado durante estos años. Estas líneas van dedicadas a todas ellas.

En primer lugar, me gustaría agradecer a Pedro y José Luis su labor ejercida como directores de tesis a lo largo de estos años. Sin ellos probablemente nunca hubiera llegado a conocer el mundo de la investigación. A ellos les debo el haberme guiado durante todo este tiempo, y el transmitirme los valores que caracterizan a un buen investigador. Y más allá de eso, también han sido las personas con las que he podido contar en las decisiones que han involucrado mi carrera académica y profesional. Han sabido darme los mejores consejos en los momentos que más me asaltaban las dudas. Espero que podamos seguir mantieniendo un estrecho contacto en el futuro.

Es necesario resaltar también la labor del profesorado de la Escuela Superior de Ingeniería Informática y del Instituto de Investigación en Informática de Albacete, que de un modo u otro aportaron en su día los conocimientos que han hecho posible esta Tesis Doctoral.

Como no podía ser de otro modo, me gustaría agradecer a mis compañeros del grupo de Redes y Arquitecturas de Altas Prestaciones por ser mucho más que eso, por convertiros en las personas con las que ahogar penas, reir y disfrutar cada segundo trascurrido en el laboratorio y fuera del mismo. Ha sido un largo período de tiempo al lado de todos vosotros, y será muy complicado que estos buenos momentos desvanezcan en mi memoria. Como suele ser común en estos casos, me tomaré la licencia de no hacer mención de ninguno de vosotros en particular, sin que ello os quite peso en estas palabras dedicatorias.

Igualmente, también me gustaría agradecer con estas palabras a la gente de los grupos de Berlín y Gante que me acogió durante mis estancias de investigación. El tiempo que estuve trabajando en estos bellos lugares me permitió conocer no sólo a unos grandes investigadores, sino también a las maravillosas personas que se encuentran detrás. Fue un privilegio poder intercambiar ideas, contrastar opiniones y conocer nuevos puntos de vista. Espero que el tiempo nos lleve a encontrarnos de nuevo en el futuro.

Las últimas palabras he preferido reservarlas para las personas más cercanas; aquellas que siempre estuvieron a mi lado. Comenzaré por dar las gracias a mi familia. A mi hermano,

José Luis, por ser un referente para mí. Aunque seamos como la noche y el día, siempre he admirado tu determinación y valentía ante el mundo. A mi madre, Llanos, por cuidar de nosotros y por tu inquebrantable cariño. Jamás podríamos agradecerte todo aquello que has hecho por nosotros. Por último, a mi padre, José Luis, por convertirme en alguien mejor, si bien mínimo reflejo de la gran persona que tú fuiste. Dejaste una profunda huella en todos nosotros. Para mí, no obstante, nunca te llegaste a marchar.

No podría dejar de mencionar también a aquellas personas que, si bien no nos unen vínculos de sangre, también me han tratado como de su propia familia. Gracias, Antonio y Mª del Pilar por acogerme en vuestra casa como uno más y tratarme casi como a un hijo.

Por último, te dedico estas palabras a ti, Estefanía. ¿Cómo podría describir en tan sólo estas líneas todo lo que has sido para mí estos años? Sencillamente, no podría. Para mí has sido el hombro sobre el que apoyarme cuando me derrumbaba; la luz en las sombras. Me hiciste ver que valía la pena luchar cuando todo parecía imposible. Y, sin darnos cuenta, los dos estamos a punto de convertirnos en doctores y emprender una nueva aventura… juntos, tú y yo. Gracias por dejarme ser parte de tu vida.

Este trabajo ha sido cofinanciado por el Ministerio de Economía y Competitividad y la Comisión Europea (fondos MINECO/FEDER) bajo los proyectos TIN2012-38341-C04-04 y TIN2015-66972-C5-2-R, y por el Ministerio de Educación, Cultura y Deporte a través de la beca FPU13/04601.

Summary

In the past few years, society has witnessed a significant increase in the consumption of multimedia contents associated with the development of new technologies, such as smartphones and faster communication networks. Nowadays, this type of content represents more than half of all Internet traffic. Consumers, in turn, are increasingly demanding higher quality content and a better user experience. For this reason, the development of video coding standards has become essential to the continued improvement of the coding efficiency of the existing ones, and to offering the required encoding tools to support emerging formats such as ultra high definition (UHD) and high dynamic range (HDR) video.

With this aim, the Joint Collaborative Team on Video Coding (JCT-VC) developed the new High Efficiency Video Coding (HEVC) standard, which was released in early 2013. Compared with its predecessor, the H.264/MPEG-4 Advanced Video Coding (AVC) standard, HEVC is able to obtain a 50% bit rate reduction for the same subjective quality, which represents a significant improvement in terms of coding efficiency. These superior capabilities have attracted the attention of the companies in the sector, many of which have already implemented this standard in a wide variety of devices.

However, this increase in coding efficiency has been brought about by the introduction of new encoding tools and an improvement of the existing ones, which leads to a significant growth in the computational complexity of the encoding process in comparison with H.264/MPEG-4 AVC. This is the main limitation encountered by this standard when it is implemented in scenarios in which real-time encoding is a requirement. For this reason, the research community is making intense efforts in the development of tools and algorithms that enable a reduction in the processing time.

In line with the necessities of HEVC, the main objective of this Doctoral Thesis is thus to reduce the computational complexity of the encoder while maintaining the coding efficiency provided by the standard. In this regard, it is difficult to ignore the enormous computing power offered by current parallel architectures and, in particular, by the heterogeneous platforms composed of graphics processing units (GPUs). These devices achieve a much higher performance than a conventional computer, especially for tasks in which a reduced set of operations are performed on a large volume of data, such as in the case of video coding. Therefore, this type of architectures proves to be very well suited for this scenario, and may help reduce the processing time of some of the encoding tools.

To achieve its main objective, this Doctoral Thesis proposes the development of a pre-analysis stage that is able to estimate preliminary motion information from the input frames. This information is used later to accelerate the inter prediction, and the splitting and mode decision algorithms by means of a dynamic predictive model. The main aim of the proposed method is thus to accelerate those parts of the encoder that take the greatest encoding time, and as a result tackle the coding complexity of HEVC to a greater extent. In this context, the GPU architecture becomes the perfect way to carry out the calculation of the aforementioned motion information.

After a thorough evaluation of the proposed scheme, the results show that it is able to reduce the total encoding time by 71% on average at the expense of a negligible bit rate increase of 1.8% for the same objective quality. These results represent an excellent trade-off between the two parameters, outperforming the vast majority of related works in the literature. By using a more aggressive predictive model, the encoding time can be reduced by more than 83% at the cost of higher bit rates, achieving similar speed-ups to, and superior coding efficiency than, some of the best-known HEVC encoders on the market. Apart from these results, it is also worth highlighting the enormous versatility of the proposed scheme in terms of using the calculated motion information for other purposes, such as rate control or the establishment of the encoding pattern.

Resumen

En los últimos años, la sociedad ha presenciado un notable incremento en el consumo de contenidos multimedia asociado al desarrollo de nuevas tecnologías como los dispositivos móviles y las redes de telefonía. A día de hoy, este tipo de contenidos representa más de la mitad de todo el tráfico que circula por Internet. A su vez, los consumidores demandan cada vez contenido de mayor calidad y que provean una mejor experiencia de usuario. Por este motivo, se hace necesaria la elaboración de códecs que permitan mejorar la eficiencia de codificación de los ya existentes, a la vez que provean las herramientas de codificación necesarias para representar los nuevos formatos emergentes como Ultra High Definition (UHD) o High Dynamic Range (HDR).

Con este objetivo, el grupo Joint Collaborative Team on Video Coding (JCT-VC) desarrolló el nuevo estándar de codificación High Efficiency Video Coding (HEVC), finalizado a principios de 2013. Comparado con su predecesor, el estándar H.264/MPEG-4 Advanced Video Coding (AVC), HEVC es capaz de obtener una reducción en la tasa de bits de un 50 % para un mismo nivel de calidad subjetiva, lo que representa una mejora significativa en términos de eficiencia de codificación. Estas prestaciones han logrado atraer el interés de las empresas del sector, las cuales han implementado ya dicho estándar en muchos de sus dispositivos.

El incremento de la eficiencia de codificación de HEVC proviene, no obstante, de la introducción de nuevas herramientas de codificación y de la mejora de las ya existentes, lo que supone un aumento significativo de la complejidad computacional del proceso de codificación en comparación con H.264/MPEG-4 AVC. Esta es la principal limitación con la que se encuentra este estándar a la hora de su implementación en escenarios en los que se requiera alcanzar la codificación en tiempo real. Por este motivo, la comunidad investigadora se ha volcado en el desarrollo de herramientas y algoritmos que permitan reducir el tiempo de cómputo requerido por el codificador.

En línea con las necesidades impuestas por HEVC, el objetivo principal perseguido por esta Tesis Doctoral es, por tanto, el de reducir la complejidad computacional del codificador a la vez que se mantiene la eficiencia de codificación obtenida por el estándar. Para ello, resulta complicado ignorar la enorme potencia de cálculo provista por las arquitecturas paralelas actuales y, en particular, por aquellas arquitecturas heterogéneas compuestas por Graphics Processing Units (GPUs). Estos dispositivos son capaces de ofrecer un rendi-

miento muy superior al de un computador convencional, en especial en tareas en las que se realiza una misma serie de operaciones sobre un gran volumen de datos, como el caso de la codificación de vídeo. Por tanto, este tipo de arquitecturas parece ideal para integrar como parte del codificador y así reducir en gran medida el tiempo de cómputo de algunas de sus herramientas.

Para conseguir este objetivo, esta Tesis Doctoral propone el desarrollo de un algoritmo de pre-análisis capaz de calcular información preliminar de movimiento a partir de las imágenes de entrada, la cual es usada posteriormente para acelerar el proceso de inter-predicción, el particionamiento de bloques y la selección de modo a través de un modelo dinámico. Este algoritmo busca, por tanto, acelerar aquellas partes del codificador que suponen un mayor tiempo de cómputo, y así obtener la mayor aceleración posible. En este contexto, la arquitectura GPU representa el medio ideal para llevar a cabo el cálculo de esta información de movimiento.

Tras evaluar el esquema anteriormente propuesto, los resultados muestran que es capaz de reducir el tiempo total de codificación en un 71 % a costa de incrementar la tasa de bits de la secuencia en tan sólo un 1.8 % para una misma calidad objetiva. Estos valores representan un equilibrio excelente entre ambos parámetros, superando a la gran mayoría de trabajos relacionados en la literatura. Con el empleo de un modelo predictivo más agresivo, el tiempo de codificación puede verse reducido más de un 83 % a costa de un mayor incremento de la tasa de bits, alcanzando una aceleración similar y una eficiencia de codificación muy superior a la de algunos de los codificadores de HEVC más conocidos del mercado. Además de estos resultados, también resulta necesario señalar la gran versatilidad introducida por este esquema al poder utilizar la información de movimiento calculada para muchos otros propósitos, como el control de tasa o la configuración del patrón de codificación.

Contents

List of Figures

List of Tables

List of Acronyms

AI	All-Intra Configuration
AMVP	Advanced Motion Vector Prediction
AVC	Advanced Video Coding
BD	Bjøntegaard Delta
CBF	Coded Block Flag
CORE	Computing Research & Education
CPU	Central Processing Unit
CTU	Coding Tree Unit
CU	Coding Unit
CUDA	Compute Unified Device Architecture
GOP	Group of Pictures
GPU	Graphics Processing Unit
HD	High Definition
HDR	High Dynamic Range
HEVC	High Efficiency Video Coding
HM	HEVC Test Model
HPC	High-Performance Computing
IEC	International Electrotechnical Commission
IF	Impact Factor
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union — Telecommunication Standardization Sector
JCR	Journal Citation Reports

JCT-VC	Joint Collaborative Team on Video Coding
ME	Motion Estimation
MPEG	Moving Picture Experts Group
MV	Motion Vector
PSNR	Peak Signal-to-Noise Ratio
PU	Prediction Unit
QP	Quantization Parameter
R-D	Rate-Distortion
RA	Random Access Configuration
RDO	Rate-Distortion Optimization
SAD	Sum of Absolute Differences
TR	Time Reduction
TU	Transform Unit
UHD	Ultra-High Definition
VCEG	Video Coding Experts Group
VR	Virtual Reality
WPP	Wavefront Parallel Processing

CHAPTER 1

Introduction

This chapter introduces the main motivation for this Doctoral Thesis. We present the current situation in the field of video coding, define the problem, and justify the development of this work. An overview of the proposed schemes is also shown, pointing out their contribution to the state of the art. Finally, we analyze the results derived from this research.

1.1 Motivation and Justification

More than a couple of decades ago, our society witnessed enormous progress in the development of new technologies. Analog formats and devices were progressively replaced by their respective digital alternatives, offering numerous advantages such as ease of storage and transmission. This transition affected every aspect of technology, including the multimedia field. In this regard, it is important to mention the significant role played by the so-called video encoders, which enabled the conversion of an input video signal into a bit stream that could be easily transferred in digital format to a video decoder, which performed the opposite operation. It was the beginning of the digital era.

Since then, a large number of on-line video services have emerged. This, along with the advent of devices such as smartphones and tablets, resulted in a significant growth in the consumption of multimedia contents over the Internet. Moreover, the market is constantly demanding a greater quality of experience and new formats. This, however, entails two major problems that need to be solved. Firstly, contents of higher quality require a greater number of bits to be represented, and thus a larger storage space, and secondly, current communication networks do not evolve at the same speed as these demands, which may limit the available bandwidth for video streaming. In fact, video traffic represented 73% of all the IP traffic in 2016, and this figure is expected to increase up to 82% by the year 2021 [1].

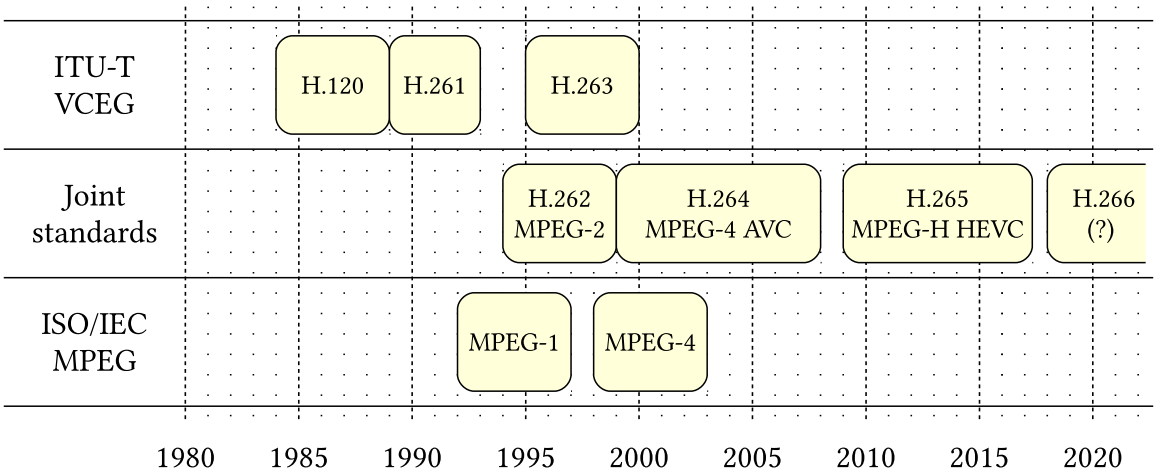


Figure 1.1: Timeline of the history of video coding standards.

To address these problems, different standardization organizations have played a crucial role over the years in the definition of video coding standards, which specify the syntax of the encoded bit streams. The standards enable not only the interoperability between devices of different manufacturers, but also the compression of the multimedia contents. The most significant groups committed to this task are the Moving Picture Experts Group (MPEG), which belongs to the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC), and the Video Coding Experts Group (VCEG), which is part of the International Telecommunication Union — Telecommunication Standardization Sector (ITU-T). The timeline of the history of the standards developed by these two groups is shown in Fig. 1.1.

For more than ten years, H.264/MPEG-4 part 10 — Advanced Video Coding (AVC) has been the most widely used standard at all levels [2]. Compared with previous standards, it defines a great number of new encoding tools, and provides support for high-definition (HD) content. Additional extensions were also defined for the standard, such as the scalable and the multiview video coding extensions. Nevertheless, as mentioned above, the market is demanding greater picture quality, and companies are exploring new ways of entertainment such as virtual reality (VR) and 360-degree video. The H.264/MPEG-4 AVC standard, however, is not able to provide support for this new generation of video coding.

In early 2013, the ISO/IEC MPEG and the ITU-T VCEG, which had formed a joint working group called the Joint Collaborative Team on Video Coding (JCT-VC), released the High Efficiency Video Coding (HEVC) standard [3]. Conceived as the successor to H.264/MPEG-4 AVC, HEVC was specifically developed by taking into account the particularities of the ultra-high definition (UHD) resolution (3840×2160 pixels), also called 4K, and beyond, such as 8K resolution (7680×4320 pixels). It was also designed to support additional color formats and high dynamic range (HDR), scalable coding, multiview coding, screen content coding, 360-degree videos, and several other applications through the definition of additional extensions.

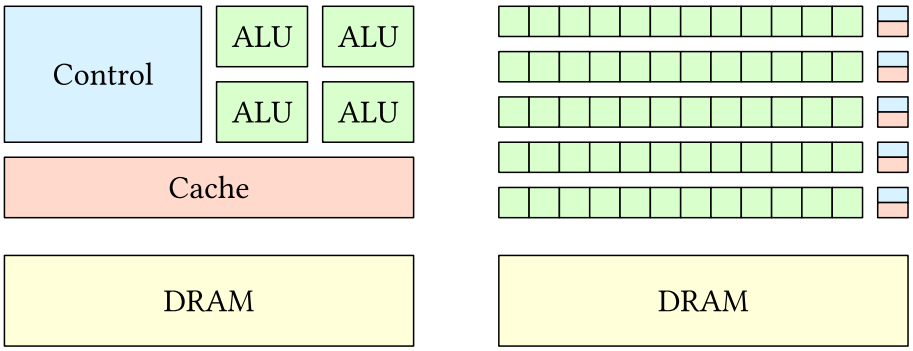
As a result of the introduction of new encoding tools in the standard and the improvement of the existing ones, HEVC is able to outperform H.264/MPEG-4 AVC in terms of coding efficiency, achieving a 50% bit rate reduction for the same perceptual quality or, in an equivalent way, a 36.9% reduction for equal peak signal-to-noise ratio (PSNR) [4]. Therefore, apart from supporting new formats and resolutions, this standard also achieves the target bit rate reduction required by current consumer trends. All these benefits have made the companies in the sector show significant interest in HEVC. In fact, many of them have already implemented this standard in their products, especially the decoding part.

While all these encoding tools have a positive effect on the coding efficiency, they also have an impact on the computational complexity of the standard. Compared with H.264/MPEG-4 AVC, it has been estimated that the encoding complexity of HEVC is three or four times higher [5]. This becomes a serious limitation when using this standard in certain scenarios in which real-time encoding is a requirement. In fact, the computational complexity of the encoding process has become one of the most significant constraints on the implementation of HEVC in current devices. This explains the importance of designing fast encoding algorithms for this standard, as it is the only way to ensure its feasibility and applicability. Accordingly, the research community is making intense efforts in this regard. In fact, the standard itself defines the high-level parallel techniques known as tiles and wavefront parallel processing (WPP) [6, 7].

Among all the novel encoding tools introduced in HEVC, it is worth highlighting the hierarchical block structure used to divide the pictures into partitions. This structure is called the coding tree unit (CTU), and can be subsequently divided into coding units (CUs), prediction units (PUs) and transform units (TUs) [8]. This wide range of possibilities provides the standard with great flexibility when adapting the partitions to the intrinsic features of the picture, which in turn results in higher coding efficiency. The process of determining the optimal partitioning requires, however, testing numerous block combinations. For this reason, many works have attempted to address the computational complexity of HEVC by omitting those combinations that are unlikely to provide the optimal results.

One of the first approaches to tackle this task involves evaluating only certain CUs of the CTU, while all the possible PU and TU combinations within these CUs are still taken into consideration. Accordingly, many works perform this selection on the basis of the statistical correlation found between the partitioning of previously coded blocks and the current block [9, 10]. Other works describe the existing relationship between the features of the pictures and the partitioning, in particular the residual generated, which enables the omission of certain CTU/CU combinations [11, 12, 13, 14]. The search for the optimal partitioning has also been considered as a classification problem. In this regard, the use of data mining techniques is present in many fast encoding methods [15, 16, 17, 18].

To reduce the encoding time further, the above approach can be complemented with an early skip detection method. The skip mode enables the efficient encoding of a CU by reusing information from neighboring blocks. The remaining modes in a CU are typi-



(a) CPU architecture. (b) GPU architecture.

Figure 1.2: Differences between the CPU and the GPU architectures.

cally omitted if this mode is selected, resulting in significant time savings. Similarly to the splitting algorithms above, the early skip decision can be taken on the basis of the spatial and temporal information obtained from neighboring blocks [19, 20, 21]. Likewise, the use of machine learning techniques is also common for this type of algorithms, in particular Bayesian models [22, 23].

While the early skip detection methods in combination with the CTU/CU splitting algorithms enable the omission of numerous CUs and PUs, a typical encoder still needs to evaluate all the prediction modes for those CUs in which the skip mode is not selected. For this reason, some works take a further step and propose a mode decision algorithm to avoid unnecessary PU evaluations. Once again, these works can be grouped into those that make use of the statistical information of neighboring blocks, and those that make use of machine learning techniques. In the former case, this information includes motion homogeneity, depth correlations, previously selected prediction modes, and the residual generated, among others [24, 25, 26, 27, 28]. In the latter case, the most common models are Bayesian rules and decision trees [29, 30].

Tackling the splitting and mode decision steps definitely contributes to a reduction in the encoding complexity, as these decisions are taken at a high level. When the evaluation of a PU is skipped, all the underlying operations such as prediction, transform and quantization are also skipped, and when a CU is omitted, the evaluation of all the PUs contained in it is omitted. Nevertheless, some encoding operations still consume a significant portion of the encoding time. In particular, motion estimation (ME) accounts for approximately 60% of the total encoding time in random access (RA) configurations [5]. The reason for the time spent on the execution of these algorithms is the large subset of samples over which they need to perform repetitive calculations. If this workload could be offloaded to a more specialized piece of hardware, the complexity of the encoder would be significantly reduced. This is precisely the scheme followed by the heterogeneous computing paradigm.

The term heterogeneous architecture usually refers to a system that is composed of more than one type of processor. Typical heterogeneous architectures include a general-

purpose processor and a specialized coprocessor, which very often are a central processing unit (CPU) and a graphics processing unit (GPU), respectively. GPUs are highly parallel, multithreaded, manycore processors with tremendous computational horsepower and very high memory bandwidth. These devices are designed in such a way that more transistors are devoted to data processing than to data caching and flow control, as shown in Fig. 1.2, which is especially suited for data-parallel computations. Therefore, operations such as ME, intra prediction, and filtering can take advantage of this type of platforms, achieving large speed-ups in comparison with normal architectures composed of a single CPU.

A few instances of the use of heterogeneous platforms can be found in H.264/MPEG-4 AVC. Some of them address the bidirectional inter prediction algorithm of the encoder, and achieve up to a 92% time reduction [31, 32]. With regard to HEVC, the design of algorithms targeted at GPUs proves more complicated because of the wide range of possibilities offered by the new partitioning scheme. Nevertheless, it is possible to find works that focus on the different modules of the encoder. With regard to intra prediction, some authors achieved the calculation of the optimal encoding mode on the device [33, 34, 35]. Similarly to H.264/MPEG-4 AVC, there are also works focused on inter prediction, mainly because of the total time spent on this encoding module, and its suitability for the GPU architecture [36, 37, 38]. Finally, it is worth highlighting the existence of works that address some minor encoding tools in terms of computational complexity, such as transform, quantization, and even deblocking filtering [39, 40, 41].

While it is clear that GPUs can provide notable time reductions for some encoding modules, algorithms based on this type of devices suffer from two major shortcomings that need be taken into consideration. Firstly, the effect of accelerating one encoding module on the total processing time depends on the percentage of time devoted to such a module. Given that the encoder is composed of several modules, the overall speed-up obtained may be limited. Secondly, the existence of data dependencies has a negative impact on the performance of the GPU-based algorithms. For this reason, these are often broken to enable higher performance rates, which also has an effect on the coding efficiency of the encoder. Nevertheless, although not optimal, the information calculated by the GPU is usually a good approximation.

In conclusion, HEVC represents a major step forward in the field of multimedia, but it requires the development of fast encoding algorithms to ensure its applicability in real-time scenarios. CTU/CU splitting and mode decision algorithms help reduce the total encoding time to a great extent, but their effectiveness depends on the accuracy of the information used to take the optimal decisions. The use of heterogeneous architectures has proven to be a good way to reduce the computational load of some of the encoding operations, but it suffers from the shortcomings described above from the point of view of the general encoding process. In this way, the idea behind the scheme proposed in this Doctoral Thesis is to make use of heterogeneous architectures as a means, not as an end, towards the reduction of the encoding time. The information calculated on the GPU will be used not to accelerate a specific encoding module, but to provide general support to the encoding process.

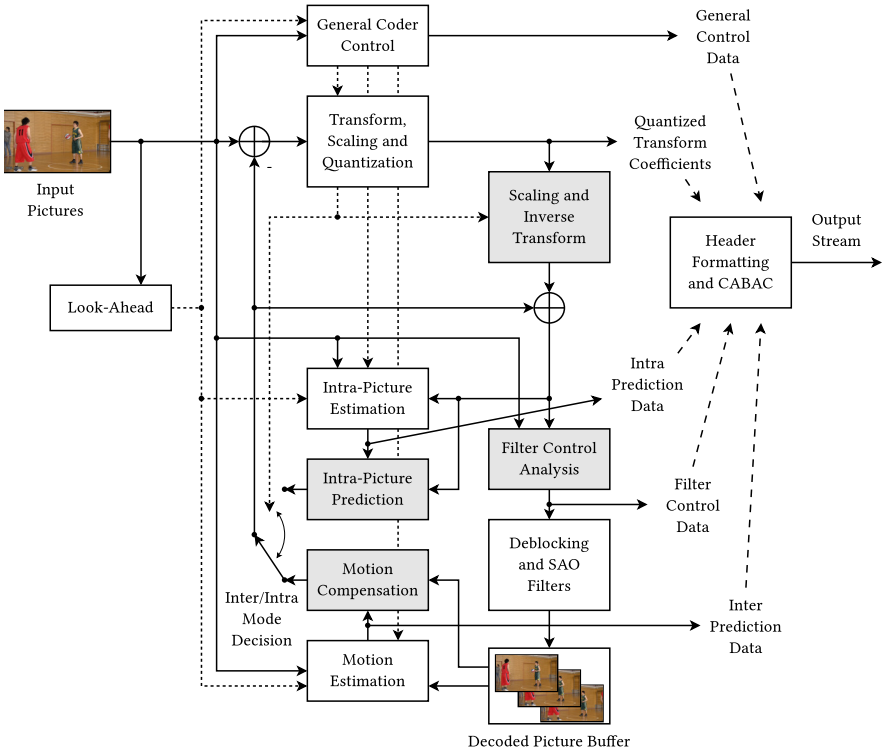


Figure 1.3: Typical HEVC encoder architecture with a look-ahead stage (blocks shaded in gray are shared with the decoder).

The GPU will thus execute what is known as a pre-analysis algorithm, also called a look-ahead algorithm. This technique makes it possible to obtain valuable information from the input sequence prior to the encoding, which can be used in the remaining encoding modules at a later stage, as shown in Fig. 1.3. Possible uses of this information include bit allocation, rate control, or scene change detection. For instance, the well-known open-source H.264/MPEG-4 AVC encoder called x264[1] makes use of a look-ahead stage for these purposes [42]. Our aim, however, is different. The information obtained in the look-ahead stage will be used to provide support to the inter prediction, and a splitting and mode decision algorithm, with the aim of targeting the most time-consuming parts of the encoding process. As a result, the encoding time will be significantly reduced while maintaining the coding efficiency of HEVC with the aim of addressing the most critical issue in the implementation of this standard in current scenarios, which is its computational complexity.

1.2 Objectives

In accordance with the motivation presented in the section above, the main objective of this Doctoral Thesis is to design a fast HEVC encoding algorithm that makes use of GPU-based heterogeneous architectures to reduce the total encoding time, while maintaining

[1] x264 Homepage. https://www.videolan.org/developers/x264.html, accessed on Jan. 28, 2018.

the coding efficiency of the standard. In order to address this main objective, the following goals are proposed as partial objectives:

- **Goal 1**: Review of the state of the art in the area of video coding to acquire an in-depth understanding of the HEVC standard and to perform a thorough study of the existing related works on fast encoding and heterogeneous platforms.
- **Goal 2**: Design, implementation and evaluation of a fast GPU-based ME algorithm for HEVC. At a first stage, this algorithm will be used to replace the integer-pel ME algorithm of the inter prediction module. Nevertheless, once the development of the look-ahead algorithm is finished, the information calculated on the device will be used as decision-making support for this stage.
- **Goal 3**: Design, implementation and evaluation of the look-ahead stage. This stage will calculate preliminary motion information from the input frames that will be used to accelerate the inter prediction in two ways. Firstly, to reduce the encoding time devoted to the ME, and secondly, to reduce the number of reference frames evaluated.
- **Goal 4**: Extension of the look-ahead stage to provide an estimate of the CTU/CU partitioning using the preliminary information mentioned above. Additionally, as part of this goal, it will be necessary to study different decision models that help determine whether to evaluate a CU or a prediction mode according to the estimation performed by the look-ahead stage.
- **Goal 5**: Integration of the GPU-based algorithm and the look-ahead stage in such a way that the motion information calculated by the latter is replaced by the information provided by the device. In this way, the look-ahead stage will have more information at its disposal, which will lead to a more accurate prediction and faster encoding.

1.3 Methodology and Work Plan

In order to achieve the main objective formulated in this Doctoral Thesis, it is necessary to define a methodology and a work plan. This plan establishes the set of tasks that need to be conducted for each goal to determine their completion. In this regard, this section describes all of the tasks grouped by goals:

- **Goal 1** — Review of the state of the art.

  Apart from the initial analysis of the HEVC standard, this particular goal cannot be divided into tasks, as it is an activity that develops throughout the course of the research. It includes the periodical review of the state of the art to search for related works on fast encoding and the use of heterogeneous platforms. In this way, it is possible to keep up to date with new advances in these two topics, which may help develop the proposed schemes.

- **Goal 2** — Implementation of a GPU-based ME algorithm.

  This goal corresponds to the first development stage of the proposed scheme. It focuses on the GPU side, developing an algorithm that is used to perform the ME, but taking into consideration that it will form the basis of the final fast encoding algorithm. The tasks included in this goal are:

  1. To analyze the HEVC test model (HM) reference software, which will be used as the baseline encoder for the proposed scheme [43].
  2. To perform a study of the existing proposals on GPU-based ME algorithms, placing special emphasis on the search patterns used and identifying the possible difficulties met in their development.
  3. To carry out the design of the algorithm, taking into consideration aspects such as synchronization points, data structures, search pattern, and the required kernels, among other aspects. It is also important to define the way in which the CPU and the GPU will work together. In this regard, it will be necessary to define whether to use a synchronous or an asynchronous model, and the granularity level of the algorithm (frame level, CTU level or PU level).
  4. To select the framework used for the development of the algorithm according to its capabilities and features.
  5. To implement the algorithm in the HM software, and to assess the validity of the results obtained.
  6. To optimize the algorithm in terms of device occupancy, communication overhead, memory hierarchy, and other factors.
  7. To perform an experimental evaluation of the algorithm on a wide range of test sequences and under common coding conditions, analyzing the results in terms of both time reduction and bit rate difference with respect to the baseline encoding.

- **Goal 3** — Implementation of a look-ahead stage.

  The first approach of this algorithm will be used to accelerate the inter prediction. In this regard, it will be designed in such a way that the entire calculation is performed on the CPU, but taking into consideration that this algorithm and the GPU-based algorithm from Goal 2 will be merged at a later stage. The tasks that are part of this goal are:

  1. To perform a study of the existing proposals on pre-analysis algorithms and fast encoding for HEVC.
  2. To design the look-ahead stage. In this regard, one of the aspects to take into account will be the type of information provided, such as distortion values, motion vectors (MVs), or the number of bits. Another aspect to determine is the level

at which this information is calculated, e.g. several frames ahead, one single frame ahead, or at CTU level. Lastly, it will be necessary to define the granularity level of the algorithm. The main two options in this regard are the use of square blocks or PU-sized blocks.

3. To implement the look-ahead stage in the HM software.

4. To integrate the information estimated by the look-ahead stage in the inter prediction module. This integration will be performed with two aims. Firstly, to reduce the search area, and secondly, to reduce the number of reference frames.

5. To parametrize the algorithm to determine the most suitable configuration for HEVC, such as the minimum and maximum block sizes.

6. To perform an experimental evaluation of the look-ahead stage.

- **Goal 4** — Extension of the look-ahead stage to implement a CTU/CU splitting and mode decision algorithm.

  In this case, this goal relies on Goal 3, and thus some of the tasks performed then can be used as a basis for this goal. Some of the new tasks are:

  1. To perform a study of the existing proposals on CTU/CU splitting and mode decision algorithms for HEVC, with special attention given to those which take the decisions on the basis of motion information.

  2. To adapt the look-ahead stage to estimate the partitioning of a CTU from the predicted motion information. In this regard, it will be necessary to define a rate-distortion (R-D) model to determine whether the decision of splitting a block is more suitable than not splitting it in terms of coding efficiency.

  3. To design a splitting and mode decision algorithm in the encoder on the basis of the partitioning estimated by the look-ahead stage. This algorithm will determine which CUs and PUs need to be evaluated to minimize the coding efficiency losses derived from prediction misses.

  4. To perform an experimental evaluation of the proposed CTU/CU splitting and mode decision algorithm based on the look-ahead stage, and its combination with our previous work on inter prediction.

- **Goal 5** — Integration of the GPU-based algorithm and the look-ahead stage.

  This corresponds to the final step towards the achievement of the main objective of this Doctoral Thesis. At this point, all the elements that make up the algorithm described in the motivation section will have already been developed, and thus it will only be necessary to merge them. Therefore, these are the tasks that are part of this goal:

1. To adapt the look-ahead stage to use the motion information calculated by the GPU. In this regard, it will be necessary to consider whether to adapt the R-D model accordingly.
2. To adapt the fast inter prediction techniques developed for the look-ahead stage in such a way that they use the motion information of the GPU instead.
3. To adapt the CTU/CU splitting and mode decision algorithm to make use of the motion information calculated by the GPU.
4. To perform an experimental evaluation of the proposed GPU-based fast encoding algorithm.

At the end of Goal 5, the main objective established for this Doctoral Thesis will have been achieved. The proposed scheme achieves a significant reduction in terms of encoding time at the expense of negligible coding efficiency losses. In spite of these results, the proposal is open to improvements and new scenarios. These are, however, beyond the scope of this dissertation, and will be summarized as future work in Section 8.

1.4 General Discussion and Description of the Proposal

As mentioned in the sections above, the main aim of this Doctoral Thesis is to reduce the total processing time of the HEVC encoder while maintaining its coding efficiency. We propose a GPU-based fast encoding algorithm as a solution, which is comprised of a splitting and mode decision algorithm, and fast inter prediction techniques. This section will provide an insight into this scheme, describing the design decisions that were taken in its implementation, and providing a better idea of its features and capabilities.

The first steps taken prior to any design or implementation stages were those that correspond to Goal 1, i.e. to analyze the state of the art. In addition to learning the details of the HEVC standard, this analysis helped discover the related works on the topic. From the study of these works, it was possible to see that many of them focused on either the CTU/CU splitting, or the inter or intra prediction depending on whether the scenario involved P and B slices or not, respectively. Some of these works were enumerated in Section 1.1. As a starting point, we decided to perform a time profile of the encoder to examine which are the encoding modules that take the longest, and thus are subject to optimization.

The results of the aforementioned profile are shown in Fig. 1.4. The evaluation was performed using the HM reference software following the coding conditions established by the JCT-VC [44]. The Full HD (1920×1080 pixel resolution) *Kimono* sequence was used for the test. The configuration used was RA, and the quantization parameter (QP) was set to 32. In order to understand the figure, it should be noted that the "Inter — Other" group involves the motion compensation operation, and the transform and quantization of the residual for inter prediction. It should also be taken into account that the "Other" group

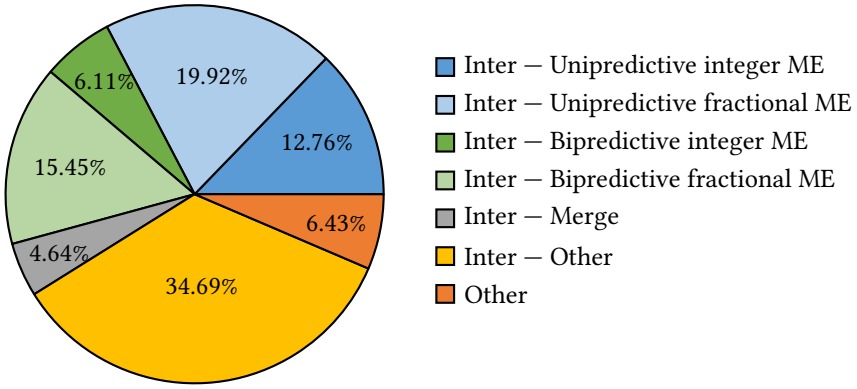


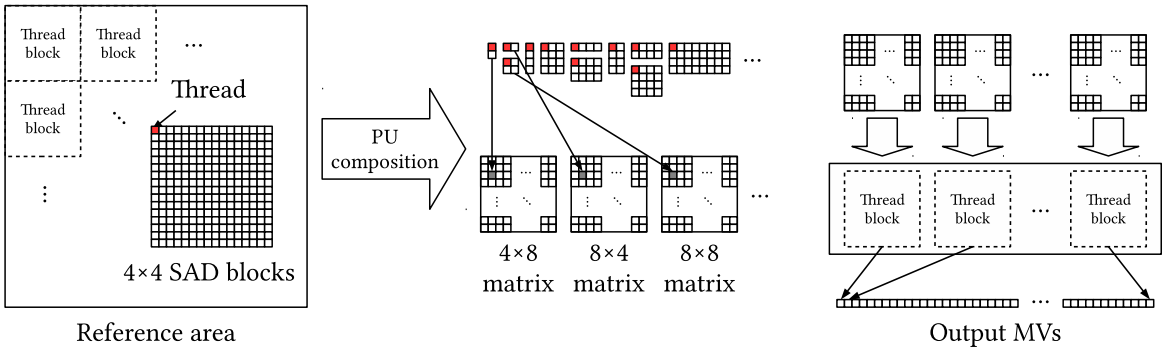
Figure 1.4: Time profile of the *Kimono* sequence using RA configuration and QP 32.

is composed of intra prediction (including the corresponding transform and quantization of the residual), in-loop filtering, and minor operations such as initializations and memory allocations. As can be seen, the vast majority of the encoding time is devoted to the inter prediction, and in particular to the ME, which accounts for more than half of the total processing time. In view of this, it seemed clear that this operation is a critical point in the encoding process, and thus it needs to be addressed in order to tackle the computational complexity of the encoder.

The ME operation basically consists in subtracting one PU from multiple positions in a set of reference frames in order to find the MV that provides the lowest R-D cost among all the positions tested. As can be seen, this operation fits into the massive data parallelism paradigm offered by the GPU. For this reason, we decided to make use of GPU-based heterogeneous platforms to offload the ME from the host. Nevertheless, the device implementation of the algorithm is not trivial, as it requires designing the algorithm for a different architecture, and dealing with the existing dependencies, which may have a negative impact on the performance if not carried out correctly.

To avoid delays caused by data transfers and kernel executions, the communication scheme between device and host is asynchronous. The original frames are copied to the GPU as soon as the corresponding group of pictures (GOP) starts being processed. The ME is performed, however, by referencing the reconstructed version of the pictures. For this reason, the original frames are replaced with their reconstructed versions once they are encoded. With regard to the assignation of workload, the GPU always executes the ME of one CTU ahead of the CPU in such a way that the motion information is always available at the host when needed. The only exception is the first CTU in a frame, for which the host might need to wait for the device to finish.

With regard to the implementation of the ME on the GPU, it calculates the MVs and their associated costs for all the possible PUs and reference frames. This is performed in two steps, and thus it is divided into two different GPU kernels. The first kernel executes the operations required to calculate the sum of absolute differences (SAD) across a search



(a) PU composition kernel. (b) Reduction kernel.

Figure 1.5: Scheme of the two kernels of which the GPU-based algorithm is composed.

area in the reference frame for each PU, while the second one determines the best MV in terms of minimum SAD.

The first kernel relies on the fact that every PU size established by the standard is divisible by four, and thus it is possible to calculate the residual information of a PU from the composition of its 4×4 SAD costs. Following this approach, the previously mentioned kernel distributes a GPU thread to each sample in the reference search area, which represents an MV. Once assigned, each thread is responsible for calculating all the 4×4 SAD blocks in a CTU, which it then puts together to obtain the cost of each possible PU. This is thus equivalent to performing a full search in the reference area. As a result, the device has at its disposal the prediction costs in terms of the SAD for each position in the search area, every PU, and every reference frame. Accordingly, the second kernel consists in a reduction algorithm that selects the best MV in terms of minimum cost for each PU and reference frame. The entire process is depicted in Fig. 1.5.

After executing the GPU-based ME, the resulting information is copied from the device to the host. This information is then used to omit the integer ME of the inter prediction in the encoder. Instead of performing the corresponding search, the CPU only needs to retrieve the MV that the GPU calculated asynchronously. As a result, not only the encoding time is reduced, but also the coding efficiency is improved in some cases because of the exhaustiveness of the search performed by the device. Moreover, one of the advantages of this method is that it can be used in combination with other schemes. In fact, we implemented it in a transcoding scenario, and also together with the parallelization techniques included in the standard, i.e. tiles and WPP. The results of some of these proposals will be summarized in Section 1.5. Nevertheless, it should be taken into account that, although this scheme achieves a reduction in the encoding time, this algorithm was designed to be used by the proposed look-ahead algorithm.

As mentioned in the motivation for this work, the aim of the look-ahead stage is to calculate preliminary information that is used later in the encoding for different purposes. In our case, the main use intended for this stage is to reduce the total encoding time, in line with the objective of this Doctoral Thesis. In this regard, two considerations have

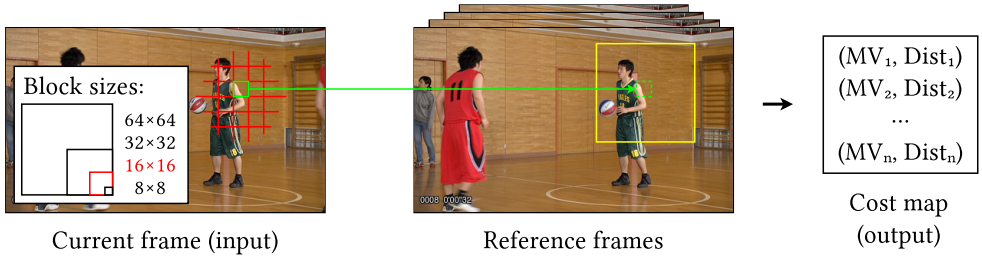


Figure 1.6: Diagram of the ME performed by the look-ahead stage.

been taken into account. Firstly, as seen above, inter prediction takes more than half of the encoding time. Secondly, the most frequent prediction mode used to encode P and B frames is inter. For these reasons, the main idea behind the proposed scheme is that the availability of motion information prior to the encoding can be helpful in the reduction of the total processing time. This reduction can be achieved in two ways:

- The use of estimated MVs can help omit part of the inter prediction in a similar way to the GPU-based algorithm proposed above. However, these MVs will still need to be refined in order to produce an accurate prediction.
- The estimated inter prediction costs can be used to determine the CTU/CU partitioning and the PU scheme of a frame.

The main role of the proposed look-ahead stage is thus to estimate the motion information that will be used for these two purposes. The GPU-based ME algorithm described above could be used as a basis for this algorithm. However, to ease the design of the proposed method, we decided to postpone its integration to a later stage of development. Therefore, in order to calculate the motion information, the look-ahead stage needs to implement an ME algorithm. Nevertheless, this algorithm needs to be simple enough to provide this estimation within a short time to avoid any latencies.

Evaluating all the PU sizes defined in HEVC would be excessively time-consuming. For this reason, our first approach was to use blocks of a specific size. We performed a study on the most suitable block size for the look-ahead stage ranging from 8×8 to 64×64 pixels, and the results showed that 16×16 provided more accurate results on average, while 8×8 also showed excellent accuracy for sequences of lower resolution [45]. Nevertheless, it was also possible to conclude that using only one block size, albeit fast, does not provide the required level of accuracy to maintain the coding efficiency of HEVC. As a consequence, we decided to use the following block sizes: 8×8, 16×16, 32×32, and 64×64. This provides high flexibility when estimating the motion information in a frame.

Similarly to the proposed GPU-based ME algorithm, the ME performed in the look-ahead stage is carried out for each block size and on every reference frame, as depicted in Fig. 1.6. As a result, each tuple is assigned a data structure that stores the MV and the corresponding distortion in terms of the SAD. This is the information used later by the encoder to aid the inter prediction, and the splitting and mode decisions.

Among the design considerations taken into account, one is aimed at the time constraints imposed on the look-ahead stage. In this regard, the motion search performed in this stage is carried out with integer-pel precision, which enables a faster prediction while maintaining the accuracy level required for the estimation. With respect to the coding efficiency, another design decision corresponds to the use of estimated predictors in a similar way to the advanced motion vector prediction (AMVP) feature defined in HEVC [46]. In this way, it is possible to avoid the drifting of MVs with respect to the neighbors, and to estimate the rate component by applying the following equation:

$$J\left(MV\right) = Dist\left(MV\right) + \lambda \cdot Rate\left(MV\right) \tag{1.1}$$

where J represents the cost function for a given MV, $Dist$ the distortion function in terms of the SAD, λ is the Lagrangian multiplier, and $Rate$ represents the function to calculate the bits required to code the MV. It should also be noted that the QP is present in the cost function via λ, so that high QP values lead to a higher weight for the rate component.

Once the look-ahead stage has estimated the motion information of a frame, the encoder can make use of it to accelerate the encoding process. As mentioned above, one of the targets of this time reduction is the inter prediction. In this regard, the aforementioned information can be used in the following two ways:

- To reduce the search area of the integer ME in the encoder, given that the look-ahead stage has already calculated an estimate of the MV, and thus it only needs a small refinement. Additionally, the search pattern used by the encoder can be replaced with a local search algorithm, such as a hexagon-based pattern search.

- To reduce the number of reference frames evaluated. This is achieved by evaluating all the reference frames for the 2N×2N PUs, while for the remaining sizes the following heuristic is used: in the case of unidirectional prediction, if the reference frame whose cost is the lowest according to the look-ahead stage for the current PU matches the reference frame chosen for the 2N×2N PU, then only this reference is used; in the case of bidirectional prediction, only the two frames used for the 2N×2N PU are evaluated if one of them matches the frame selected by the look-ahead stage.

While these techniques enable a reduction in the time devoted to the inter prediction, and thus in the total encoding time, there are still some other ways in which the motion information from the look-ahead stage can be used to reduce it even more. In particular, as mentioned earlier in this section, one of these ways is to estimate the CTU/CU splitting and the mode selection decision in P and B frames. It is assumed that, given that inter is the dominant mode in these two types of frames, the calculated inter prediction costs can help obtain an accurate estimate of both the partitioning and the mode. Taking into account that these decisions are made at a higher level, and that the execution of the remaining encoding operations depends on them, the speed-up that can be obtained with the omission of some of the possible CU and PU combinations is very high.

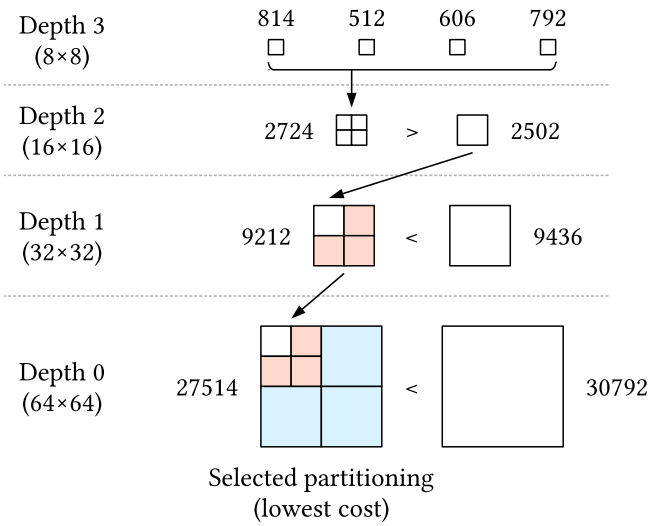
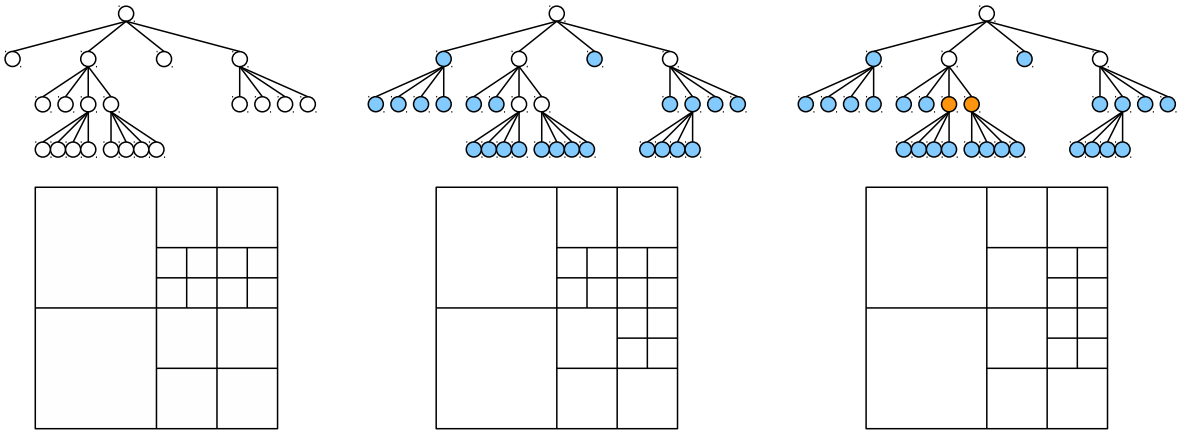


Figure 1.7: Estimation of the CTU/CU partitioning following a bottom-up approach.

The look-ahead stage uses the motion information calculated using Eq. 1.1 to predict the partitioning quadtree of a CTU. From the leaves of the tree to the root, these costs are recursively grouped into fours. If the sum of the costs estimated for each child CU is lower than the cost of a given CU, then it will be split, as this results in better coding efficiency according to the look-ahead algorithm. The CU will remain unsplit otherwise. This process is exemplified in Fig. 1.7, in which the colored blocks represent CUs whose partitioning has been determined by following this procedure, but these become irrelevant in understanding the figure. The numbers on the left represent compound costs, and the values at the top and on the right represent the look-ahead costs associated with the blocks of the corresponding depths.

On the basis of the estimated partitioning, it would be possible to evaluate only the CUs determined by the look-ahead stage, which would result in very large speed-ups. Nevertheless, this may also lead to coding inefficiencies because of prediction misses. Accordingly, a safer approach would be to also evaluate some of the modes in the parent and in the child CUs. In this regard, we carried out a thorough analysis of different static CU/PU models to determine the best scheme in terms of both time reduction and coding efficiency. For this purpose, we made use of statistical tests. These enabled the selection of the models that offered the optimal trade-off between the two parameters. However, the disadvantage of using static models is their limited adaptivity to the encoding process in comparison with dynamic models.

With the aim of improving the accuracy of the static models, we decided to evaluate other CUs and modes in an adaptive way. Apart from correcting possible prediction errors, this approach also enables the early termination of a block if it is detected that the optimal partitioning has already been found. In this way, the scheme followed to carry out this dynamic evaluation is to first adopt a top-down approach, starting from the CUs that the look-ahead stage decided not to split, and splitting them only under certain conditions. In



(a) Look-ahead estimation. (b) After top-down evaluation. (c) After bottom-up evaluation.

Figure 1.8: Example of the dynamic CTU/CU splitting evaluation on the basis of the partitioning estimated by the look-ahead stage.

other words, the encoder can ignore the decision of the look-ahead stage if this could result in greater coding efficiency. The splitting decision is taken on the basis of the coded block flag (CBF), which accounts for the significance of the entire TU, i.e. it signals whether a TU contains non-zero transform coefficients [47]. After the top-down exploration is finished, a bottom-up approach is followed for the parent CUs that were omitted in the first pass. In this case, the information concerning the corresponding four child CUs is available for the splitting decision. The evaluation of the current CU and its modes is thus taken on the basis of the complexity of the child CUs, i.e. whether they were split or not. To illustrate the entire process, Fig. 1.8 shows an example of the CTU evaluation following this approach, in which the nodes colored in blue and orange represent CUs evaluated in the top-down or in the bottom-up passes, respectively. It should be noted, however, that the set of prediction modes evaluated in each CU may differ.

The definition of this model completes the acceleration framework proposed at the beginning of this section. The motion information calculated in the look-ahead stage is used to accelerate both the inter prediction, and the splitting and mode decision algorithms. As a last step, it remains to consider the integration of the proposed GPU-based algorithm and the look-ahead stage in such a way that the latter makes use of the motion information calculated by the former. There are, however, some considerations that should be taken into account prior to the integration:

- The look-ahead stage calculates the motion information on a square-block basis. Nevertheless, the GPU-based algorithm calculates this information for all the possible PU sizes. Therefore, the techniques proposed above need to be adapted to make full use of the new data.

- The look-ahead stage defines an R-D function to estimate the MV of each block. This, however, would involve a great number of dependencies in the GPU-based algorithm,

which would in turn lead to a notable reduction in its performance. For this reason, only the distortion is taken into account in the ME carried out by the GPU. It is thus necessary to examine whether this design decision has a negative impact on the coding efficiency, and to what extent.

After a study of these aspects, we integrated both algorithms into a single GPU-based fast encoding algorithm. With regard to the fast methods defined for the inter prediction, we opted for a compromise solution between the two approaches. In a similar way to when the MVs of the look-ahead blocks were used to reduce the number of reference frames and the size of the search area, the MVs provided by the GPU for the corresponding PUs are now used instead for that purpose. The local refinement step performed in the integer ME is maintained, as well as the R-D model used to calculate the partitioning costs. Regarding the splitting and mode decision algorithm, we explored two different options. On the one hand, performing the estimation using the cost of the 2N×2N blocks calculated on the GPU in a similar way to the look-ahead algorithm, and on the other hand, taking into consideration all the estimated PU costs. While the former provides similar results to the standalone look-ahead stage, the latter is able to limit the number of combinations evaluated to a larger extent, and thus to achieve greater speed-ups.

In conclusion, the proposed GPU-based fast encoding algorithm satisfies the main objective of this Doctoral Thesis. It enables the reduction of the total encoding time by using the motion information calculated by the GPU in a so-called look-ahead stage. Furthermore, as will be shown in the next section, the coding efficiency of the standard is also maintained. It thus becomes an excellent solution for overcoming the complexity of HEVC. The details of each of the proposed methods will be discussed in Chapters 2 to 7.

1.5 Results

This section aims at describing the results obtained by the proposed scheme. These include the numeric results in terms of time savings and coding efficiency, and the research items published in the literature.

The tasks associated with Goal 1 do not apply, as they correspond to the review of the state of the art. After completion of the first goal, we carried out the work required to achieve Goal 2, which corresponds to the design and implementation of the GPU-based ME algorithm described in the section above. An experimental evaluation of the algorithm using the RA configuration showed that it is able to achieve 10.46% time savings on average at the expense of only a 0.2% Bjøntegaard delta (BD) rate. The BD-rate is a measure of coding efficiency that represents the percentage of bit rate variation between two sequences with the same objective quality [48]. While this increase in BD-rate proves negligible in practical scenarios, it is worth mentioning that in some cases the algorithm achieved a negative BD-rate, and thus an improvement in coding efficiency. This is due to the exhaustiveness of the

full search performed on the GPU. With regard to the time savings achieved, this percentage corresponds to the portion of time devoted to the unipredictive integer ME in the encoder, which is offloaded to the GPU when using this scheme.

As mentioned in Section 1.4, one of the advantages of the proposed GPU-based ME algorithm is that it can be used in combination with many other approaches. In particular, it is especially suitable for parallel platforms. Several processing units can use parallel encoding techniques such as tiles or WPP to carry out the encoding concurrently, offloading the integer ME to a single GPU. In this regard, we combined the proposed algorithm with the WPP technique. An evaluation of this scheme on a quad-core platform showed that it could successfully scale without the GPU becoming a bottleneck. Consequently, the device could achieve approximately the same time savings as before in addition to the time reduction already provided by WPP. These results were first included in the paper *Accelerating HEVC Using GPU-based Heterogeneous Platforms*, published in the *HPC Symposium* of the *Proceedings of the 14th International Conference on Computational and Mathematical Methods in Science and Engineering* [49]. Later, this work was extended to include an analysis of the combination of the proposed method with a GOP-based parallel approach. These new results were included in the paper *Accelerating HEVC Using Heterogeneous Platforms*, published in the *Journal of Supercomputing* [50].

The results obtained by the proposed algorithm led to a series of collaborations with other universities and research groups. In this regard, we continued exploring other parallel techniques to be used in combination with the GPU-based ME algorithm, such as slicing and tiles. In the former case, a combination with a slice-based parallel algorithm was described in *GPU-Based Heterogeneous Coding Architecture for HEVC*, published in *Algorithms and Architectures for Parallel Processing* as part of the *Lecture Notes in Computer Science* book series [51]. In the latter case, the use of tiles was considered in the paper called *Heterogeneous CPU Plus GPU Tile-Based Approach for HEVC*, published in the *HPC Symposium* of the *Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering* [52]. The two approaches were analyzed and compared in further detail in a paper extension called *Heterogeneous CPU Plus GPU Approaches for HEVC*, which is currently under review in the *Journal of Supercomputing* [53]. Unlike the previous works, the hardware platform used in these ones enabled the evaluation of the scalability of the algorithm using a higher number of processing units. The results showed that the GPU was able to scale successfully, with minimum waits derived from synchronization points.

Finally, we applied the GPU-based ME algorithm to scenarios other than fast encoding. In particular, we integrated this algorithm into a parallel H.264-to-HEVC transcoding framework, showing that it could help accelerate the ME performed by HEVC. As a result, the total transcoding time was reduced. This framework was proposed in *Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding*, published in the *Journal of Real-Time Image Processing* [54].

With regard to Goal 3, which corresponds to the development of a look-ahead stage, we first implemented a two-stage ME algorithm. In the first step, the look-ahead stage estimates the motion information on a fixed-size block basis, while in the second, which corresponds to the inter prediction, the estimated motion information is used to accelerate the ME of all the PUs. This scheme enabled a reduction of the search area, and consequently of the encoding time by 14.63% on average at the expense of a 0.9% BD-rate for the RA configuration. These results, however, are not comparable with the ones of the GPU-based ME algorithm because the base encoder used in the experiments was different, and thus the time distribution. Nevertheless, the main aim of this first approach was to analyze the different parameters that define the look-ahead stage, such as the block size, and the size of the search area. The main conclusion from this analysis was that the optimal configuration involved a block size of 16×16 pixels, and a search area of 8×8 pixels for the integer ME. These conclusions were collected in *Reducing HEVC Encoding Complexity Using Two-Stage Motion Estimation*, published in *2015 Visual Communications and Image Processing (VCIP)* [45].

In view of the results of this first approach, we decided to evaluate the use of multiple block sizes in the look-ahead stage with the aim of improving the coding efficiency of the proposal. In this way, as mentioned in Section 1.4, the set of block sizes was extended to include those from 8×8 to 64×64 pixels. Additionally, the reference frame selection algorithm was included in the inter prediction module. The results of this and the above approach were tested separately, showing that the search area reduction led to 5.73% time savings at the cost of a 0.1% BD-rate on average, and the reference frame selection resulted in a 7.80% time reduction for a 0.2% BD-rate increase on average. When both approaches were combined, the time reduction rose to 16.10% at the expense of only a 0.3% BD-rate on average. These results are part of the paper *A Pre-Analysis Algorithm for Fast Motion Estimation in HEVC*, published in *2016 IEEE International Conference on Image Processing (ICIP)* [55]. In short, the modifications performed on the look-ahead stage proved to be beneficial, at the expense of a small processing overhead.

One of the shortcomings of the proposed look-ahead stage is that it becomes ineffective in scenarios in which only I frames are used. For this reason, we provided it with some intra prediction functionality. A rough decision mode was implemented in the look-ahead stage, which consisted in testing the 35 intra modes using the SAD as a distortion metric. There were, however, some limitations, such as the unavailability of the reconstructed samples. As a result, it was possible to reduce the encoding time by 7.07% on average, at the expense of a 0.5% BD-rate in all-intra (AI) scenarios, as shown in *Two-Stage Intra Prediction Algorithm for HEVC*, published in the *HPC Symposium* of the *Proceedings of the 16th International Conference on Computational and Mathematical Methods in Science and Engineering* [56].

In order to provide an integral fast encoding solution, both the inter and intra prediction approaches were combined in such a way that the proposed scheme could work in any scenario. However, given that intra prediction is rarely used in P and B frames, the proposed approach was applied only to I frames to avoid unnecessary processing in the look-ahead stage. In this way, the results obtained in the AI were maintained, while the ones of the

RA configuration varied slightly, displaying a time reduction of 16.19% at the expense of a 0.4% BD-rate. As can be seen, these values are very similar to those obtained using only the inter prediction approach, but with the added benefit of being able to process I frames. The full study was included in *Inter and Intra Pre-Analysis Algorithm for HEVC*, published in the *Journal of Supercomputing* [57]. With this, the tasks of Goal 3 were considered completed.

The tasks of Goal 4 were aimed at extending the look-ahead stage to support the splitting and mode decisions. This support was achieved by using the distortion costs to estimate the partitioning of the CTUs. As a first step, we studied the results obtained by a set of static models generated from the partitioning estimated by the look-ahead stage. A statistical analysis of these results showed that it was not possible to find a model that offered an optimal solution from the point of view of both time savings and coding efficiency. However, it was possible to find two different models offering the best trade-off from the point of view of each parameter separately. In this regard, a speed-up-oriented configuration displayed a time reduction of 61.41% with a 3.1% BD-rate, while a BD-rate-oriented configuration obtained 56.80% time savings at the expense of a 2.4% BD-rate for RA configurations. While the coding efficiency was slightly affected, the achieved time savings were significantly higher in comparison. It should be noted that the inter prediction approaches proposed in prior works were not taken into consideration in these results. This scheme was submitted for publication to *IEEE Transactions on Circuits and Systems for Video Technology* under the name *Look-Ahead Partitioning Based on Motion Information for HEVC* [58].

The proposed static models led to significant time savings, but the coding efficiency was also affected as a result. To improve the accuracy of these models, and thus the resulting coding efficiency, we proposed the use of adaptive models. In this way, the BD-rate obtained was reduced to 1.4%, while the time savings increased to 65.33% in RA configurations. When combined with the inter prediction approaches proposed in prior works, these adaptive models obtained a 70.55% time reduction at the expense of a 1.8% BD-rate. These results represent an excellent trade-off between the two variables, saving around two thirds of the encoding time at the cost of negligible losses in terms of coding efficiency. In fact, we performed an exhaustive comparison with a large number of related works, showing that the proposed scheme was able to outperform the vast majority of them. These results were submitted for publication to *IEEE Transactions on Multimedia* under the title *Fast Inter CU Partitioning Based on a Look-Ahead Stage for HEVC* [59].

Finally, with regard to the integration of the GPU-based algorithm and the look-ahead stage, the initial results were similar to those of the standalone look-ahead stage, which proved that the two approaches could be merged into a GPU-based fast encoding algorithm without having a negative effect on either the coding efficiency or the performance. After pertinent adaptations, the proposed method was able to achieve average time savings of 83.53% at the expense of an increase in the bit rate. When compared with some of the best-known HEVC encoders on the market, it could be seen that the proposed scheme outperformed them in terms of coding efficiency while achieving similar speed-ups. There are, however, no works published or submitted for consideration at the moment of writing.

The proposed integration will be submitted to a journal after taking into account the future work guidelines described in Chapter 8.

Summary of Results

To sum up, all the goals proposed in Section 1.2 have been completed by following the methodology described in Section 1.3, and the list below shows all the works that have been published, or that are currently under review:

- **Goal 2** – Implementation of a GPU-based ME algorithm.
  - *Accelerating HEVC Using GPU-based Heterogeneous Platforms*, published in the *HPC Symposium* of the *Proceedings of the 14th International Conference on Computational and Mathematical Methods in Science and Engineering* [49]. Conference paper. Description: design and evaluation of the GPU-based ME algorithm, and its combination with the WPP technique.
  - *Accelerating HEVC Using Heterogeneous Platforms*, published in the *Journal of Supercomputing* [50]. Journal paper, JCR2015 Q2, IF 1.088. Description: evaluation of the GPU-based ME algorithm, and its combination with WPP and a parallel GOP-based approach.
  - *GPU-Based Heterogeneous Coding Architecture for HEVC*, published in *Algorithms and Architectures for Parallel Processing* as part of the *Lecture Notes in Computer Science* book series [51]. Conference paper, CORE B. Description: evaluation of the GPU-based ME algorithm using slices as a parallelization technique.
  - *Heterogeneous CPU Plus GPU Tile-Based Approach for HEVC*, published in the *HPC Symposium* of the *Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering* [52]. Conference paper. Description: evaluation of the GPU-based ME algorithm using tiles.
  - *Heterogeneous CPU Plus GPU Approaches for HEVC*, submitted for publication to the *Journal of Supercomputing* [53]. Journal paper, JCR2016 Q2, IF 1.326. Description: evaluation of the combination of the GPU-based ME algorithm, and the slice-based and tile-based parallel algorithms.
  - *Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding*, published in the *Journal of Real-Time Image Processing* [54]. Journal paper, JCR2016 Q2, IF 2.010. Description: use of the GPU-based ME algorithm in a parallel transcoding scenario.

- **Goal 3** – Implementation of a look-ahead algorithm.
  - *Reducing HEVC Encoding Complexity Using Two-Stage Motion Estimation*, published in *2015 Visual Communications and Image Processing (VCIP)* [45]. Conference paper, CORE B. Description: primitive version of the look-ahead stage used to accelerate the integer ME. The conclusions of this work were used to determine the optimal parameters of the look-ahead stage.
  - *A Pre-Analysis Algorithm for Fast Motion Estimation in HEVC*, published in *2016 IEEE International Conference on Image Processing (ICIP)* [55]. Conference paper, CORE B. Description: the look-ahead stage was improved to evaluate multiple block sizes. Moreover, a reference frame selection algorithm was developed.
  - *Two-Stage Intra Prediction Algorithm for HEVC*, published in the *HPC Symposium* of the *Proceedings of the 16th International Conference on Computational and Mathematical Methods in Science and Engineering* [56]. Conference paper. Description: development of a rough intra prediction mode selection in the look-ahead stage to enable the processing of I frames.
  - *Inter and Intra Pre-Analysis Algorithm for HEVC*, published in the *Journal of Supercomputing* [57]. Journal paper, JCR2016 Q2, IF 1.326. Description: combination of the intra and inter prediction approaches mentioned above into a single fast encoding scheme for all the coding scenarios.
- **Goal 4** – Extension of the look-ahead stage to implement a CTU/CU splitting and mode decision algorithm.
  - *Look-Ahead Partitioning Based on Motion Information for HEVC*, submitted for publication to *IEEE Transactions on Circuits and Systems for Video Technology* [58]. Journal paper, JCR2016 Q1, IF 3.599. Description: added support for the CTU/CU splitting and mode decision algorithm on the basis of static models generated from the estimation performed by the look-ahead stage.
  - *Fast Inter CU Partitioning Based on a Look-Ahead Stage for HEVC*, submitted for publication to *IEEE Transactions on Multimedia* [59]. Journal paper, JCR2016 Q1, IF 3.509. Description: added support for the CTU/CU splitting and mode decision algorithm, but using adaptive models. Additionally, this scheme was combined with the inter prediction approaches formulated in prior works.

Other Results

To conclude, it is worth mentioning other proposals that have been published or are under review as a result of additional research, and do not fit in any of the above groups. Apart from 4 papers published in national conferences and workshops, other works are:

- **Parallel encoding techniques**.
  - *OpenMP HEVC Parallel Version based on a GOP Approach*, published in the *Proceedings of the Ninth International Conference on Engineering Computational Technology* [60]. Conference paper.
  - *Synchronous and Asynchronous HEVC Parallel Encoder Versions Based on a GOP Approach*, published in *Advances in Engineering Software* [61]. Journal paper, JCR2016 Q1, IF 3.000.
- **Fast inter prediction algorithms**.
  - *A Motion Vector Re-Use Algorithm for H.264/AVC and HEVC Simultaneous Video Encoding*, published in the *Proceedings of the 13th International Conference on Advances in Mobile Computing and Multimedia* [62]. Conference paper, CORE B.
- **Analysis of open-source encoders and performance evaluation**.
  - *On the Capabilities of the Open-Source HEVC Codecs*, published in the *HPC Symposium* of the *Proceedings of the 16th International Conference on Computational and Mathematical Methods in Science and Engineering* [63]. Conference paper.
  - *Parallelization and Performance Evaluation of Open-Source HEVC Codecs*, published in the *Journal of Supercomputing* [64]. Journal paper, JCR2016 Q2, IF 1.326.
- **Next generation video coding**.
  - *Rate-Distortion/Complexity Analysis of Video Compression with Capability beyond HEVC*, published in the *HPC Symposium* of the *Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering* [65]. Conference paper.
  - *Acceleration of the Integer Motion Estimation in JEM through Pre-Analysis Techniques*, submitted for publication to the *Journal of Supercomputing* [66]. Journal paper, JCR2016 Q2, IF 1.326.

CHAPTER 2

Accelerating HEVC Using Heterogeneous Platforms

- **Title**: Accelerating HEVC Using Heterogeneous Platforms.
- **Authors**: Gabriel Cebrián-Márquez, José Luis Hernández-Losada, José Luis Martínez, Pedro Cuenca, Minhao Tang, and Jiangtao Wen.
- **Type**: journal paper.
- **Journal**: Journal of Supercomputing.
- **Publisher**: Springer US.
- **ISSN**: 0920-8542.
- **Status**: published.
- **Publication date**: February 2015.
- **Volume**: 71.
- **Issue**: 2.
- **Pages**: 613–628.
- **DOI**: 10.1007/s11227-014-1313-8
- **JCR IF/ranking**: 1.088/Q2 (JCR2015).

J Supercomput (2015) 71:613–628
DOI 10.1007/s11227-014-1313-8

Accelerating HEVC using heterogeneous platforms

Gabriel Cebrián-Márquez · José Luis Hernández-Losada · José Luis Martínez · Pedro Cuenca · Minhao Tang · Jiangtao Wen

Published online: 18 October 2014

Abstract The high efficiency video coding (HEVC) standard achieves double compression efficiency compared with H.264/advanced video coding at the cost of huge computational complexity. Parallelizing HEVC encoding is an efficient way of fulfilling this computational requirement. The parallelization algorithms considered in HEVC, such as Tiles or wavefront parallel processing (WPP), rely on creating picture partitions that can be processed concurrently in a multi-core architecture. However, this paper focuses on the design of a heterogeneous parallel architecture composed of a graphic processing unit (GPU) plus a multi-core central processing unit (CPU) to take advantage of these techniques. Experimental results indicate that our approach outperforms WPP in terms of speed-up and reduces the delay introduced by alterna-

This work has been jointly supported by the Spanish Ministry of Economy and Competitiveness (MINECO) and the European Commission (FEDER funds) under the project TIN2012-38341-C04-04.

G. Cebrián-Márquez (✉) · J. L. Hernández-Losada · J. L. Martínez · P. Cuenca
Albacete Research Institute of Informatics (I3A),
University of Castilla-La Mancha, Albacete, Spain
e-mail: Gabriel.Cebrian@uclm.es

J. L. Hernández-Losada
e-mail: JoseLuis.Hernandez2@alu.uclm.es

J. L. Martínez
e-mail: JoseLuis.Martinez@uclm.es

P. Cuenca
e-mail: Pedro.Cuenca@uclm.es

M. Tang · J. Wen
Department of Electrical Engineering, Tsinghua University, Beijing, China
e-mail: tmh920811@163.com

J. Wen
e-mail: jtwen@tsinghua.edu.cn

tive techniques such as the group of pictures-based processing pattern. Moreover, the proposed algorithms obtain speed-up values of over 4× on an Intel quad-core CPU and an NVIDIA GPU with negligible quality losses.

Keywords HEVC · Parallelization · GPU · Multicore · Heterogeneous computing

1 Introduction

The new high efficiency video coding (HEVC) standard [3] has recently been established by the joint collaborative team on video coding (JCT-VC), an expert group proposed by the ISO/IEC moving expert group (MPEG) and the ITU-T video coding expert group (VCEG). HEVC was initially conceived with the purpose of achieving adequate efficiency and performance to deliver high-quality multimedia services over bandwidth-constrained networks, and also to support formats beyond high definition (HD) resolution, such as the new 4 and 8K formats. This standard is based on a well-known block-based hybrid video coding architecture, as was its predecessor, namely H.264/MPEG4 part 10—advanced video coding (AVC) [9], which it outperforms in terms of bitrate reduction at the same quality [12]. Among other features, HEVC incorporates multiple new coding tools, such as highly flexible quad-tree coding block partitioning, which includes new concepts such as coding unit (CU), prediction unit (PU) and transform unit (TU) [2,12].

All these improvements imply a considerable increase in the encoding time. Fortunately, this computational cost can be efficiently reduced by adapting the sequential algorithm to parallel architectures. Over the past few years the computing industry has tended towards including several processing units on a single shared chip. In fact, in terms of massive data computing, there are devices called graphic processing units (GPUs) which are normally used as co-processors to assist the central processing unit (CPU). CPUs and GPUs have different instruction set architectures, forming what it is known as a heterogeneous computing platform [7].

As an aid to this parallelism, HEVC places special emphasis on a hardware-friendly design and parallel-processing architectures. These parallelization approaches are *Tiles* [11] and wavefront parallel processing (WPP) [8], and these will be described in Sect. 2. Basically, these parallelization algorithms rely on creating picture partitions that break some dependencies for prediction, context-adaptive binary arithmetic coding (CABAC) modelling, and/or slice header overhead. As a result, coding losses may appear.

At this point, this paper proposes a GPU-based algorithm that makes use of this device in order to efficiently parallelize the motion estimation (ME) carried out in the HEVC inter-prediction algorithm. This algorithm is completely asynchronous from the CPU, in such a way that it can execute operations from other modules, resulting in lower execution times. Additionally, in terms of coding efficiency, this algorithm makes use of a full search pattern, providing even better Bjøntegaard Delta rate (BD-rate) results than the reference algorithm in some cases.

Furthermore, this algorithm can be combined with multiple coarse-grained parallelization techniques such as the aforementioned ones in a heterogeneous architec-

ture. In fact, this paper shows the results of combining this GPU-based proposal with two different parallelization techniques: WPP and a group of pictures (GOP) based coarse-grained algorithm used in previous standards. These algorithms allow a multicore system to process multiple coding tree units (CTUs) by splitting the frame in rows or the sequence in GOPs, respectively. In either case, the motion estimation of these regions is issued to the device, reducing the idle time of the whole architecture.

The proposed architecture is tested by comparing their results with the ones provided by the HEVC Test Model (HM) [10], outperforming them in terms of speed-up and coding efficiency. Results show speed-ups of up to 4.78× on a quad-core CPU using four threads plus simultaneous multithreading (SMT) with negligible rate-distortion (RD) penalties. Additionally, Radicke et al. [13] propose another GPU-based algorithm that makes use of a diamond search. Compared with this another proposal, our algorithm achieves better BD-rate values, ranging from 0.8 to 1.8 %.

The remainder of this paper is organized as follows: Sect. 2 includes the technical background of the new HEVC standard, while Sect. 3 identifies the related work which is being carried out on the topic. Section 4 introduces our proposed architecture. Experimental results are shown in Sect. 5. Section 6 concludes the paper and includes some possible lines of action for future work.

2 Technical background

As mentioned in the previous section, the main target of HEVC is to achieve lower bitrates for video streams while maintaining the same observable quality. In order to make this possible, HEVC introduces new coding tools with respect to its predecessor, H.264/AVC, and these enable a notable increase in coding efficiency. One of the most important changes affects picture partitioning, which is now performed following a quadtree structure. HEVC dispenses with the terms macro-block (MB) and block for the ME and the transform, respectively, and introduces three new concepts: CU, PU and TU. This structure leads to a flexible coding to suit the particularities of the frames. Each picture is partitioned into 64 × 64 pixel. pixel square regions called CTUs, which replace the MB structure of previous standards. Each CTU can be recursively divided into four smaller sub-areas called CUs, whose size ranges from 8 × 8 to 64 × 64 pixel. These regions may contain one or several PUs and TUs. This quadtree structure is shown in Fig. 1, in which each node represents one of the four CUs in which a CTU might be divided. For each level (current size of the CU), the split flag (SF) indicates whether it has been decided to split the corresponding CU or not.

For intra-picture prediction, a PU uses the same 2N × 2N size as that of the CU to which it belongs, allowing it to be split into quad N × N PUs only for CUs at the maximum depth level. Therefore, the PU size ranges from 64 × 64 to 4 × 4 pixels. For inter-picture prediction, several rectangular block shapes (square and non-square) are available, defining eight different PU sizes (2N × 2N, 2N × N, n × 2N, N × N, 2N × U, 2N × D, nL × 2N, nR × 2N). The prediction residual obtained in each PU is transformed using the residual quad tree (RQT) structure, which supports various TU sizes from 32 × 32 to 4 × 4. For the transform of intra 4× 4 PU residuals, an integer approximation of the discrete sine transform (DST) is used instead.

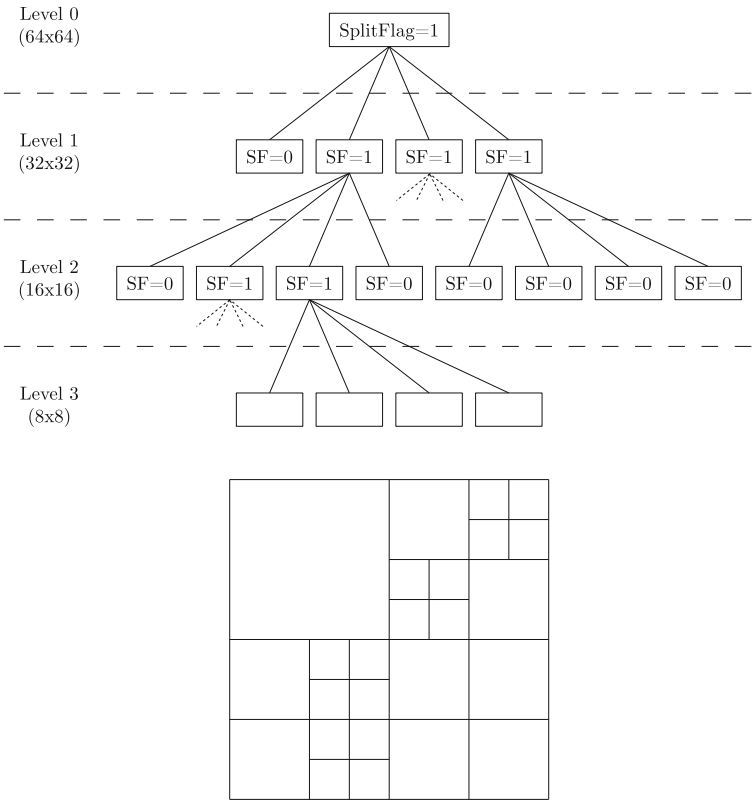
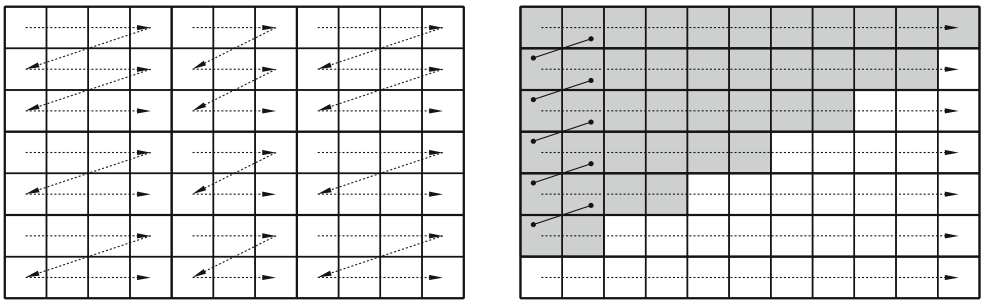


Fig. 1 CTU quadtree structure partitioning

High efficiency video coding (HEVC) checks most of the PUs (inter and intra modes) to decide whether it should split a CU or not, depending on the best RD case. Furthermore, in the case of inter prediction, for each of these PU partitions an ME algorithm is called. The wide range of possibilities makes this module much more computationally expensive than its predecessor, the one in the H.264/AVC standard. HEVC introduces changes in other modules too, such as intra prediction (where a total of 35 different coding modes can be selected), the new PU modes (asymmetric modes), additional image filters or new transform sizes, among others. As expected, the selection of the optimal partitioning for each CU/PU/TU is an intensive time-consuming process due to the huge number of combinations that have to be evaluated.

With the aim of reducing this huge complexity, the new HEVC codec also includes new parallelization techniques such as Tiles [11] and WPP [8]. On the one hand, Tiles are square or rectangular shape partitions where dependencies are broken across tile boundaries [11], making it possible to process them independently, taking into account that coding losses may appear. The in-loop filters (deblocking and sample adaptive offset, SAO), however, can still cross these boundaries. The number of tiles and their location can be defined for the entire sequence or changed from picture to picture. On the other hand, WPP allows the creation of picture partitions (rows) that can be processed in parallel. Unlike Tiles, entropy encoding and prediction are allowed to cross partitions in order to minimize coding losses. Nevertheless, coding dependencies



(a) Tiles partitioning **(b)** WPP partitioning

Fig. 2 Partitioning and processing order of Tiles (**a**) and WPP (**b**)

make it necessary to have a delay of at least two CUs between consecutive rows, in a similar way to segmentation in a computer architecture [5,8]. For this reason, not all the processes can start encoding these rows at the same time, which means low CPU utilization at the beginning and at the end of a frame, thus incurring the so-called "ramping inefficiencies". Both techniques are depicted in Fig. 2, in which an example of partitioning is shown for Tiles and WPP, respectively, as well as the corresponding processing order of the inner CUs.

Tiles and WPP have different merits and disadvantages. WPP is generally well suited for the parallelization of the encoder and the decoder due to its high number of picture partitions with low compression losses. However, the amount of parallelism with Tiles can be even higher, as the number of regions into which a frame is divided may vary. Additionally, WPP does not introduce artifacts at partition boundaries, as is the case for Tiles. In order to simplify the implementation, it is not possible to use Tiles and WPP simultaneously in the same compressed video sequence.

In either case, these approaches need parallel architectures to exploit their potential and, hence, to reduce the computational complexity of HEVC. In this respect, new architectures composed of multi-core CPUs and GPUs are being introduced in high-performance computing. A multi-core processor is composed of several processors sharing the same chip, while GPUs are composed of hundreds of similar simple processing cores, designed and organized with the goal of achieving high performance. These cores are grouped into stream processors that perform single instruction multiple data (SIMD) operations, which are suitable for arithmetic intensive applications. The main feature of these devices is a large number of processing elements integrated on a single chip at the expense of a significant reduction in cache memory.

3 Related work

As far as the related work in the literature is concerned, there have been many approaches focusing on accelerating different modules of the H.264/AVC encoding algorithm by means of parallel computing [4,14,16]. However, in the framework of HEVC, the first parallel approaches were focused on reducing the complexity of the decoding algorithm; in [5], the authors improve the WPP approach. The idea consists

of once there are no available rows in the current picture, the next one starts being processed. In this way, the ramping inefficiencies of WPP can be mitigated by overlapping the execution of consecutive pictures. This proposal was called overlapped wavefront (OWF). As a limitation, search areas need to be constrained to the region of the reference frame that has been already reconstructed.

In the context of this paper, which is more focused on the encoder side, there are not many approaches. OWF might work for the encoder, but no results were given in [5]. Yu et al. [18] proposed in a parallel candidate list to parallelize the motion vector prediction, but the proposal is not standard compliant. Later, in [17], the authors reduced the encoding time by up to 13 times using a 64-core architecture, which is far less accessible than the one used in this paper. Finally, Wang et al. proposed in [15] a scheme similar to the one in this paper based on a GPU plus multi-core CPU, but the major shortcoming of this paper lies in the fact that they did not use the HM reference software [10] and, thus, the RD results provided are worse due to the fact that not all coding tools were implemented [15].

In [13], the authors propose a GPU-based motion estimation algorithm that differs in some aspects from the one proposed in this paper. On the one hand, instead of performing a full search, the pattern used in this paper corresponds to a diamond search. This pattern may not obtain the best possible results, as not every position in the search area is checked. On the other hand, fractional motion estimation is also performed on the device, which leads to higher speed-up values. However, the absence of motion vector predictors (MVP) may have some negative effects on the coding efficiency of the algorithm.

4 Proposed algorithms

As seen before, parallelization is possible in both the encoder and the decoder using the algorithms defined in the standard, among others. Nonetheless, these are designed to be executed on a multi-core CPU, taking advantage of the capabilities that multiple threads may offer, but not taking into account other devices. Heterogeneous architectures such as the ones formed by the association of a multi-core CPU and a GPU are utilized in this paper, making use of the immeasurable power they can provide. The combination of a GPU-based motion estimation algorithm and two coarse-grained parallelization techniques is described in the following subsections.

4.1 GPU-based inter prediction algorithm

As motion estimation is the most resource intensive operation on the encoder side [12], this algorithm aims to reduce the time spent on the CPU by performing these searches on a GPU device. Nevertheless, taking into account that data transfers between host and GPU are highly time-consuming, these operations are performed asynchronously. In this way, time spent on the integer motion estimation (IME) is negligible compared with the default search algorithm.

As soon as a GOP starts being processed, it is possible to transfer the original frames that will be encoded to the device, making them available for subsequent uses. Later

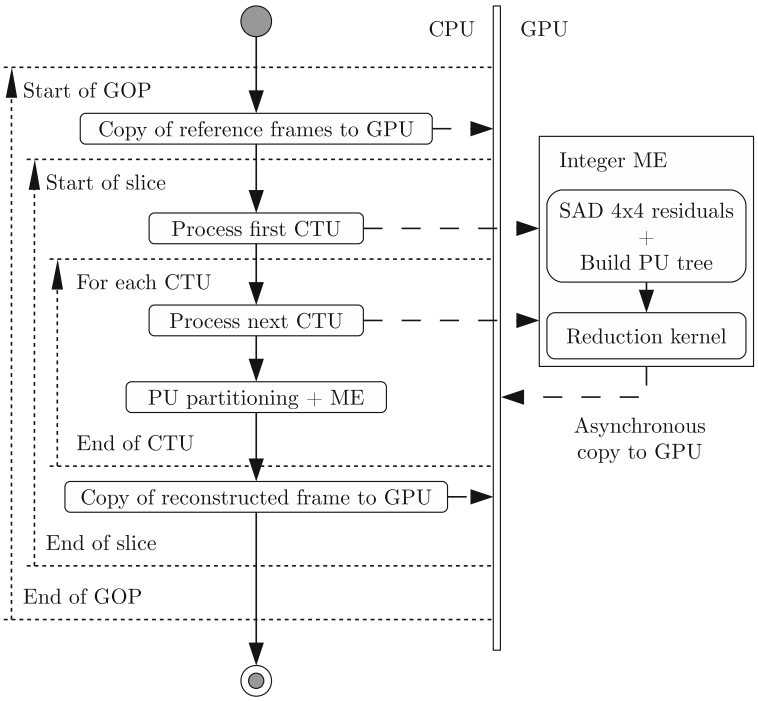


Fig. 3 GPU-based interprediction algorithm activity diagram

on, these frames are updated with their reconstructed version when they are encoded (and decoded in-loop) to correctly carry out motion estimation on the device. These operations are shown in Fig. 3.

When the encoder starts processing a slice, the host queues the execution of two consecutive kernels that perform the integer motion estimation of every PU partition in the first CTU. The first kernel executes the operations required to calculate the sum of absolute differences (SAD) residuals across a search area in the reference frame, while the second one determines which motion vector (MV) may offer the best possible result.

This algorithm relies on the fact that every PU size established by the standard is divisible by four, and taking into account the nature of the SAD operation, it is possible to calculate the residual information of a PU partition from the composition of its 4×4 SAD partitions.

Following this approach, the previously mentioned kernel distributes a device thread per sample in the reference search area. Every thread is responsible for calculating all the 4×4 SAD blocks in a CTU, taking as MV its position in the search area. Once these blocks are calculated, all the running threads put them together to obtain the PU partitions into which a CTU might be divided. From another point of view, the results of this step would be equivalent to a full-search algorithm performed for every PU partition.

At this point, the second kernel runs a reduction algorithm over the residual data obtained from the first one, so that the result of the GPU algorithm is a single table containing the best MV for every PU partition, which is copied asynchronously to the

host. After the transfer is finished, motion search operations related to the next CTU are then issued to the device.

By the time the host needs to perform the motion estimation of the CTU, integer MVs should be ready to be queried, it only being necessary to perform fractional motion estimation (FME) of the PU partitions which have not been skipped by the encoder.

4.2 Joint algorithms

As a consequence of the computational limit of a single processor, the idea of having multiple cores on the same chip was successfully introduced some time ago. One of its most relevant benefits is that a parallel application can achieve speed-up values in direct proportion to the number of cores. This, along with the ever-expanding existence of this kind of architectures, motivated the JCT-VC to include parallelism in HEVC, which was implemented by breaking some dependencies while trying to provide as much coding efficiency as possible.

These new techniques, however, have some issues that make them inadequate in some particular situations. Nevertheless, many algorithms used in previous standards are still applicable to HEVC. In this way, this subsection will present two different joint algorithms, one based on WPP and another one on a traditional GOP-based division pattern. This will show that our GPU-based proposal can be combined with many coarse-grained parallelization techniques.

4.2.1 WPP + GPU-based inter prediction

Unlike Tiles, WPP is able to keep some dependencies across partition boundaries, thus obtaining very good coding efficiency results. Even though this might have an effect on the maximum speed-up that it can achieve [5], WPP has been the algorithm chosen as the basis for the first of our joint proposals, aiming to achieve a trade-off between both parameters.

In our heterogeneous architecture, both the multi-core CPU and the GPU algorithms are independent. While WPP performs a coarse-grained parallelization of the whole encoding process, our GPU-based algorithm is able to carry out the IME operation. This independence makes it possible to combine these algorithms in a single proposal, thus obtaining higher speed-up values at the expense of a negligible increment in coding efficiency losses.

As depicted in Fig. 4, WPP allows multiple threads to process several CTUs concurrently (shaded regions). A single GPU device can carry out the integer motion estimation of these CTUs, queueing several kernels into the GPU. In particular, all these CTUs are located in the current frame and share the same reference list, i.e. the search areas can be found in the same frame, as shown in the figure. As a result of processing multiple CTUs, the device is fully utilized, lowering idle times. In addition, the GPU can process different kernels independently of the CPU in such a way that the host can continue processing other modules concurrently.

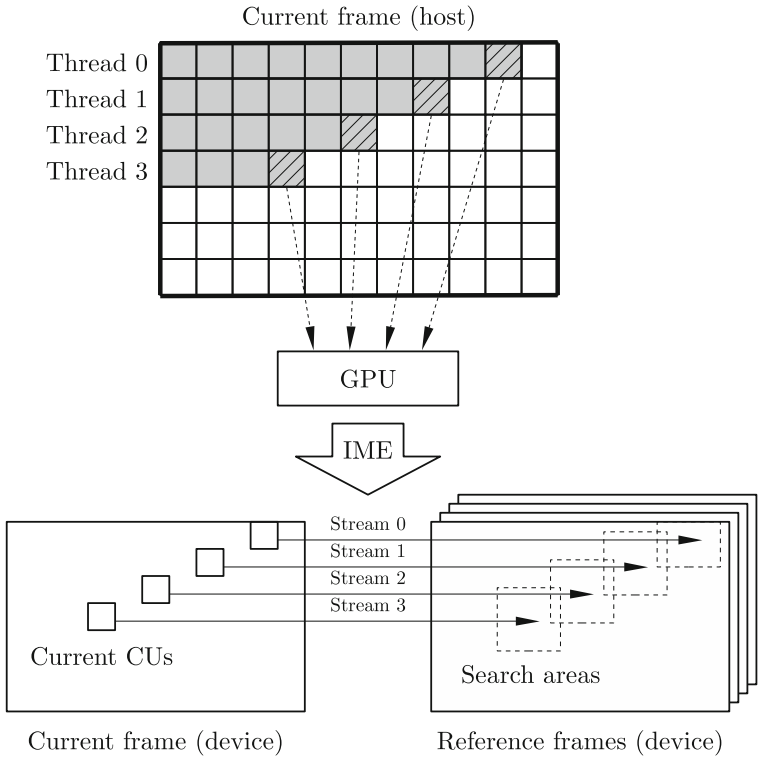


Fig. 4 Combination of WPP and the GPU-based proposal (4 threads)

4.2.2 GOP-based pattern + GPU-based inter prediction

Just as in previous video coding standards, HEVC offers an independent GOP-based encoding pattern into which a sequence can be divided. In this way, this second joint algorithm consists in assigning each GOP to a different thread, taking into account that no dependencies have to be broken between them. Due to this independence, the RD results obtained by this algorithm are very similar to the ones from the sequential algorithm. Moreover, this architecture is scalable to any number of cores.

The proposed architecture has a module splitter and joiner to allocate tasks to available cores and receive their results. Moreover, a dynamic module has been included to set up the execution schedule of the threads, taking into account that GOPs can have different delays. This is mainly because (1) the encoding algorithm can be carried out without following a sequential order and (2) different threads may have different time requirements because of variable residual data and, thus, time spent to encode it may diverge for each one.

Figure 5 shows how the GOPs into which a sequence is divided are issued to every thread. In this way, each thread processes a different frame and, hence, the motion estimation is performed on different CTUs. Therefore, the corresponding search areas are not located in the same frame. However, in a similar way to the previous joint algorithm, a single GPU device carries out the integer motion estimation of multiple threads, filling up its occupancy and taking full advantage of the available resources.

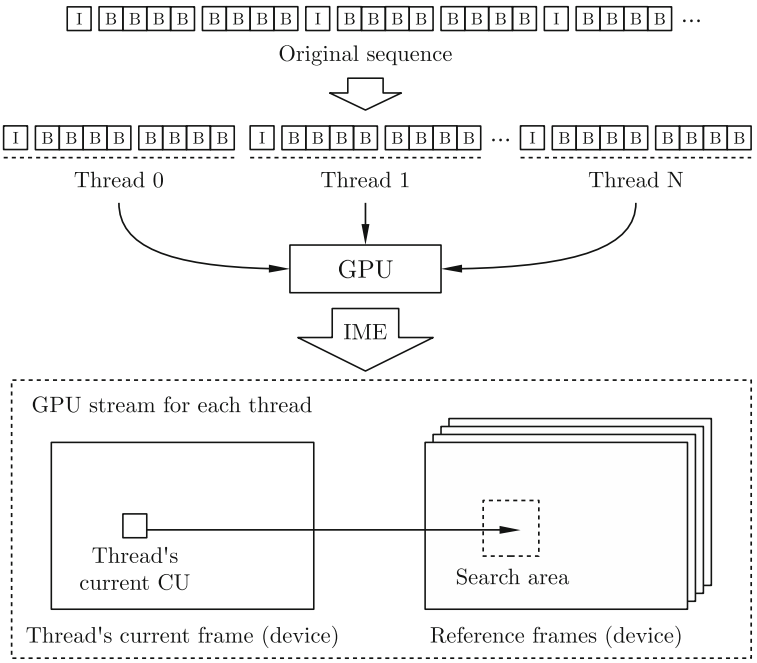


Fig. 5 Combination of the GOP-based algorithm and the GPU-based proposal

Once again, these operations are performed asynchronously, so that the CPU can keep carrying out other operations of the encoding process.

5 Performance evaluation

In order to ensure a common framework, the JCT-VC group defined a document [1] in which some test conditions are set out to homogenize comparisons between experiments. Therefore, this performance evaluation has been carried out in accordance with these guidelines, including the defined sequences and configuration parameters. This document also defines the Bjøntegaard Delta metric (BD-rate) used to measure the coding efficiency obtained by a proposal.

Random access (RA) with a color sub-sampling of 4:2:0 and 8-bit encoding was the configuration chosen to carry out the evaluation, as it is the most widely used configuration in real scenarios, but any other alternative might also work with our proposed algorithm. The selected quantization parameter (QP) values were {22, 27, 32, 37}, but only their average results will be presented. No other changes were made to the default parameters.

In this scenario, the aforementioned document [1] defines the set of video sequences to be encoded, which are grouped into classes according to their resolution, with the exception of Class F. This parameter will be relevant when presenting the results achieved by WPP.

- Class A (2,560 × 1,600): *Traffic*, *PeopleOnStreet*.
- Class B (1,920×1,080): *Kimono*, *ParkScene*, *Cactus*, *BasketballDrive*, *BQTerrace*.

- Class C (832×480): *BasketballDrill*, *BQMall*, *PartyScene*, *RaceHorses*.
- Class D (416×240): *BasketballPass*, *BQSquare*, *BlowingBubbles*, *RaceHorses*.
- Class F: *BasketballDrillText* (832 × 480), *ChinaSpeed* (1,024 × 768), *SlideEditing* (1,280×720), *SlideShow* (1,280×720).

The sequential algorithm of HM version 10.0 [10] has been used as the reference algorithm to calculate the corresponding speed-up and coding efficiency values. The proposed GPU-based algorithm has been tested separately to calculate its influence on the overall processing time. Later, the results achieved by the combination of this algorithm along with WPP and the GOP-based algorithm are also presented, aiming to show that the GPU-based proposal can be combined with several types of coarse-grained algorithms.

All the measurements have been performed on a quad-core Intel Core i7-2600 CPU running at 3.40 GHz and an NVIDIA GTX 560 Ti GPU running 384 CUDA cores at the frequency of 1.6 GHz. Consequently, tests have been carried out with 2 and 4 threads, as well as 4 plus SMT, enabling the processor to execute eight threads.

To start with, Table 1 shows the results of the proposed GPU-based algorithm. As can be seen, performing the IME operation on the GPU involves accelerating the encoding process by 1.12× while incurring very low coding efficiency losses (due to MV prediction), or even improving it in some cases. This is because the proposed algorithm performs a more exhaustive search. It is necessary to emphasize that these results are the theoretical limit of the integer ME, as the GPU has already calculated every MV when the host needs to perform this operation. In other words, the IME is performed in virtually perfect time.

On the other hand, Table 2 shows a comparison between the results provided by the combination of WPP and the GPU-based algorithm, and the ones provided by WPP itself. As can be seen, the proposal can reach speed-up values close to the ones from a parallel efficient algorithm (i.e. threads are almost fully utilized), providing that the frame size is large enough to exploit the available parallelism (see Class A). Accordingly, WPP obtains lower speed-up results with lower resolutions (see class D).

These results also show that combining both WPP and the GPU-based algorithm surpasses the results of WPP in terms of speed-up, reaching values of up to 4.33× on average (for Class A) compared with 3.92×, respectively. This behaviour is also present in other classes besides Class A, although the achieved speed-up values may be lower due to the aforementioned limitations of WPP. With regard to coding efficiency, this time reduction has a negligible impact of 1.3 % average BD-rate.

Table 3, in turn, shows the same comparison on the basis of the GOP-based parallelization technique. Unlike WPP, it can be seen that this other algorithm does not depend on the dimensions of the sequence, i.e. every class is able to achieve the same amount of parallelism. Moreover, as there are no dependencies between GOPs, and hence between threads, this algorithm is almost perfectly scalable.

Once again, it is shown that the combination of a coarse-grained algorithm, such as the GOP-based pattern, and the GPU-based motion estimation module outperforms the sole utilization of the former. In this way, this joint version can reach speed-ups of 4.46× on average (for Class A), compared with 4.13×, respectively. Moreover, it

Table 1 Speed-up and coding efficiency results of the GPU-based proposal

Class	Video sequence	Speed-up	BD-rate (%)
A	Traffic	1.07	0.2
	PeopleOnStreet	1.17	−0.5
B	Kimono	1.14	0.5
	ParkScene	1.07	0.0
	Cactus	1.11	0.2
	BasketballDrive	1.19	3.5
	BQTerrace	1.08	−1.0
C	BasketballDrill	1.14	−0.5
	BQMall	1.11	0.8
	PartyScene	1.09	−0.3
	RaceHorses	1.20	0.0
D	BasketballPass	1.15	0.0
	BQSquare	1.06	−0.1
	BlowingBubbles	1.07	−0.4
	RaceHorses	1.16	−0.5
F	BasketballDrillText	1.13	−0.7
	ChinaSpeed	1.15	−3.1
	SlideEditing	1.05	3.0
	SlideShow	1.08	4.3
	Class A	1.12	−0.2
	Class B	1.12	0.7
	Class C	1.14	0.0
	Class D	1.11	−0.2
	Class F	1.10	0.9
	Total average	1.12	0.2

achieves speed-up values of up to 4.78×. The average BD-rate of this joint algorithm stands at 0.6 %.

As can be seen, the results of both joint algorithms can be connected with the ones from Table 1, as the difference in speed-up and BD-rate compared with the GPU-based algorithm by itself stands at around 1.10× and 0.2 %, respectively. This means that the device is almost fully utilized, taking advantage of its resources and potential.

Choosing one or another coarse-grained parallelization technique will depend on the scenario in which the HEVC encoder is involved, e.g. WPP is adequate for large resolutions, while the GOP-based algorithm is ideal for transcoding operations. In either case, our GPU-based proposal is able to improve upon the achieved speed-up results with negligible coding efficiency penalties.

5.1 Comparison with related works

As stated in Sect. 3, Radicke et al. [13] propose a GPU-based motion estimation algorithm that is substantially different from the one presented in this paper. This subsection aims to evaluate these differences in terms of speed-up and coding efficiency.

Table 2 Speed-up and BD-rate results comparison between WPP and its corresponding joint algorithm

Class	Video sequence	Speed-up						BD-rate (%)	
		2 threads		4 threads		4 th. + SMT			
		WPP	w/GPU	WPP	w/GPU	WPP	w/GPU	WPP	w/GPU
A	Traffic	1.88	1.99	3.37	3.56	3.90	4.12	0.7	0.9
	PeopleOnStreet	1.89	2.17	3.35	3.85	3.95	4.53	0.7	0.1
B	Kimono	1.89	2.13	3.36	3.80	3.80	4.31	1.2	1.7
	ParkScene	1.88	2.01	3.33	3.56	3.70	3.97	0.7	0.7
	Cactus	1.89	2.07	3.31	3.65	3.72	4.13	1.1	1.4
	BasketballDrive	1.90	2.22	3.40	3.98	3.76	4.43	1.5	5.0
	BQTerrace	1.88	2.00	3.31	3.53	3.79	4.06	1.2	0.1
C	BasketballDrill	1.80	2.00	2.73	3.09	2.72	3.09	1.4	1.1
	BQMall	1.81	1.96	2.83	3.11	2.83	3.09	1.5	2.3
	PartyScene	1.78	1.91	2.70	2.94	2.71	2.96	0.6	0.2
	RaceHorses	1.78	2.09	2.79	3.30	2.84	3.35	0.8	0.8
D	BasketballPass	1.67	1.88	1.75	2.01	1.75	2.01	0.9	0.9
	BQSquare	1.60	1.66	1.84	1.92	1.84	1.92	1.3	1.3
	BlowingBubbles	1.59	1.66	1.80	1.91	1.80	1.90	0.9	0.6
	RaceHorses	1.63	1.85	1.81	2.10	1.81	2.10	0.9	0.4
F	BasketballDrillText	1.79	1.97	2.71	3.05	2.70	3.05	1.4	0.7
	ChinaSpeed	1.86	2.12	3.15	3.54	3.28	3.74	0.8	-2.3
	SlideEditing	1.80	1.86	3.13	3.24	3.33	3.46	1.0	3.9
	SlideShow	1.80	1.92	3.00	3.21	3.24	3.45	2.2	6.7
	Class A	1.88	2.08	3.36	3.71	3.92	4.33	0.7	0.5
	Class B	1.89	2.08	3.34	3.70	3.75	4.18	1.1	1.8
	Class C	1.79	1.99	2.77	3.11	2.78	3.13	1.1	1.1
	Class D	1.62	1.76	1.80	1.99	1.80	1.98	1.0	0.8
	Class F	1.81	1.97	3.00	3.26	3.14	3.43	1.3	2.3
	Total average	1.80	1.98	2.85	3.15	3.08	3.41	1.0	1.3

In order to achieve comparable results, the same test conditions as in [13] have been used. In this way, the same four sequences have been encoded (*Traffic* and *PeopleOnStreet* from Class A, and *Kimono* and *ParkScene* from Class B) using the *Low Delay P* encoding configuration. This mode involves using P slices and, thus, biprediction is not taken into consideration. As a result, both GPU-based algorithms achieve higher speed-ups. The rest of the configuration parameters remain equal to the ones provided in [1].

Our proposed heterogeneous architecture performs the fractional motion estimation module on the host, as opposed to [13], which carries it out on the device. Therefore, the speed-up results provided in Table 4 are not completely comparable. Nevertheless, the fractional motion search consists on executing the same algorithm once again on a

Table 3 Speed-up and BD-rate results comparison between the GOP-based parallelization technique and its corresponding joint algorithm

Class	Video sequence	Speed-up						BD-rate (%)	
		2 threads		4 threads		4 th. + SMT			
		GOP	w/GPU	GOP	w/GPU	GOP	w/GPU	GOP	w/GPU
A	Traffic	1.95	2.05	3.60	3.80	4.10	4.19	0.0	0.3
	PeopleOnStreet	1.96	2.24	3.66	4.16	4.16	4.73	0.1	−0.6
B	Kimono	1.94	2.18	3.44	3.95	3.95	4.48	2.3	2.8
	ParkScene	1.93	2.06	3.61	3.86	4.09	4.38	1.5	1.6
	Cactus	1.96	2.14	3.68	4.03	4.21	4.61	−0.5	−0.4
	BasketballDrive	1.96	2.28	3.60	4.16	4.08	4.78	0.4	4.0
	BQTerrace	1.96	2.07	3.64	3.88	4.07	4.33	−0.9	−1.9
C	BasketballDrill	1.96	2.19	3.70	4.13	4.26	4.75	0.3	−0.1
	BQMall	1.94	2.11	3.43	3.81	3.95	4.38	0.4	1.2
	PartyScene	1.95	2.09	3.53	3.83	4.01	4.41	0.9	0.7
	RaceHorses	1.93	2.27	3.54	4.20	4.06	4.71	−0.5	−0.4
D	BasketballPass	1.85	2.12	3.42	3.95	3.98	4.55	0.4	0.5
	BQSquare	1.95	2.02	3.67	3.81	4.16	4.30	0.8	0.6
	BlowingBubbles	1.87	1.98	3.40	3.64	3.73	4.04	1.8	1.4
	RaceHorses	1.92	2.19	3.52	4.03	4.11	4.60	−0.1	−0.6
F	BasketballDrillText	1.97	2.17	3.68	4.08	4.24	4.70	0.4	−0.4
	ChinaSpeed	1.95	2.21	3.64	4.15	4.09	4.72	−0.2	−3.2
	SlideEditing	1.96	2.00	3.60	3.67	4.02	4.16	−1.0	2.1
	SlideShow	1.86	1.96	3.23	3.44	3.45	3.75	2.0	6.8
	Class A	1.95	2.14	3.63	3.98	4.13	4.46	0.0	−0.2
	Class B	1.95	2.15	3.59	3.98	4.08	4.52	0.6	1.2
	Class C	1.95	2.16	3.55	3.99	4.07	4.56	0.3	0.3
	Class D	1.90	2.08	3.50	3.86	4.00	4.37	0.7	0.5
	Class F	1.93	2.09	3.54	3.84	3.95	4.33	0.3	1.3
	Total average	1.94	2.12	3.56	3.93	4.05	4.45	0.4	0.6

Table 4 Comparison between this proposal and the related work

Class	Video sequence	Speed-up		BD-rate (%)	
		Proposal	Radicke et al. [13]	Proposal	Radicke et al. [13]
A	Traffic	1.12	1.92	0.0	1.8
	PeopleOnStreet	1.32	2.07	−1.0	0.6
B	Kimono	1.30	2.15	−0.1	0.7
	ParkScene	1.17	1.94	0.1	1.4

subsampled version of the region referenced by the best motion vector found. In this way, it would be possible to adapt our proposed algorithm to perform this additional operation. As a result, it would be expected to obtain equivalent speed-up results.

The main difference, however, lies in the fact that our proposed algorithm performs a full search, while the algorithm in [13] carries out a diamond search, a similar pattern to the one used as the default algorithm in HM. Taking into account that this pattern focuses the search on the centre positions of the reference area, it might be unable to find the best MV in the absence of motion predictors. On the contrary, a full search is able to obtain the best MV. Since both proposals perform the same fractional motion estimation operation afterwards, it is possible to compare the coding efficiency results shown in Table 4. As can be seen, our algorithm achieves BD-rate savings ranging from 0.8 to 1.8 % compared with the one proposed by Radicke et al. [13].

As a conclusion, the effects of choosing the right search pattern may have a noticeable impact on the coding efficiency results. In this way, a full search is able to obtain the best results with no performance impact on a SIMD architecture such as a GPU.

6 Conclusion and future work

In this paper, we have proposed a GPU-based motion estimation algorithm that, in combination with some coarse-grained parallelization techniques, forms an efficient parallel framework for the HEVC encoder. In this framework, these parallelization techniques are performed on a multi-core CPU, while the GPU carries out the ME operation concurrently. Comparing our approach to some of these techniques (WPP and a GOP-based algorithm), our experiments show that the corresponding joint algorithms achieve better performance in terms of speed-up with negligible coding efficiency penalties. Ongoing work will focus on using multiple GPUs and parallelizing other modules, as well as considering other architectures such as Intel Integrated Graphics or Intel Xeon Phi [6].

References

1. Bossen F (2013) Common test conditions and software reference configurations (Doc. JCTVC-L1100). http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=7281. Accessed 14 May 2013
2. Bossen F, Bross B, Suhring K, Flynn D (2012) HEVC complexity and implementation analysis. IEEE Trans Circuits Syst Video Technol 22(12):1685–1696. doi:10.1109/TCSVT.2012.2221255
3. Bross B, Han W, Ohm J, Sullivan G, Wang YK, Wiegand T (2013) High efficiency video coding (HEVC) text specification draft 10 (Doc. JCTVC-L1003). http://phenix.it-sudparis.eu/jct/doc_end_user/current_document.php?id=7243. Accessed 21 March 2013
4. Cheung NM, Fan X, Au O, Kung MC (2010) Video coding on multicore graphics processors. IEEE Signal Process Mag 27(2):79–89. doi:10.1109/MSP.2009.935416
5. Chi CC, Alvarez-Mesa M, Juurlink B, Clare G, Henry F, Pateux S, Schierl T (2012) Parallel scalability and efficiency of HEVC parallelization approaches. IEEE Trans Circuits Syst Video Technol 22(12):1827–1838. doi:10.1109/TCSVT.2012.2223056
6. Fang J, Varbanescu AL, Sips H (2013) Identifying the key features of Intel Xeon Phi: a comparative approach. Parallel and Distributed Systems Report Series PDS-2013-006, Delft University of Technology
7. Wc Feng, Manocha D (2007) High-performance computing using accelerators. Parallel Comput 33(10–11):645–647

8. Henry F, Pateux S (2011) Wavefront Parallel Processing (Doc. JCTVC-E196). http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=2122. Accessed 25 March 2013
9. ITU-T, ISO/IEC JTC (2012) Information technology—coding of audio-visual objects—part 10: advanced video coding. ITU-T Recommendation H.264 and ISO/IEC 14496–10
10. JCT-VC (2013) HM reference Software. https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/. Accessed 23 April 2013
11. Misra K, Segall A, Horowitz M, Xu S, Fuldseth A, Zhou M (2013) An overview of Tiles in HEVC. IEEE J Sel Top Signal Process 7(6):969–977. doi:10.1109/JSTSP.2013.2271451
12. 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. doi:10.1109/TCSVT.2012.2221192
13. Radicke S, Hahn J, Grecos C, Wang Q (2014) A highly-parallel approach on motion estimation for high efficiency video coding (HEVC). In: IEEE international conference on consumer electronics (ICCE), 2014, pp 187–188. doi:10.1109/ICCE.2014.6775965
14. Su H, Wu N, Zhang C, Wen M, Ren J (2011) A multilevel parallel intra coding for H.264/AVC based on CUDA. In: Sixth international conference on image and graphics (ICIG), 2011, pp 76–81. doi:10.1109/ICIG.2011.99
15. Wang X, Song L, Chen M, Yang J (2013) Paralleling variable block size motion estimation of HEVC on CPU plus GPU platform. In: IEEE international conference on multimedia and expo workshops (ICMEW), 2013, pp 1–5. doi:10.1109/ICMEW.2013.6618412
16. Yan C, Dai F, Zhang Y (2011) Parallel deblocking filter for H.264/AVC on the TILERA many-core systems. In: Advances in multimedia modeling, lecture notes in computer science, vol 6523, Springer, Berlin, pp 51–61. doi:10.1007/978-3-642-17832-0_6
17. Yan C, Zhang Y, Dai F, Li L (2013) Highly Parallel Framework for HEVC motion estimation on many-core platform. In: Data compression conference (DCC), 2013, pp 63–72. doi:10.1109/DCC.2013.14
18. Yu Q, Zhao L, Ma S (2012) Parallel AMVP candidate list construction for HEVC. In: Visual communications and image processing (VCIP), 2012 IEEE, pp 1–6. doi:10.1109/VCIP.2012.6410775

CHAPTER 3

Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding

- **Title**: Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding.
- **Authors**: Antonio Jesús Díaz-Honrubia, Gabriel Cebrián-Márquez, José Luis Martínez, Pedro Cuenca, José Miguel Puerta, and José Antonio Gámez.
- **Type**: journal paper.
- **Journal**: Journal of Real-Time Image Processing.
- **Publisher**: Springer Berlin Heidelberg.
- **ISSN**: 1861-8200.
- **Status**: published.
- **Publication date**: August 2016.
- **Volume**: 12.
- **Issue**: 2.
- **Pages**: 311–327.
- **DOI**: 10.1007/s11554-014-0477-z
- **JCR IF/ranking**: 2.010/Q2 (JCR2016).

J Real-Time Image Proc (2016) 12:311–327
DOI 10.1007/s11554-014-0477-z

SPECIAL ISSUE PAPER

Low-complexity heterogeneous architecture for H.264/HEVC video transcoding

Antonio Jesús Díaz-Honrubia · Gabriel Cebrián-Márquez · José Luis Martínez · Pedro Cuenca · José Miguel Puerta · José Antonio Gámez

Received: 31 July 2014 / Accepted: 30 November 2014 / Published online: 13 December 2014

Abstract High efficiency video coding (HEVC) was developed by the Joint Collaborative Team on video coding to replace the current H.264/AVC standard, which has been widely adopted over the last few years. Therefore, there is a lot of legacy content encoded with H.264/AVC, and an efficient conversion to HEVC is needed. This paper presents a hybrid transcoding algorithm which makes use of soft computing techniques as well as parallel processing. On the one hand, a fast quadtree level decision algorithm tries to exploit the information gathered at the H.264/AVC decoder to make faster decisions on coding unit splitting in HEVC using a Naïve–Bayes probabilistic classifier that is determined by a supervised data mining process. On the other hand, a parallel HEVC-encoding algorithm makes use of a heterogeneous platform composed of a multi-core central processing unit plus a graphics processing unit (GPU). In this way, from a coarse point of view, groups of frames or rows of a frame (both options are possible) are divided into threads to be executed on each core (each of which executes one of the aforementioned classifiers) and, from a finer point of view, all these threads work in a collaborative way on a single GPU to perform the motion estimation process on the co-processor. Experimental results show that the proposed transcoder can achieve a good tradeoff between coding efficiency and complexity compared with the anchor transcoder.

Keywords HEVC · H.264/AVC · Transcoding · Heterogeneous computing · CTU splitting

1 Introduction

Recently, the new HEVC standard [5] has been established by the JCT-VC, an expert group proposed by the ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG). HEVC is based on a well-known block-based hybrid video coding architecture as well as its predecessor, namely H.264/MPEG4 part 10—advanced video coding (AVC) [17], which it outperforms in terms of bit rate reduction at the same quality [22], making it suitable for new applications such as resolutions beyond high definition (HD) (so-called 4K, 3840 × 2160 pixels, and 8K, 7680 × 4320 pixels). Among others, 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) [4, 22].

This work has been jointly supported by the MINECO and European Commission (FEDER funds) under the projects TIN2012-38341-C04-04, TIN2010-20900-C04-03 and TIN2013-46638-C3-3-P. Likewise, this work has also been supported by the Spanish Ministry of Education, Culture and Sports under Grants FPU 12/00994 and FPU 13/04601.

A. J. Díaz-Honrubia (✉) · G. Cebrián-Márquez · J. L. Martínez · P. Cuenca · J. M. Puerta · J. A. Gámez
Albacete Research Institute of Informatics (I3A), University of Castilla-La Mancha, 02071 Albacete, Spain
e-mail: antonio.dhonrubia@uclm.es

G. Cebrián-Márquez
e-mail: Gabriel.Cebrian@uclm.es

J. L. Martínez
e-mail: JoseLuis.Martinez@uclm.es

P. Cuenca
e-mail: Pedro.Cuenca@uclm.es

J. M. Puerta
e-mail: Jose.Puerta@uclm.es

J. A. Gámez
e-mail: Jose.Gamez@uclm.es

Considering both the superior compression performance of HEVC, as well as 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 [7]. Furthermore, there is a wide availability of H.264/AVC encoders in 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 rate and distortion (RD) performance of the HEVC standard.

Thus, an H.264/AVC to HEVC transcoder could be applied in two scenarios: in real-time applications which are encoding in H.264/AVC and broadcasting the stream to HEVC decoders (among others), and films already encoded in H.264/AVC which must be broadcast to HEVC decoders. Both scenarios would need to be refreshed occasionally to be error resilient and to provide random access.

However, all the aforementioned new coding tools imply a considerable increase in the encoding time in HEVC. With this challenge in mind, this paper presents a low complexity heterogeneous H.264/HEVC video transcoder architecture that greatly reduces the complexity of the transcoding process. On the one hand, the computational cost is efficiently reduced by adapting the sequential encoding algorithm to parallel architectures. On the other hand, a statistical algorithm which makes use of the already H.264/AVC coded information is also used to accelerate the transcoder.

With respect to the parallelism, over the last few years the computer industry has tended towards including several processing units on a single shared chip. Furthermore, in terms of massive data computations, there are also devices called graphics processing units (GPUs). These devices, also referred to as many core, are highly parallel and are normally used as co-processors to assist the central processing unit (CPU). CPUs and GPUs have different instruction set architectures, forming what it is known as a heterogeneous computing platform [10]. In support of this parallelism, HEVC places special emphasis on hardware-friendly design and parallel-processing architectures. These parallelization approaches are *Tiles* [20] and wavefront parallel processing (WPP) [16], which will be described in Sect. 2. Basically, these parallelization approaches considered in HEVC rely on creating picture partitions and, thus, coding losses may appear due to breaking dependencies for prediction, CABAC context modeling, and/or slice header overhead.

At this point, this paper proposes a heterogeneous video transcoding platform on which two different levels of parallel algorithms can be executed. The first one makes use of a multi-core processor to map different groups Of pictures (GOPs) or rows from a single frame onto different cores; the second one makes use of the GPU to efficiently parallelize the motion estimation carried out in the HEVC Inter prediction algorithm. As these algorithms are independent of one another, they can be combined into a single approach. In this way, a GPU-based parallel architecture is utilized in this paper.

As mentioned above, as part of the proposed heterogeneous video transcoding platform this paper also presents a soft computing approach which tries to exploit the information gathered in the H.264/AVC decoder to assist decisions on CU splitting in HEVC using a statistical Naïve–Bayes (NB) classifier to avoid an exhaustive Rate-Distortion Optimization search over all possible CU sizes and its modes. In an offline data mining process, all the knowledge needed is extracted from the H.264/AVC decoding statistics by means of *Machine Learning* techniques, and it is then converted into mathematical models which can be executed in the online transcoding process.

Experimental results show that the combination of the algorithms proposed in this paper executed over the proposed H.264/AVC to HEVC architecture can achieve a speedup in the video transcoder of up to 7.10× when run on a quad-core Intel Core i7-2600 CPU, without significant loss in RD performance, which also represents an improvement upon some of the related works available in the literature and other parallel techniques included in the HEVC standard [5].

The remainder of this paper is organized as follows: Sect. 2 includes some technical background to the new HEVC standard, while Sect. 3 identifies the related work which is being carried out on the topic. Section 4 introduces our proposed transcoder architecture and experimental results are given in Sect. 5. Section 6 concludes the paper.

2 Technical background

As mentioned in the previous section, the main target of HEVC is to achieve lower bitrates for video streams while maintaining the same quality. To make this possible, HEVC introduces new coding tools with respect to its predecessor, H.264/AVC, as well as improving upon others

which were already used [22, 28]; all of them make it possible to notably increase coding efficiency. One of the most important changes affects picture partitioning. HEVC dispenses with the terms macro-block (MB) and Block for the ME and the transform, respectively, and introduces three new concepts: CU, PU and TU. This structure leads to more flexible coding to suit the particularities of the frame. 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 from 8 $\times$ 8 to 64 $\times$ 64 pixels, may contain one or several PUs and TUs. To fix the size of each CU, a picture is divided into 64 $\times$ 64 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 8 $\times$ 8 pixels, as shown in Fig. 1.

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. 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 its predecessor, 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), the PU modes (it introduces asymmetric modes), new image filters or new transform sizes [22, 28], among others.

2.1 Parallelization techniques in HEVC

Previous video codecs, in particular H.264/AVC, have been parallelized using either frame-level, slice-level, or MB-level parallelism. Each of these approaches, however, has some limitations such as limited scalability, significant coding losses, or high memory requirements. To overcome these limitations, the new HEVC codec includes new parallelization techniques such as Tiles [20] and WPP [16] among slices. On the one hand, Tiles are square or rectangular-shaped partitions where dependencies are broken across Tile boundaries [20], entailing, hence, substantial coding losses, while making it possible to process them independently. Nevertheless, the deblocking and *Sample Adaptive Offset* filters can still cross these boundaries. The number of Tiles and their location can be defined for the whole sequence or changed from picture to picture.

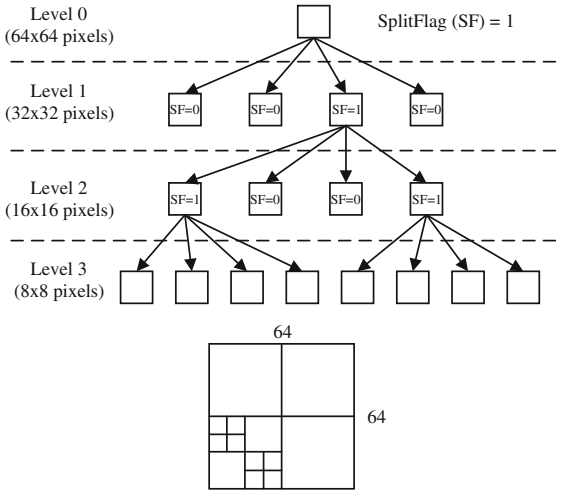


Fig. 1 Quadtree splitting illustration

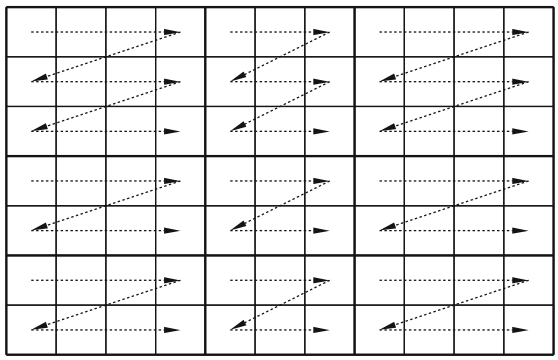
On the other hand, WPP allows the pictures to be partitioned in rows, each of which can be processed in parallel, whereas entropy encoding and prediction are allowed to cross partitions to minimize coding losses. However, coding dependencies make it necessary to have a delay of, at least, two CUs between consecutive rows in a similar way to segmentation in a computer architecture [16]. For this reason, not all the processes can start encoding these rows at the same time, which introduces the so-called "ramping inefficiencies" at the beginning and at the end of a frame, not allowing a parallel encoder/decoder to reach its peak performance. Both techniques, Tiles and WPP, are depicted in Fig. 2.

Tiles and WPP have different merits and disadvantages. While WPP is generally well suited for the parallelization of the encoder and the decoder due to its high number of picture partitions with low compression losses, the amount of parallelism with Tiles is not fixed, as the number of regions in which a frame is divided may vary. Additionally, WPP does not introduce artifacts at partition boundaries as is the case for Tiles. To simplify the implementation, it is not possible to use Tiles and WPP simultaneously in the same compressed video sequence.

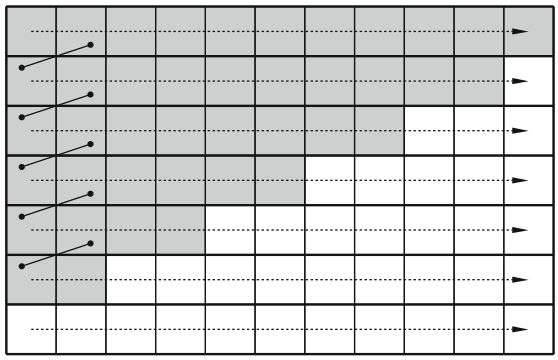
3 Related work

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 characteristics. The transcoding process should perform the conversion without making it necessary to perform the complete process of decoding and re-encoding [29]. Transcoding has been a hot research topic in the last few years in the framework of MPEG-2 to H.264/AVC [12] or H.263 to H.264/AVC transcoders [11], and also between H.264-extensions such as H.264/AVC to SVC [14] or, even, between *Distributed Video Coding* and H.264/AVC [8].

As far as the authors of this paper know, there are currently few approaches that deal with the problem of



(a) Tiles partitioning



(b) WPP partitioning

Fig. 2 Partitioning and processing order of Tiles (**a**) and WPP (**b**)

converting streams already encoded in H.264/AVC into the new HEVC standard. The approach presented in [34] focuses on reducing the number of CU and PU partitions to be checked by means of an improved rate-distortion optimization metric. In [24], the authors proposed the reuse of Motion Vectors as well as a similarity metric to decide which HEVC CU partitions should be tested.

In [25], 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 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).

Regarding parallelization, as far as related work in the literature is concerned, in the past there have been many approaches focusing on accelerating different modules of the H.264/AVC-encoding algorithm by means of parallel computing [6, 27, 31]. On the contrary, in the framework of HEVC, the first parallel approaches were focused on reducing the complexity of the decoding algorithm; in [7], the authors improve the WPP approach included in the HEVC Test Model (HM) reference software [1]. The idea consists of once there are no available rows in the current picture, the next one starts being processed. This proposal was called *Overlapped Wavefront* and it might work for the encoder, but no results were given in [7].

In the context of this paper, which is focused on the encoder side (since it is the side which is included in the H.264/HEVC transcoder), there are not many approaches. Yu et al. proposed in [33] a parallel candidate list to parallelize the motion vector prediction, but the proposal is not standard compliant. Later, in [32], the authors reduced the encoding time by up to 13 times using a 64-core architecture, which is far more expensive than the one used in this paper. Finally, Wang et al. proposed in [30] a scheme similar to the one proposed in this paper, based on a GPU plus multi-core CPU, but the major weakness of this paper lies in the fact that they did not use the reference software HM [1] and, thus, the RD results are worse due to the fact that not all coding tools were implemented [30].

Other possibility would be using the straightforward idea of exploiting the independence given by a GOP encoding pattern in which an Intra frame is encoded periodically [19]. This pattern does not allow Motion Vectors to reference frames across the I frame, so this technique would be able to execute each GOP on a different core. In this way, parallel encoding without any RD penalty could be achieved due to the fact that GOPs are completely independent. Moreover, this architecture would be scalable to any number of cores, obtaining an almost linear speedup even when using a higher number of threads. This proposal will be called the *GOP Independence Parallel Technique* (GIPT) in the remainder of the paper.

Finally, in [26] a proposal which combines parallelization and reutilization of information fetched from the H.264/AVC decoder can be found. The first part of this proposal simply uses WPP for parallelization, while the second part uses the frame resolution to restrict the quadtree splitting and it reuses the partitioning modes to select the available PU modes in each case. Furthermore, in a third part of the proposal, the authors implement some of the most commonly used functions in the HEVC encoder using assembly code.

4 Proposed architecture

In this paper, a novel architecture for accelerating the H.264/AVC to HEVC transcoder is proposed that is based on a combination of a Bayesian Classifier and CPU+GPU parallelization. Regarding the classifier, the simplest one (in

terms of computational complexity) is used: a Naïve-Bayes classifier (see e.g. [13]). The existence of a correlation between some information from H.264/AVC (residual, motion vectors, modes, among others) and HEVC partitions is considered to learn a classifier to be used for the selection of the best CU partition. This technique converts a very complex process into a simple set of multiplications of probabilities, which significantly reduces the complexity of the HEVC encoder, as shown by the results presented.

On the other hand, and as seen before, parallelization is possible in both the encoder and the decoder using the algorithms defined in the standard or other straightforward techniques. Nonetheless, these are designed to be executed in a multi-core CPU, taking advantage of the capabilities that multiple threads may offer, but not taking into account other devices. Heterogeneous architectures such as the ones formed by the association of a multi-core CPU and a GPU are utilized in this paper, making use of the immeasurable power they can provide. This paper proposes an approach to parallelize the HEVC encoding algorithm based on multi-core processing and on a GPU architecture.

Hence, this paper proposes two algorithms, namely the Bayesian Classifier one and the GPU-based one which are independent of one another. The final proposed architecture combines them with WPP, which was described in Sect. 2.1, and GIPT, the technique which exploits the GOP independence, which was described in Sect. 3. It is not difficult to see that both CPU parallel algorithms are also independent from the GPU and Bayesian Classifier ones. So, in the overall architecture, first, one of the multi-core techniques perform a coarse-grained parallelization of the whole encoding process, then the classifier accelerates the quadtree partitioning and, finally, the GPU carries out the *Integer ME* operation. Thus, this independence makes it possible to combine the three algorithms in a single proposal, obtaining, hence, higher speedup values for the overall proposed architecture at the expense of a negligible increment in coding efficiency losses.

More specifically, Figs. 3 and 4 depict the whole architecture when using WPP and GIPT, respectively. Specifically, in Fig. 3 every frame is partitioned into rows and in Fig. 4 the whole stream is partitioned into sub-streams. Then, each of these rows or sub-streams are encoded in a different thread. After that, each thread makes use of an M_i classifier, which is explained in Sect. 4.1. Finally, a single GPU device can carry out the Integer ME of multiple threads and, hence, multiple CTUs, as shown in Sect. 4.2. Therefore, it is necessary to queue several kernels into the GPU. In this way, the device is fully utilized, lowering idle times. In addition, the GPU can process different kernels independently of the CPU, so the latter can continue processing other modules concurrently.

4.1 Bayesian CTU splitting algorithm

This algorithm aims to reduce the computational complexity of deciding the appropriate depth for each quadtree. The proposed algorithm has an incremental design, so that for each level the algorithm decides whether it is more likely 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. Therefore, C_S and C_N are the labels of the class variable to be predicted by the proposed approach.

Moreover, 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 and all PUs are checked for level 2 of deciding, otherwise all PUs at this CU depth are evaluated and the algorithm for this CTU finishes. Thus, if some of the RD costs of higher levels are better than the best RD cost for the final level, the quadtree is allowed to go back to the best one among those calculated. Figure 5 schematically describes this CU splitting algorithm, which we have called fast quadtree level decision (FQLD).

After defining the global behavior of the proposed algorithm, the models for making a decision (M_{level}) must be defined too. In this study we focus on the use of machine learning (supervised classification) at levels 0 and 1, while at level 2 a much simpler approach is followed. Basically, as CU size at this level is 16 $\times$ 16 pixels is the MB size in H.264/AVC too, the proposed algorithm simply mimics H.264/AVC: if MB size was 16 $\times$ 16, 16 $\times$ 8 or 8 $\times$ 16, then the decision would be C_N, otherwise (smaller sizes), the decision would be C_S.

At levels 0 and 1 of the quadtree (CU sizes of 64 $\times$ 64 and 32 $\times$ 32 pixels, respectively), probabilistic classifiers are learnt and used to make the decision. In particular, a Naïve–Bayes classifier has been selected since this model is computationally efficient while achieving a high hit rate. A Naïve–Bayes classifier computes the posterior probability of each label C_i given the set of features $\mathbf{F} = \{\mathbf{w_1}, \ldots, \mathbf{w_n}\}$ as input: $P(C_i|\mathbf{F}) \propto \mathbf{P(C_i)} \prod_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{P(w_j|C_j)}$, and it chooses the output label as one with high probability.

Four Naïve-Bayes classifiers, for levels 0 and 1 and frame types P and B, (M_{0P}, M_{1P}, M_{0B} and M_{1B}) have been trained offline using $QP = \{22, 27, 32, 37\}$ and six sequences from those described in [3] (*PeopleOnStreet, ParkScene, BQMall, RaceHorses, Johnny* and *SlideEditing*). The first 1,000 CUs, after skipping the first frame (since it is coded as Intra), for each QP-sequence pair were selected for this learning process.

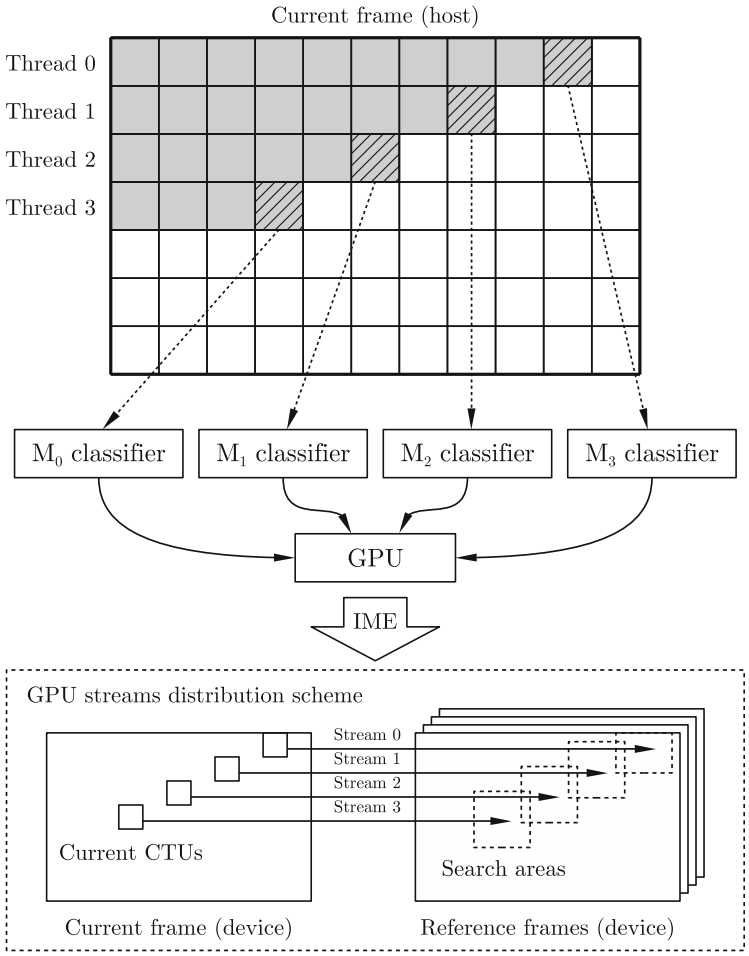


Fig. 3 Proposed transcoder architecture block diagram with WPP

For the initial set of features, $\mathbf{F} = \{\mathbf{w_i}\}$, each feature was fetched from the H.264/AVC stream, and 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, this initial set of features, **F**, is:

- 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 16 × 16, Inter 4 × 4 and other Inter size 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 Motion Vector components contained in the frame.
- w_{resAvg}, w_{resVar}: average and variance of the residue for the covered area, 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 above 4 values.
- w_{sobelH}, w_{sobelV}: sum of applying the Sobel operator [23] to the residue in horizontal and vertical directions.
- w_{MVxAvg}, w_{MVyAvg}, w_{MVxVar}, w_{MVyVar}: average and variance of x and y Motion Vector components, respectively, for the covered area.

Starting from this original set of features, four independent machine learning processes are carried out to learn the corresponding models, M_{0P}, M_{1P}, M_{0B} or M_{1B}, as mentioned above. The steps of the learning process are:

1. To avoid the standard (and improbable) assumption that the values of each feature given in each class label follow a parametric distribution (e.g. a Normal

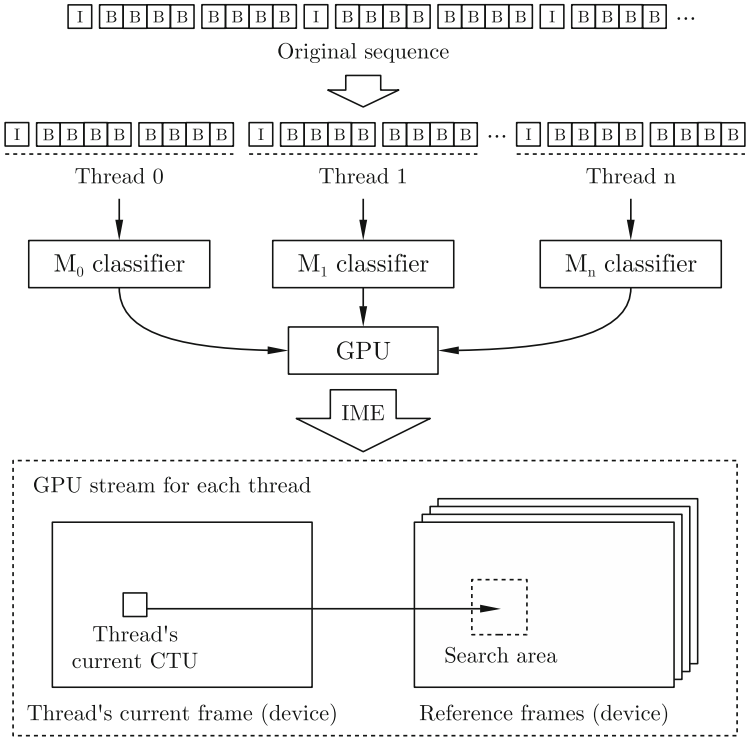


Fig. 4 Proposed transcoder architecture block diagram with GIPT

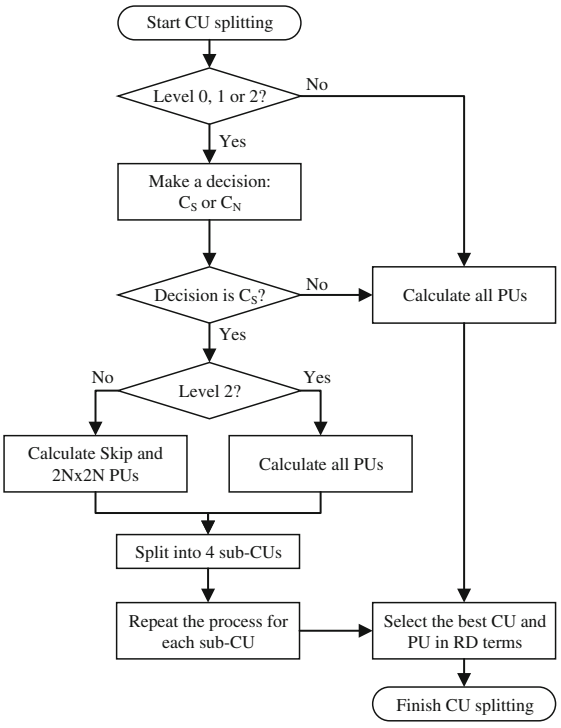


Fig. 5 Diagram of the proposed FQLD algorithm

distribution), we discretize all the numerical variables using a supervised method [9], that is, the intervals are chosen in such a way that the resulting *discrete* variable has as much discriminative power as possible regarding the class variable.

2. After that, a wrapper stepwise forward attribute selection process [15] is carried out to select the *best* subset of features according to the prediction of the class labels (C_S and C_N). An Naïve-Bayes algorithm is used during the search to evaluate the goodness of each subset, removing those redundant and irrelevant variables that may reduce its accuracy.
3. From the resulting discretized and reduced dataset a Naïve-Bayes classifier is learnt, and subsequently calibrated to accommodate the particularities of the addressed problem. That is, since 64×64 and 32×32 CUs are not usually chosen, a cost analysis has been carried out by increasing the cost of choosing C_N when it was really C_S. This cost has been set based on the quadtree level and the resolution (see Table 1)[1]: higher resolutions tend to have higher CU sizes and viceversa.

[1] A value of 2.0 means that we have estimated that the cost of wrongly classifying C_S as C_N is twice that of the contrary error. This type of error at high levels (0 or 1) can have a great impact on the output sequence, because subsequent low-level splitting is not considered.

Table 1 Costs of wrongly choosing C_N at levels 0 and 1 (rest of costs of misclassifying are 1, while the cost of correct classification is 0)

	Res. $<$ Full HD	Res. $\geq$ Full HD
Level 0	2.0	2.0
Level 1	2.0	1.0

The output from this analysis is the threshold to be used in the classification, instead of the standard 0.5 one.

4.2 GPU-based inter prediction Algorithm

On the hardware side, new architectures that are being introduced in high-performance computing are composed of multi-core CPUs and GPUs. A multi-core processor is composed of several processors sharing the same chip, while GPUs are composed of hundreds of similar simple processing cores which are designed and organized with the goal of achieving high performance. These cores are grouped in stream processors that perform single instruction multiple data (SIMD) operations, which are suitable for arithmetic intensive applications. In the particular case of NVIDIA, a powerful GPU architecture called the compute unified device architecture (CUDA) [21] has been developed. The main feature of these devices is a large number of processing elements integrated into a single chip at the expense of a significant reduction in cache memory.

As Motion Estimation is the most resource intensive operation on the encoder side [22], this algorithm aims to reduce the time spent on the CPU by performing these searches on a GPU device. Nevertheless, taking into account that data transfers between host and GPU are highly time consuming, these operations are performed asynchronously. In this way, time spent on *Integer ME* is negligible compared to the default search algorithm.

As soon as a GOP starts being processed, it is possible to transfer the original frames that will be encoded to the device, making them available for subsequent uses. Later on, these frames are updated with their reconstructed version when they are encoded (and decoded in-loop) to correctly carry out the ME on the device.

When the encoder starts processing a slice, the host queues the execution of two consecutive kernels that perform the Integer ME of every PU partition in the first CTU. The first kernel executes the required operations to calculate the sum of absolute differences (SAD) residuals across a search area in the reference frame, while the second one determines which one of them may offer the best possible result (Fig. 6).

This algorithm relies on the fact that every PU size established by the standard is divisible by four, and taking into account the nature of the SAD operation, it is possible to calculate the residual information of a PU partition from the composition of its 4×4 SAD partitions.

Following this approach, the previously mentioned kernel distributes a device thread per sample in the reference search area. Every thread is responsible for calculating all the 4×4 SAD blocks in a CTU, taking as motion vector its position in the search area. Once these blocks are calculated, all the running threads put them together to obtain the PU partitions in which a CTU might be divided. From another point of view, the results of this step would be equivalent to a full-search algorithm performed for every PU partition, as shown in Fig. 7.

At this point, the second kernel performs a reduction algorithm over the residual data obtained from the first one, so that the result of the GPU algorithm is a single table containing the best Motion Vector for every PU partition, which is copied asynchronously to the host. After the transfer is finished, motion search operations related to the next CTU are then issued to the device.

By the time the host needs to perform the motion estimation of the CTU, integer Motion Vectors should be ready to be required, being only necessary to perform fractional motion estimation of the PU partitions which have not been skipped by the encoder.

5 Performance evaluation

This section aims to evaluate the heterogeneous architecture presented in this paper by showing the performance of the algorithms described. First, the results of each proposal working separately will be shown and, then, the results for the overall proposed transcoder will also be shown. To ensure a common framework, *Joint Collaborative Team on Video Coding* defined a document in [3] 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 which were used are $\{22, 27, 32, 37\}$, and the configuration was random access main (RA) (since, as mentioned in Sect. 1, real scenarios need to be refreshed occasionally, and other Inter configurations do not fulfill this requirement). As GOPs need to be fully independent of one another for GIPT, *Instantaneous Decoding Refresh* access points were chosen, that is, dependencies cannot reference frames across an I frame.

The results are shown for each sequence, as well as the average value for all of them. Moreover, the sequences are grouped as described in [3], according to their resolution:

– Class A (2560×1600 pixels): *Traffic* and *PeopleOnStreet* sequences.

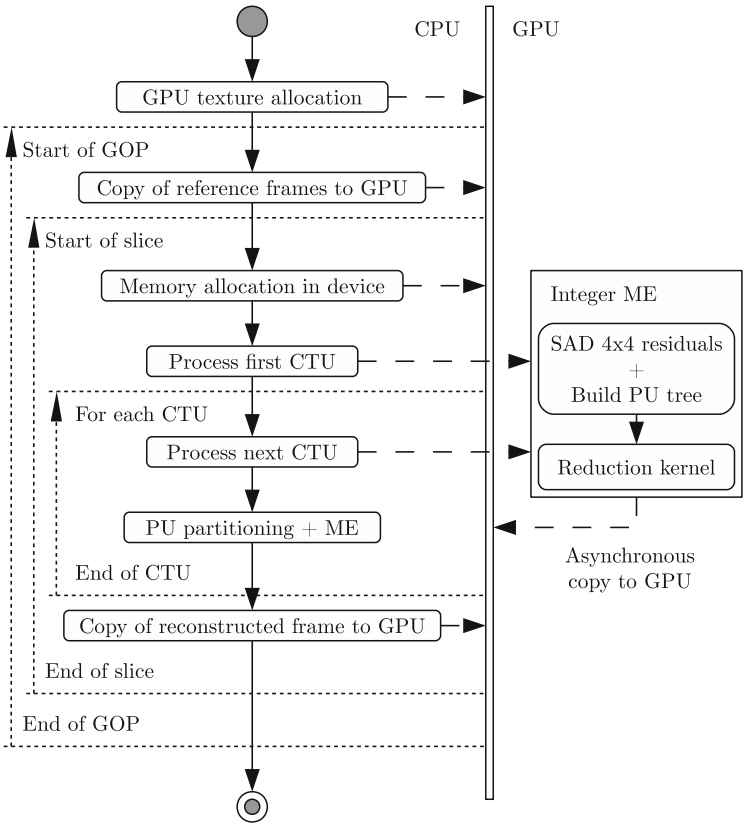


Fig. 6 GPU-based algorithm activity diagram

- Class B (1920 × 1800 pixels): *Kimono*, *ParkScene*, *Cactus*, *BQTerrace* and *BasketballDrive* sequences.
- Class C (832 × 480 pixels): *RaceHorsesC*, *BQMall*, *PartyScene* and *BasketballDrill* sequences.
- Class D (416 × 240 pixels): *RaceHorses*, *BQSquare*, *BlowingBubbles* and *BasketballPass* sequences.

The software used is JM 18.4 [18] for H.264/AVC and HM 10.0 [1] for HEVC. The remainder of the 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 which are included in the reference software JM 18.4 [18].
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 to obtain the BD-rate and the speedup.

All measurements have been performed on a quad-core Intel Core i7-2600 CPU running at 3.40GHz and a NVIDIA GTX 560 Ti GPU running 384 CUDA cores at the frequency of 1.6 GHz. Consequently, tests for parallel algorithms have been carried out with 2 and 4 threads, as well as 4 plus *Simultaneous MultiThreading* (SMT), enabling the processor to execute up to 8 threads. However, since the FQLD algorithm is not parallel by itself, tests which only involve this algorithm have been carried out with only 1 thread.

All the results are presented in terms of speedup and BD-rate [2], which measures the increment in bitrate while maintaining the same objective quality. 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 (1) and (2), respectively, where t_{anchor} is the execution time of the encoder side of the non-accelerated reference transcoder, and $t_{proposed}$ is the execution time of the encoder side of the proposed transcoder. The BD-rate is the weighted average of the Y, U and V components as shown in (3) (given that the Y component is four times larger than the U and V components, since the video format is 4:2:0).

$$\text{Speedup} = \frac{\text{t}_{\text{anchor}}}{\text{t}_{\text{proposed}}} \tag{1}$$

$$\text{Timereduction}(\%) = \frac{\text{t}_{\text{anchor}} - \text{t}_{\text{proposed}}}{\text{t}_{\text{anchor}}} \times 100 \tag{2}$$

$$\text{BD} - \text{rate}(\%) = \frac{4 \times \text{BD} - \text{rateY} + \text{BD} - \text{rateU} + \text{BD} - \text{rateV}}{6} \tag{3}$$

5.1 Results for GIPT and WPP

To start with, Table 2 shows the results for WPP and GIPT for 2, 4 and 4 plus SMT threads, respectively. SMT allows the CPU to execute instructions from multiple threads in any stage of the pipeline at the same time. As this feature is usually enabled in most modern CPUs and this paper has the aim of presenting real-world scenarios, the tests shown in this paper kept this technology enabled to see its effectiveness when running eight threads, i.e. twice the number of cores that are present in our test hardware.

Moreover, running eight threads would increase the utilization of the GPU, as more kernels are issued to the device. If one thread needs to wait for a result from the device (if necessary), a second thread can continue executing other modules of the encoding. As a result, the percentage of effective utilization of both the CPU and the GPU increases.

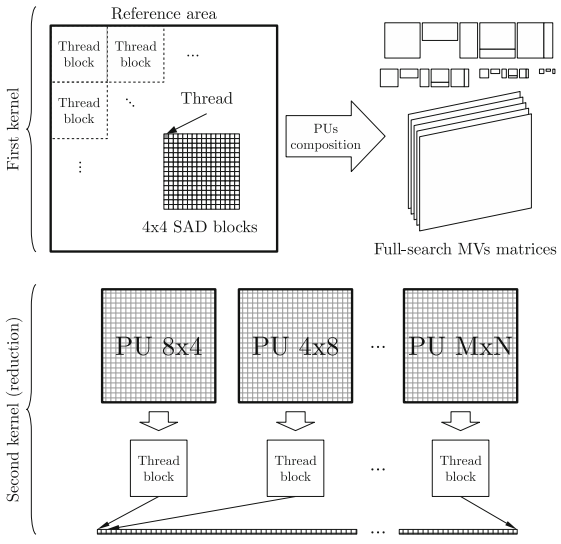


Fig. 7 GPU kernel operations to perform Integer ME

As can be seen in the table, both algorithms reach speedup values close to the ones from a parallel efficient algorithm, i.e. threads are almost fully utilized regardless of the class, while coding losses remain bellow 1.0 % in most cases and they do not exceed 2.3 % in any case. Specially, GIPT works very well, since its results already surpass the ones of the WPP algorithm, since WPP speedup values are smaller than those of GIPT while, at the same time, it achieves a BD-rate reduction greater than half that of WPP. Specifically, it means an improvement of up to 31.5% of the speedup and a difference of up to −0.6 % of the BD-rate compared to WPP.

Furthermore, it can be seen that WPP efficiency is proportionally worse when the number of threads is incremented. Moreover, the smaller the frame size is, the greater the advantage GIPT obtains, since WPP speedup values in those cases are much smaller due to the fact that there are few rows to process in parallel, while GIPT maintains the same performance in all the cases.

However, GIPT would not obtain good results when the GOP size is too large, the sequence is too short or when an *Instantaneous Decoding Refresh* picture cannot be included. For example, when evaluating class A in Table 2, it can be seen that the results for 4 threads + SMT (a whole of 8 virtual threads) are the same than for 4 threads since the length of the sequence is too short and the GOP size too large (32 frames) to allocate parts of the sequence beyond the fourth thread.

5.2 Results for the FQLD Proposal

Table 3 contains the results for the FQLD proposal. This table shows the difference made by applying the FQLD algorithm to incremental levels 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 could achieve a moderate speedup without a significant loss. It can also be seen that from level 1 to level 2 the BD-rate does not increase significantly, since, in this last level, the algorithm copies the results from H.264/AVC which already performed the ME. Therefore, the quality-complexity could be adjusted by the user deciding whether to apply the FQLD algorithm to 1, 2 or 3 levels depending on the requirements of the application.

In particular, the algorithm applied to the three levels shows a speedup of 2.24× (more than a 55 % of encoding time reduction) on average, while the loss is only 4.5 % in BD-rate terms. Specifically, it can be seen that the higher

Table 2 Speedup and coding efficiency results of the GIPT and WPP

	BD-rate (%)		Speedup					
			2 Threads		4 Threads		4 Threads + SMT	
	GIPT	WPP	GIPT	WPP	GIPT	WPP	GIPT	WPP
Class A								
Traffic	−0.2	0.7	1.96	1.89	3.65	3.37	3.65	3.89
PeopleOnStreet	0.0	0.8	1.95	1.88	3.65	3.34	3.65	3.93
Class B								
Kimono	2.3	1.2	1.94	1.89	3.44	3.36	3.95	3.80
ParkScene	1.5	0.7	1.93	1.88	3.61	3.33	4.09	3.70
Cactus	−0.5	1.1	1.96	1.89	3.68	3.31	4.21	3.72
BasketballDrive	0.4	1.4	1.96	1.90	3.60	3.40	4.08	3.76
BQTerrace	−0.9	1.1	1.96	1.88	3.64	3.31	4.07	3.79
Class C								
BasketballDrill	0.3	1.4	1.96	1.80	3.70	2.73	4.26	2.72
BQMall	0.4	1.4	1.94	1.81	3.43	2.83	3.95	2.83
PartyScene	0.9	0.6	1.95	1.78	3.53	2.70	4.01	2.71
RaceHorsesC	−0.5	0.8	1.93	1.78	3.54	2.79	4.06	2.84
Class D								
BasketballPass	0.4	0.9	1.85	1.67	3.42	1.75	3.98	1.75
BQSquare	0.9	1.3	1.95	1.60	3.67	1.84	4.16	1.84
BlowingBubbles	1.8	0.9	1.87	1.59	3.40	1.80	3.73	1.80
RaceHorses	−0.1	0.9	1.92	1.63	3.52	1.81	4.11	1.81
Average	0.4	1.0	1.94	1.79	3.57	2.78	4.05	2.99

the sequence resolution is, the better this algorithm works. This is due to the fact that higher resolutions tend to have higher CU blocks and, hence, using FQLD, it is not necessary to explore the deepest levels of the quadtree.

5.3 Results for the GPU-based proposal

Table 4 shows the results of the proposed GPU-based algorithm. As can be seen, performing the Integer ME operation on the GPU involves accelerating the encoding process by 1.12× while incurring in very low coding efficiency losses (due to Motion Vectors prediction), or even improving it in some cases. This happens due to the fact that the proposed algorithm performs a more exhaustive search than the default search algorithm. We would like to emphasize that these results are the theoretical limit of the Integer ME, as the GPU has already calculated every Motion Vector when the host needs to perform the ME. In other words, the Integer ME is performed in a virtually perfect time.

In this case, it can be seen that the results are independent of the resolution of the sequences. This is due to the fact that both, the whole encoding time and the time spent in the ME process, are proportional to the frame size, so the relation between them remains uniform. However, some sequences obtain better results than other (e.g. *BasketballDrive*, *RaceHorsesC* or *RaceHorses*), since the effectiveness of this algorithm does depend on the complexity of the scene. This happens because the proposed algorithm always calculates all the ME steps in the same instant of time while the original algorithm carries out more steps when the complexity of the scene is higher.

5.4 Results for the complete transcoder architecture

Finally, Table 5 contains the overall results for the proposed transcoder architecture combining the multi-core (using either WPP or GIPT), FQLD and GPU-based proposals, using 4 threads (as it is the amount of real parallel threads which can be executed in the used CPU) and with the FQLD algorithm applied to 0, 1 and 2 levels.

On the one hand, if using GIPT as part of the low complexity transcoder architecture, it can be seen that the speedup achieved on the quad-core processor which has been described above is 7.10× on average (which means more than an 85 % time reduction, which is a very considerable fraction of the whole time) and, for some sequences, the increment is close to 8.50×. This time saving has a BD-rate impact of 7.4 % on average, which represents a good tradeoff between time reduction and bitrate increases. In the worst case, the speedup is 6.39×, which still means more than a 84 % of time reduction.

Table 3 Speedup and coding efficiency results of the FQLD proposal

	BD-rate (%)			Speedup		
	Level 0	Level 1	Level 2	Level 0	Level 1	Level 2
Class A						
Traffic	0.2	4.0	4.5	1.44	2.77	3.19
PeopleOnStreet	0.1	5.9	7.5	1.10	1.96	2.37
Class B						
Kimono	0.1	6.6	7.3	1.24	2.34	2.19
ParkScene	0.1	4.7	3.9	1.31	2.47	2.28
Cactus	0.2	6.9	3.9	1.32	2.46	2.23
BasketballDrive	0.1	7.6	7.7	1.20	2.28	2.11
BQTerrace	0.3	5.9	4.9	1.39	2.53	2.36
Class C						
BasketballDrill	0.2	2.4	4.5	1.24	1.85	2.19
BQMall	0.3	2.5	3.5	1.26	1.88	2.20
PartyScene	0.5	1.9	3.6	1.22	1.83	2.17
RaceHorsesC	0.1	3.3	4.6	1.06	1.62	1.92
Class D						
BasketballPass	0.3	2.2	3.4	1.14	1.70	1.94
BQSquare	0.5	1.9	2.1	1.25	2.06	2.41
BlowingBubbles	0.7	2.2	2.9	1.30	2.03	2.28
RaceHorses	0.1	1.5	3.1	1.09	1.62	1.85
Average	0.3	4.0	4.5	1.23	2.09	2.24

Table 4 Speedup and coding efficiency results of the GPU-based proposal

	BD-rate (%)	Speedup
Class A		
Traffic	0.3	1.06
PeopleOnStreet	−0.5	1.16
Class B		
Kimono	0.5	1.14
ParkScene	0.0	1.07
Cactus	0.2	1.11
BasketballDrive	3.5	1.19
BQTerrace	−1.0	1.08
Class C		
BasketballDrill	−0.5	1.14
BQMall	0.8	1.11
PartyScene	−0.3	1.09
RaceHorsesC	0.0	1.20
Class D		
BasketballPass	0.0	1.15
BQSquare	−0.1	1.06
BlowingBubbles	−0.4	1.07
RaceHorses	−0.5	1.16
Average	0.1	1.12

On the other hand, if using WPP as part of the low complexity transcoder architecture, it can be seen that a speedup of 5.69× can be achieved on the same processor (which means more than a 82 % of encoding time reduction), while the BD-rate has a negligible impact of 7.2 % on average. The use of GIPT should be chosen if the sequence to be transcoded is large enough while its GOP size is not too large. In other case, WPP should be chosen, specially when the frame resolution is large enough to decrease the impact of the "ramping inefficiencies". In this case, the worst speedup is 3.70×, which means a time reduction of almost three quarters, which is still a very good value.

Moreover, this speedup could be even higher if the transcoder is run on a multi-core processor able to execute more parallel threads. Specifically, it could scale proportionally to the frame height when using WPP and proportionally to the sequence length when using GIPT (which would result in a very high parallel capability).

Furthermore, it can be seen that a combination of all the aforementioned effects of applying the different proposals can be appreciated in this table. For example, FQLD applied to less levels results in lower BD-rates and speedups and vice versa, the CPU parallelism decreases when the frame resolution also decreases when using WPP or the speedups depends on the scene content due to the impact of the GPU proposed algorithm. Each of these proposals could also be switched on or off by the user according to the requirements (this is the reason why the results have also been presented separately, so that the effect of each part of the transcoder architecture could be seen if the user wished to switch off any of the parts).

Table 5 Results for the complete proposed transcoder architecture

	BD-rate (%)		Speedup/Time reduction (%)	
	GIPT+NB+GPU	WPP+NB+GPU	GIPT+NB+GPU	WPP+NB+GPU
Class A				
Traffic	5.2	6.2	7.04/85.80	6.61/84.87
PeopleOnStreet	6.7	7.9	8.44/88.15	7.62/86.88
Class B				
Kimono	11.4	8.5	7.83/87.23	6.98/85.67
ParkScene	7.1	6.0	7.50/86.67	6.53/84.69
Cactus	9.3	10.5	7.48/86.63	6.47/84.54
BasketballDrive	14.2	14.8	8.38/88.07	7.27/86.24
BQTerrace	7.4	8.3	7.34/86.38	6.31/84.15
Class C				
BasketballDrill	6.1	8.4	7.08/85.88	5.52/81.88
BQMall	6.8	6.8	6.74/85.16	5.69/82.43
PartyScene	6.0	6.2	6.73/85.14	5.30/81.13
RaceHorsesC	6.2	5.8	7.27/86.24	5.91/83.08
Class D				
BasketballPass	6.0	5.6	6.39/84.35	3.70/72.97
BQSquare	5.9	3.8	5.86/82.94	3.92/74.49
BlowingBubbles	6.7	5.3	5.69/82.43	3.75/73.33
RaceHorses	5.3	3.9	6.69/85.05	3.79/73.61
Average	7.4	7.2	7.10/85.92	5.69/82.43

Table 6 Percentage of use of the GPU in the different scenarios

	GPU only (1 Thread)		GIPT+NB+GPU (4 Threads)		WPP+NB+GPU (4 Threads)	
	QP 22	QP 37	QP 22	QP 37	QP 22	QP 37
PeopleOnStreet (class A)	7.1	11.1	51.8	84.7	47.0	75.4
Kimono (class B)	8.8	12.9	61.3	90.3	57.8	77.1
RaceHorses (class C)	6.5	11.7	39.2	71.6	31.1	59.0
BasketballPass (class D)	9.3	14.6	50.4	76.6	26.7	52.3
Average	7.9	12.6	50.7	80.8	40.7	66.0

Table 6 shows the percentage of utilization of the GPU for the GPU-based proposal (using only 1 thread) for some sequences and QP values and for the full transcoding architecture using either GIPT or WPP, in which case the simulations were performed using 4 threads. As can be seen, with only one encoding thread, the device is idle a great percentage of the time. Therefore, it would be possible to perform more operations on the device and, in that way, increase the overall performance of the encoder. However, special attention should be given to the results when using multiple threads and, more specifically, with high QP values. In this case, the percentage of utilization grows up to 90.3 %, so assigning new tasks to the GPU would result in a lower performance due to the hardware limits of the device.

Finally, to demonstrate that the transcoding process produces video streams in HEVC format that are, at least, as good as the original H.264/AVC streams, Table 7 shows the average bitrate difference between the H.264/AVC and the HEVC streams for $QP = \{22, 27, 32, 27\}$ ($\overline{\Delta Bitrate}$). This table also shows the average *Peak Signal-to-Noise Ratio* (PSNR) difference between the decoded streams when comparing them with the original YUV sequence ($\overline{\Delta PSNR}$). All the results are shown for the transcoder architecture using either GIPT or WPP.

It can be seen that in the transcoding process 1.17 dB are lost in average in the PSNR due to the fact that the reference to calculate the PSNR is the original YUV sequence and, after transcoding, the video has been quantified twice: first it was done by the H.264/AVC

Table 7 Comparison between the H.264/AVC and the transcoded HEVC streams

	GIPT+NB+GPU		WPP+NB+GPU	
	$\overline{\Delta Bitrate}$ (%)	$\overline{\Delta PSNR}$ (dB)	$\overline{\Delta Bitrate}$ (%)	$\overline{\Delta PSNR}$ (dB)
Class A				
Traffic	−27.66	−1.34	−27.45	−1.33
PeopleOnStreet	−19.30	−1.36	−19.10	−1.35
Class B				
Kimono	−18.75	−0.87	−18.39	−0.85
ParkScene	−13.00	−1.05	−12.65	−1.06
Cactus	−25.33	−0.78	−24.44	−0.79
BasketballDrive	−25.06	−0.67	−24.36	−0.66
BQTerrace	−32.03	−0.51	−31.41	−0.50
Class C				
BasketballDrill	−29.98	−1.48	−29.07	−1.50
BQMall	−26.34	−0.89	−25.44	−0.87
PartyScene	−12.91	−1.35	−12.42	−1.37
RaceHorsesC	−12.58	−1.53	−11.91	−1.51
Class D				
BasketballPass	−21.09	−1.42	−20.60	−1.42
BQSquare	−25.00	−0.98	−24.59	−0.94
BlowingBubbles	−24.03	−1.10	−23.39	−1.09
RaceHorses	−15.84	−1.50	−15.15	−1.47
Average	−22.06	−1.17	−21.54	−1.16

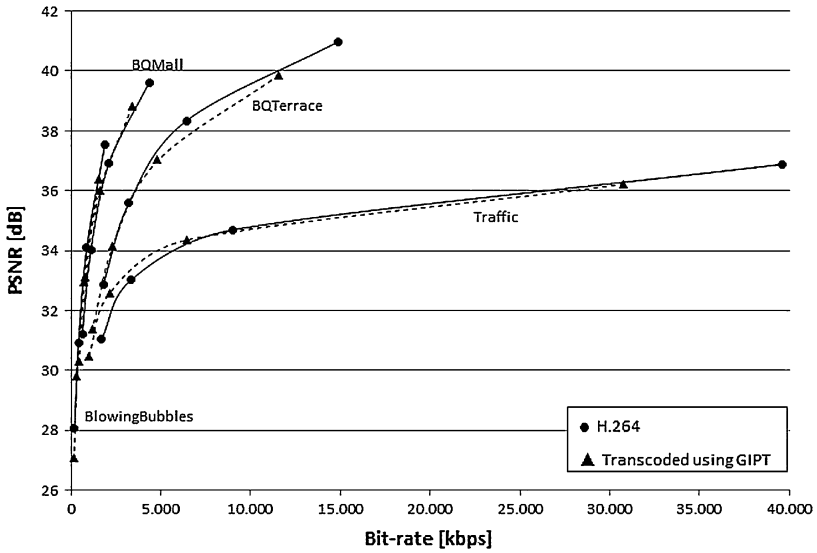
encoder and now by the HEVC encoder. However, a bitrate saving of around a 22 % is obtained, i.e. the quality is slightly lower, as expected, but the bitrate saving offsets this loss.

Moreover, Fig. 8 shows the Rate-Distortion curves corresponding to some of the sequences in Table 7. In this figure, the combined effect of bitrate and PSNR can be seen, making it evident that the H.264/AVC and HEVC curves are pretty close from each other (which means that the quality is similar), while the HEVC curves are more to the left than the the H.264/AVC ones, making it also evident that the bitrate is lower.

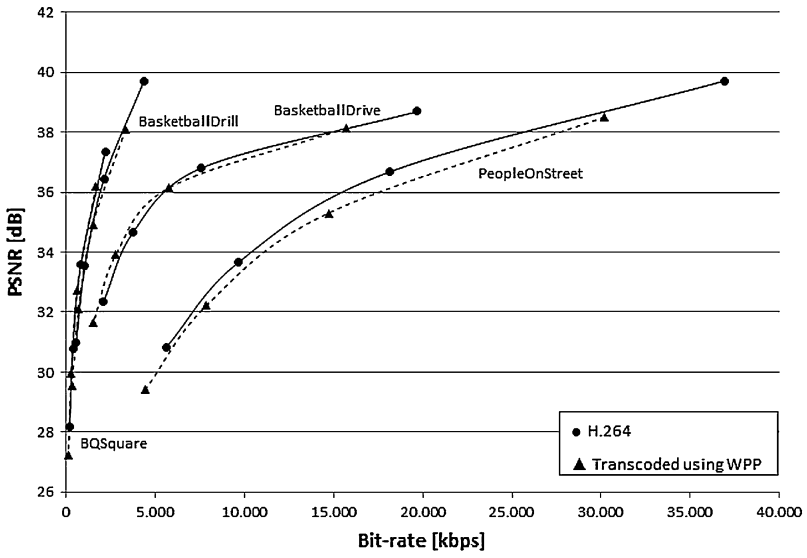
6 Conclusions

This paper contains the proposal of an architecture of an accelerated H.264/AVC to HEVC transcoder which is executed on the HEVC encoder side as part of the transcoder by taking into account what has occurred in the H.264/AVC decoder. First, a coarse-grained parallelization of the whole encoding process using a multi-core CPU is performed using known techniques such as GIPT and WPP. After that, a classifying algorithm which decides on the most appropriate quadtree level without the need for testing all the possible CUs/PUs is proposed and, finally, a parallel Integer ME algorithm is proposed to be executed on a GPU which servers as a co-processor to the main multi-core CPU. Thus, the proposed video transcoder architecture combines software and hardware techniques for an H.264/AVC to HEVC in a heterogeneous platform.

Results show that a good tradeoff between quality loss and acceleration is achieved: in the case of GIPT, a 7.10× of speedup (or more than 85 % in terms of encoding time reduction) on average on a quad-core Intel i7 processor with a negligible increment of 7.4 % in the BD-rate. In the case of WPP a 5.69× speedup on average (more than a 82 % in terms of encoding time reduction) with a 7.2 % of increment in the BD-rate is achieved. The use of GIPT or WPP as part of the low-complexity transcoder architecture should be decided by the user according to sequence characteristics. The user could also choose the number of levels (0, 1 or 2) where the FQLD algorithm will be applied according to the complexity-quality requirements. Moreover, full parts of the proposed architecture (CPU parallelism, GPU or FQLD) could also be switched on or off by the user depending on, for example, the kind of physical platform where it will be run.



(a) Transcoded using GIPT+NB+GPU.



(b) Transcoded using WPP+NB+GPU.

Fig. 8 RD curves of the H.264 and the transcoded HEVC streams using GIPT (**a**) or WPP (**b**), the FQLD algorithm and the GPU

Furthermore, the proposed architecture is scalable to CPUs which can execute more parallel threads (specially if using GIPT). In this situation, the GPU could be a bottleneck but this could also be avoided by adding more GPUs. Moreover, an independent classifier could also be run on each core. This situation makes the proposed architecture a fully scalable transcoder, in which the more parallel threads the processor is able to execute, the faster the transcoder will be.

References

1. Bjontegaard, G.: Improvements of the BD- PSNR model. ITU-T SG16 Q 6, 35 (2008)
2. Bossen, F.: Common HM test conditions and software reference configurations. In: Proceedings of 12th JCT-VC Meeting, Doc. JCTVC-L1100 (2013)
3. Bossen, F., Bross, B., Suhring, K., Flynn, D.: HEVC complexity and implementation analysis. IEEE Trans. Circuits Syst. Video Technol. **22**(12), 1685–1696 (2012). doi:10.1109/TCSVT.2012.2221255

4. Bross, B., Han, W., Ohm, J., Sullivan, G., Wang, Y.K., Wiegand, T.: High efficiency video coding (HEVC) text specification draft 10. Doc. JCTVC-L1003 (2013)
5. Cheung, N.M., Fan, X., Au, O., Kung, M.C.: Video coding on multicore graphics processors. Sig. Process. Mag. IEEE **27**(2), 79–89 (2010). doi:10.1109/MSP.2009.935416
6. Chi, C.C., Alvarez-Mesa, M., Juurlink, B., Clare, G., Henry, F., Pateux, S., Schierl, T.: Parallel scalability and efficiency of hevc parallelization approaches. Circuits Syst. Video Technol. IEEE Trans. **22**(12), 1827–1838 (2012)
7. Corrales-Garcia, A., Martinez, J.L., Fernandez-Escribano, G., Quiles, F.J.: Variable and constant bitrate in a DVC to H.264/avc transcoder. Sig. Process. Image Commun. **26**(6), 310–323 (2011)
8. Fayyad, U.M., Irani, K.B.: Multi-Interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the International Joint Conference on Uncertainty in AI, pp. 1022–1027 (1993)
9. Feng, C.W., Manocha, D.: High-performance computing using accelerators. Parallel computing 33(10–11), 645–647 (2007). http://dblp.uni-trier.de/db/journals/pc/pc33.html#FengM07
10. Fernandez-Escribano, G., Bialkowski, J., Gamez, J., Kalva, H., Cuenca, P., Orozco-Barbosa, L., Kaup, A.: Low-complexity heterogeneous video transcoding using data mining. IEEE Trans. Multimed. **10**(2), 286–299 (2008)
11. Fernandez-Escribano, G., Kalva, H., Cuenca, P., Orozco-Barbosa, L., Garrido, A.: A fast mb mode decision algorithm for MPEG-2 to H.264 p-frame transcoding. IEEE Trans. Circuits Syst. Video Technol. **18**(2), 172–185 (2008)
12. Flores, J., Gámez, J.A., Martínez, A.M.: Supervised classification with Bayesian networks. In: Intelligent data analysis for real-life applications: theory and practice, pp. 72–102 (2012)
13. Garrido-Cantos, R., De Cock, J., Martinez, J., Vanleuven, S., Cuenca, P.: Motion-based temporal transcoding from H.264/ AVC-to- SVC in baseline profile. Consum. Electr. IEEE Trans. **57**(1), 239–246 (2011)
14. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
15. Henry, F., Pateux, S.: Wavefront parallel processing. Tech. Rep. JCTVC-E196 (2011)
16. HM reference software. http://hevc.hhi.fraunhofer.desvnsvn.HEVCSoftware
17. ITU-T, JTC, I.: Advanced video coding for generic audiovisual services. ITU-T Rec. H.264 and ISO/IEC 14496–10 (AVC) version 16 (2012)
18. Joint collaborative team on video coding: reference software to Committee Draft, version 18.4 (2012)
19. Lopez-Granado, O.M., Malumbres, M.P., Migallón, H., Piñol, P.J.: Analyzing GOP-based parallel strategies with the HEVC encoder. In: Proceedings of the 13th International Conference Computational and Mathematical Methods in Science and Engineering (2013)
20. Misra, K., Segall, A., Horowitz, M., Xu, S., Fuldseth, A., Zhou, M.: An overview of tiles in hevc. IEEE J. Sel. Topics Sig. Process. **7**(6), 969–977 (2013). doi:10.1109/JSTSP.2013.2271451
21. NVIDIA: NVIDIA CUDA compute unified device architecture programming guide, version 3.2 (2010)
22. Ohm, J., Sullivan, G., Schwarz, H., Tan, T.K., Wiegand, T.: Comparison of the coding efficiency of video coding standards: including high efficiency video coding (HEVC). IEEE Trans. Circuits Syst. Video Technol. **22**(12), 1669–1684 (2012)
23. Patnaik, S., Yang, Y.M.: Soft computing techniques in vision science, vol. 395. Springer (2012)
24. Peixoto, E., Izquierdo, E.: A complexity-scalable transcoder form H.264/ AVC to the new HEVC codec. In: International Conference on Image Processing (ICIP), Orlando, FL, USA (September 2012) (2012)
25. Peixoto, E., Shanableh, T., Izquierdo, E.: H.264/ AVC to HEVC video transcoder based on dynamic thresholding and content modeling. IEEE Trans. Circuits Syst. Video Technol **24**(1), 99–112 (2014)
26. Shen, T., Lu, Y., Wen, Z., Zou, L., Chen, Y., Wen, J.: Ultra fast H.264/ AVC to HEVC transcoder. In: Data Compression Conference (DCC), 2013, pp. 241–250 (2013)
27. Su, H., Wu, N., Zhang, C., Wen, M., Ren, J.: A multilevel parallel intra coding for H.264/AVC based on CUDA. In: Image and Graphics (ICIG), 2011 Sixth International Conference on, pp. 76–81 (2011) doi:10.1109/ICIG.2011.99
28. Sullivan, G.J., Ohm, J.R., Han, W.J., Wiegand, T.: Overview of the high efficiency video coding (HEVC) standard. IEEE Trans. Circuits Syst. Video Technol **22**(12), 1649–1668 (2012)
29. Vetro, A., Christopoulos, C., Sun, H.: Video transcoding architectures and techniques: an overview. IEEE Sig. Process. Mag. **20**(2), 18–29 (2003)
30. Wang, X., Song, L., Chen, M., Yang, J.: Paralleling variable block size motion estimation of HEVC on CPU plus GPU platform. IEEE Int. Conf. Multimed. Expo Workshops (ICMEW) **2013**, 1–5 (2013). doi:10.1109/ICMEW.2013.6618412
31. Yan, C., Dai, F., Zhang, Y.: Parallel deblocking filter for H.264/ AVC on the TILERA many-core systems. In: Advances in Multimedia Modeling, Lecture Notes in Computer Science, vol. 6523, pp. 51–61. Springer, Berlin (2011). doi:10.1007/978-3-642-17832-0_6
32. Yan, C., Zhang, Y., Dai, F., Li, L.: Highly parallel framework for HEVC motion estimation on many-core platform. Data Compress. Conf. (DCC) **2013**, 63–72 (2013). doi:10.1109/DCC.2013.14
33. Yu, Q., Zhao, L., Ma, S.: Parallel AMVP candidate list construction for HEVC. In: Visual communications and image processing (VCIP), 2012 IEEE, pp. 1–6 (2012). doi:10.1109/VCIP.2012.6410775
34. Zhang, D., Li, B., Xu, J., Li, H.: Fast transcoding from H.264 avc to high efficiency video coding. In: IEEE International Conference on Multimedia and Expo (ICME) 2012, pp. 651–656 (2012)

Antonio Jesús Díaz-Honrubia received his B.Sc. degree in Computer Science and Engineering in 2011 and the M.Sc. in Advanced Computing Technologies and the M.Sc. in Computer Science and Engineering in 2012 all of them from the University of Castilla-La Mancha. Nowadays he is studying a Ph.D. in Advanced Computing Technologies at the same University, where he carries out his researching activities in the Computer Architecture and Technology group at the Albacete research institute of informatics (I3A) since 2011. His main field of research is the video transcoding from H.264/AVC to High Efficiency Video Coding standard. He is also interested in 3D video coding.

Gabriel Cebrián-Márquez received the B.Sc. and the M.Sc. degrees in Computer Science and Engineering from the University of Castilla-La Mancha (Spain) in 2013 and 2014, respectively. He is currently studying a Ph.D. in Advanced Computing Technologies at the same university. In 2013, he joined the Laboratory of High-Performance Networks and Architectures (RAAP) as a research assistant at the Albacete Research Institute of Informatics (I3A). His current research interests and areas of publication include video processing and coding, parallel and heterogeneous architectures and High-Performance Computing (HPC).

José L. Martínez received his M.Sc. and Ph.D. degrees in Computer Science and Engineering from the University of Castilla-La Mancha, Albacete, Spain in 2007 and 2009 respectively. In 2005, he joined the Department of Computer Engineering at the University of Castilla-La Mancha, where he was a researcher of Computer Architecture and

Technology group at the Albacete research institute of informatics (I3A). After completing his Ph.D, he was postdoc researcher at Centre for Communication System Research (CCSR), at the University of Surrey, Guildford (UK). In 2010, he joined the department of Computer Architecture of Complutense University in Madrid (Spain) where he was assistant professor. In 2011, he came back to University of Castilla-La Mancha where he is currently assistant professor. His research interests include Distributed Video Coding (DVC), multimedia standards, video transcoding, scalable video coding, video applications for multicore and GPUs architectures, video quality metrics. He has also been a visiting researcher at the Florida Atlantic University, Boca Raton (USA) for 9 months. He has 23 publications in these areas in international refereed journals and 50 conference proceedings.

Pedro Cuenca received his M.Sc. degree in Physics (award extraordinary) from the University of Valencia in 1994. He got his Ph.D. degree in Computer Engineering in 1999 from the Polytechnic University of Valencia. In 1995 he joined the Department de Computer Engineering at the University of Castilla-La Mancha. He is currently a Full Professor of Communications and Computer Networks and Dean of the Escuela Superior de Ingeniería Informática in Albacete. His research topics are centered in the area of wireless LAN, video compression, QoS video transmission and error-resilient protocol architectures. He has published over 100 papers in international Journals and Conferences. He has served in the organization of International Conferences as Chair, Technical Program Chair and Technical Program Committee member. He was the Chair of the IFIP 6.8 Working Group.

José Miguel Puerta received his M.Sc. and Ph.D. degrees in Computer Science in 1991 and 2001, respectively, from the University of Granada. He joined the Department of Computer Science at the University of Castilla-La Mancha in 1991, where he is currently an Associate Proffesor and Vice-Dean of the Escuela Superior de Ingeniería Informática in Albacete. His research activity is developed at the Laboratory of Intelligent Systems and Data Mining since its founding. His main research interest include machine learning, specifically in probabilistic graphical models and Bayesian Networks, metaheuristics and evolutionary algorithms, specifically estimation of distribution algorithms. He has edited several books and a special issue of journal, as well as he is author of several papers in journals specialized in his research topics. He has served as Co-Chair in the 1st European Workshop on Probabilistic Graphical Models and in some special sessions about metaheuristics and data mining.

José A. Gámez received the M.Sc. degree in Computer Science in 1991, and the Ph.D. degree in Computer Science in 1998, both from the University of Granada, Spain. He joined the Department of Computer Science at the University of Castilla-La Mancha (UCLM) in 1991, where he is currently a Full Professor. He has served as Vice-Dean of the Escuela Politecnica Superior de Albacete (UCLM) from 1998 to 2004 and as Chair of the Department of Computing Systems (UCLM) from 2006 to 2012. His research activity is developed at the Laboratory of Intelligent Systems and Data Mining (SIMD) as the co-director of this lab. His main research interest include probabilistic reasoning, Bayesian networks, evolutionary algorithms, machine learning and data mining. In these topics Dr. Gamez has edited five books and five special issues of international journals. He is the (co)author of more than one hundred papers in journals, books and refereed international conferences. He has served as Co-Chair in the 1st European Workshop on Probabilistic Graphical Models, as Vice-Chair in the XIV Spanish Conference of Artificial Intelligence and as Chair in the VIII Spanish Conference on Metaheuristics and Evolutionary Algorithms.

CHAPTER 4

Inter and Intra Pre-Analysis Algorithm for HEVC

- **Title**: Inter and Intra Pre-Analysis Algorithm for HEVC.
- **Authors**: Gabriel Cebrián-Márquez, José Luis Martínez, and Pedro Cuenca.
- **Type**: journal paper.
- **Journal**: Journal of Supercomputing.
- **Publisher**: Springer US.
- **ISSN**: 0920-8542.
- **Status**: published.
- **Publication date**: January 2017.
- **Volume**: 73.
- **Issue**: 1.
- **Pages**: 414–432.
- **DOI**: 10.1007/s11227-016-1882-9
- **JCR IF/ranking**: 1.326/Q2 (JCR2016).

J Supercomput (2017) 73:414–432
DOI 10.1007/s11227-016-1882-9

Inter and intra pre-analysis algorithm for HEVC

Gabriel Cebrián-Márquez[1] · **José Luis Martínez**[1] · **Pedro Cuenca**[1]

Published online: 3 October 2016

Abstract The unceasing demands for high quality multimedia contents and the advent of new resolutions such as ultra high-definition motivated the development of the high efficiency video coding (HEVC) standard, which outperforms prior standards by up to 50 % in terms of coding efficiency. While this improvement meets the aforementioned demands, it also involves a considerable increase in computational cost of the encoder. For this reason, fast and efficient coding algorithms are now a requirement for HEVC-compliant real-time encoders. In this regard, this paper proposes a pre-analysis algorithm designed to provide coding information to both the intra and the inter modules of the encoding stage. As a result, experiments show that the algorithm is able to reduce the encoding time between 12.77 and 18.32 % for random access configurations, at the expense of negligible losses in terms of coding efficiency.

Keywords HEVC · Pre-analysis · Intra prediction · Inter prediction · Reference frame selection · Motion estimation

✉ Gabriel Cebrián-Márquez
gabriel.cebrian@uclm.es

José Luis Martínez
joseluis.martinez@uclm.es

Pedro Cuenca
pedro.cuenca@uclm.es

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

1 Introduction

For more than 10 years, H.264/advanced video coding (AVC) [1] has established itself as the most widespread video compression standard for all types of applications and scenarios, e.g. Blu-ray and also various high-definition (HD) television broadcasts. However, the advent of new video formats such as the ultra high-definition (UHD) resolution and the ensuing increase in bit rate along with the demands for higher video quality motivated the development of the high efficiency video coding (HEVC) standard [2]. Defined by the joint collaborative team on video coding (JCT-VC) in early 2013, HEVC roughly doubles the rate-distortion (R–D) performance of H.264/AVC, which implies nearly 50 % bit rate reduction for the same video quality [3]. This improvement in coding efficiency comes, however, at the expense of extremely high computational requirements [4].

To reduce the total processing time, HEVC introduces some high-level tools, such as tiles and wavefronts, with the aim of allowing the parallel encoding and decoding of a video sequence. These tools are, however, insufficient to achieve real-time video compression. For this reason, the research community is making a considerable effort to find complementary ways of reducing the processing time of the encoder not only from the point of view of parallelism, but also algorithmically.

Figure 1 shows the profiling results obtained from the baseline HEVC encoder for the *BasketballDrive* 1080 p sequence, setting the quantization parameter (QP) to 32 and using the widespread random access configuration, which makes use of both temporal (inter) and spatial (intra) prediction. As can be seen, the vast majority of the time is devoted to the inter prediction module of the encoder, becoming the most computationally expensive operation for this configuration, whereas only 33.69 % of the total execution time corresponds to the intra prediction and the remaining modules (labelled as "Others"), including transform, quantization, entropy coding, and in-loop filtering. The computational requirements of the inter prediction module can be justified by the motion estimation (ME) algorithm, which conducts large amount of repetitive operations that the encoder has to perform on the same picture samples but with different block partitions, i.e. different prediction units (PU) for the same coding tree unit (CTU). While this distribution of time applies to most configurations, the All Intra configuration dispenses with the temporal prediction and opts for the spatial

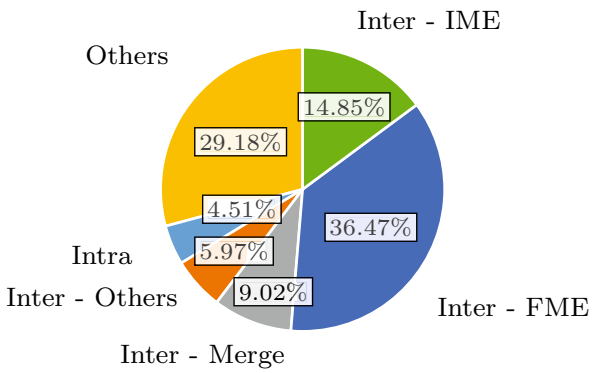


Fig. 1 Time profile of the *BasketballDrive* sequence (1920 × 1080) using random access configuration and $QP = 32$

prediction instead. As a result, the intra prediction module can exceed 90 % of the total encoding time, given the considerable number of modes it has to test for every PU, namely 33 directional modes, and the DC and planar modes. For this reason, it has to be noted that both prediction modules are equally important depending on the coding configuration used.

A remarkable number of multimedia applications require real-time encoders, such as broadcasting. For this reason, these encoders typically implement fast algorithms and techniques that support the encoding in several ways. In particular, many commercial encoders include the so-called pre-analysis algorithm, which is responsible for providing the encoder some preliminary information about the input pictures and the encoding process. This information is used by the encoder for different purposes, e.g. adjusting the rate-control parameters or deciding the encoding pattern in random access configurations. In this paper, we design and implement a pre-analysis algorithm for HEVC with the aim of reducing the encoding time of both the inter and intra coding modules. In this way, the aforementioned algorithm performs a fast calculation of the costs of the different 35 intra modes, which is later used in the intra module of the encoder, and also carries out a fast motion estimation to reduce the processing time of the inter prediction module. As a result, the encoding time is reduced having little impact on the coding efficiency. It also has to be noted that this algorithm can be applied to several other video codecs with complex partitioning schemes as in HEVC. This work constitutes an extension of the intra pre-analysis algorithm proposed in [5], and also the two-stage ME algorithm proposed in [6,7].

The rest of this paper is organized as follows. Section 2 presents briefly the principles of HEVC. Section 3 covers some of the most relevant related works in the topic. Following this, Sect. 4 provides an overview of the pre-analysis architecture and the proposed algorithm. An experimental evaluation of this algorithm is carried out in Sect. 5, showing its results in terms of time reduction and coding efficiency. Finally, in Sect. 6 conclusions are drawn.

2 Technical background

The HEVC standard can be seen as an evolution of the current H.264/AVC standard, since it maintains the same block-based approach used in previous coding standards. However, not only did HEVC introduce new coding tools in the standard, but also it improved others which were already present, notably increasing the overall compression efficiency. In fact, one of the most important changes affects the picture partitioning. HEVC discards the term macroblock (MB) and replaces it with a new structure termed CTU. CTUs are squared regions into which the picture is divided. Each CTU, whose size is typically 64 × 64, can be recursively partitioned into four coding units (CU). These CUs, ranging from 8 × 8 to 64 × 64 pixels, are composed of PUs and transform units (TU). CTUs and CUs form a quadtree structure as the one shown in Fig. 2.

With regard to the partitioning of CUs into PUs, HEVC defines up to eight possible inter and intra partitions for each CU size (2N × 2N, 2N × N, N × 2N, N × N, 2N×nU, 2N×nD, nL×2N and nR×2N) to determine the optimal trade-off between rate and

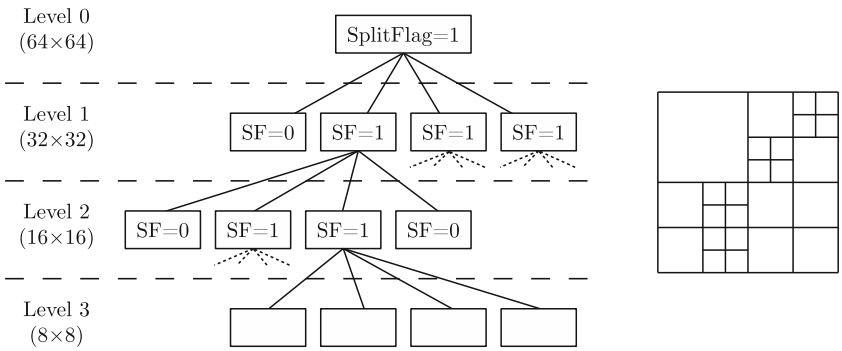


Fig. 2 Quadtree partitioning structure in HEVC

distortion. The last four PU types correspond to the asymmetric motion partitioning (AMP), and they were not present in previous standards. The introduction of these new sizes involves an increase in the computational requirements of the encoder. However, it also allows the standard to be more flexible to adapt itself for edges, specially with larger CU sizes. For intra prediction, a PU inherits the same 2N × 2N size of the CU to which it belongs, except when the CU cannot be further split, in which case the PU can be further divided into four smaller N × N regions. Concerning inter prediction, the standard allows both symmetrical and asymmetrical PU partitions for all CU sizes, but restricts the usage of asymmetric partitions to CUs whose size is larger than 8 × 8 pixels.

As a result of performing temporal or spatial prediction, a residual is generated. This residual is transformed using various TU sizes ranging from 32 × 32 to 4 × 4, forming what is known as the residual quadtree (RQT), with its root at the CU level. This allows the encoding process to differentiate between prediction and transform, as the TU partitioning is not required to match the PU partitioning. Other new features in HEVC include a total of 35 different intra coding modes, the definition of Tiles [8] and Wavefront Parallel Processing (WPP) [9] as parallel tools, and new in-loop filters.

3 Related work

Even though the standard already defines parallel tools that assist the encoding process, several related works focus on further exploiting the parallelism of HEVC. For example, authors in [10] describe the overlapped wavefront (OWF) algorithm, an improved version of WPP that solves the problem of the so-called ramping inefficiencies at the beginning and at the end of the frames. Some others make use of techniques already present in previous standards, e.g. slice-based parallelism [11], which results in high bit rate penalties, or through independent group of pictures [12], which introduce some latency to the encoder. Alternatively, an analysis of the existing dependencies in the ME module is shown in [13], where a highly parallel framework designed for many-core processors is proposed. Many other works make use of heterogeneous platforms,

such as those based on GPU [14,15]. However, as mentioned in the introduction of this paper, these techniques are not enough to reduce the total encoding time to practical levels, and fast encoding algorithms are still necessary.

Other related works focus on reducing the processing time of the encoder from an algorithmic point of view. The approach followed by many is to prune the decision tree to skip those partitions in which it is unlikely to find a better prediction candidate than the current one. For example, in the case of intra coding, authors in [16] make use of Bayesian techniques to determine the splitting or pruning of the CTU tree based on a set of statistical parameters, while authors in [17] show that the splitting of the current CU can be determined by the existing relationship in terms of mode and cost with the neighbouring CUs. All of these techniques perform their operations on a higher level, and thus they are complementary to the algorithm proposed in this paper. Some other works focus on designing faster intra mode selection algorithms such as [18,19]. However, while the authors obtain a considerable speed-up, the resulting BD-rate is higher than the one obtained in this paper.

With regard to the inter prediction, a considerable number of works present alternative ME algorithms, e.g. authors in [20] make use of spatiotemporal variables to reduce the number of partitions tested. A survey of algorithms combining both alternatives can be found in [21], in which time reduction results of more than 50 % can be found at the expense of small coding efficiency penalties. However, similar to the previous intra-related works, these alternative algorithms make their decisions at a higher level and still have to carry out the ME algorithm, so higher speed-ups could be expected if they are combined with the algorithm proposed in this paper.

With regard to pre-analysis algorithms, it is worth mentioning the look-ahead module of x264 H.264/AVC encoder [22,23]. This module splits a subsampled version of the input frame into fixed-size blocks, and calculates their corresponding intra and inter costs. Some of the ways in which this information is used in the encoder include determining the encoding pattern, speeding up some operations and adjusting the parameters of the rate-control module. However, this technique is targeted for H.264/AVC, which involves less block partitions than HEVC, and even though it has been assimilated in x265 [24] (the version of x264 for HEVC), it does not completely meet the requirements of the new standard. Other uses for the pre-analysis algorithm range from the adjustment of the rate-control algorithm [25] to the analysis of the texture of a picture for CU-size selection in intra configurations [26], and are completely complementary to the algorithm proposed in this paper.

4 Proposed pre-analysis algorithm

As introduced in the first section of this paper, a pre-analysis algorithm is responsible for obtaining some preliminary information from the input frames, which is in turn used by the encoder in various ways. Figure 3 shows the architecture of a typical HEVC-compliant encoder, in which a pre-analysis stage has been added. The decoder processing blocks that the encoder duplicates have been shaded in grey. As can be seen, the input pictures are analysed by this stage before any encoding is performed. This involves that any information has to be gathered from the original samples, i.e. without

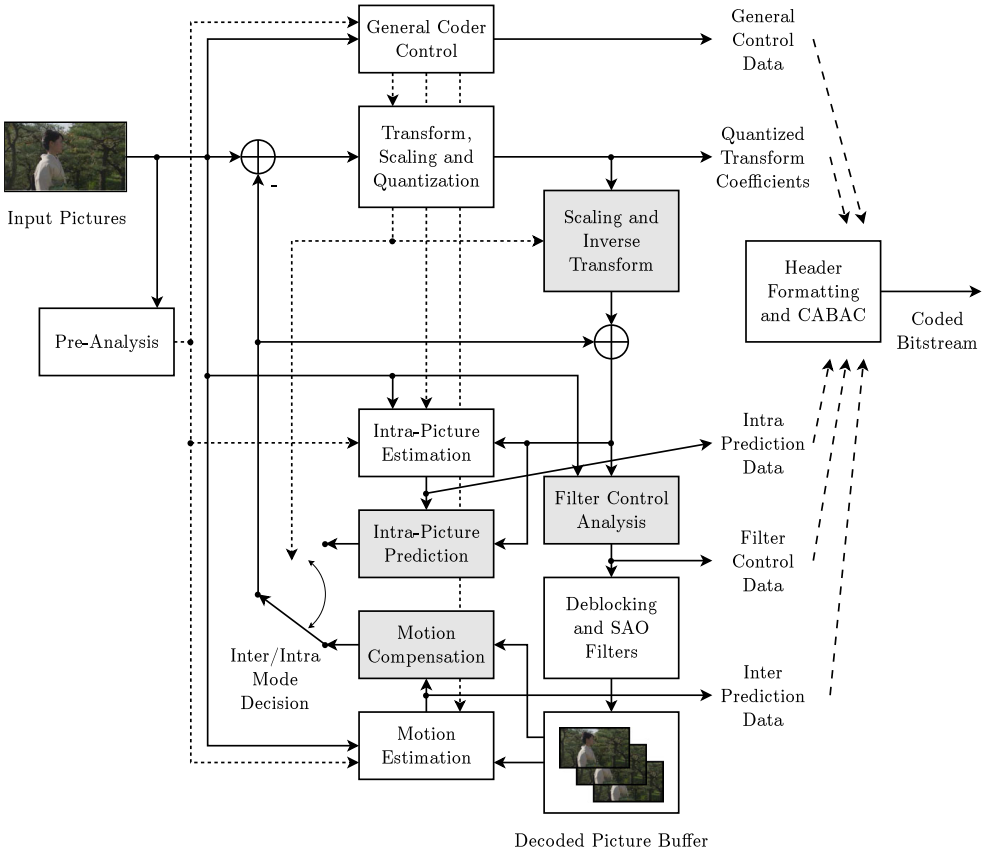


Fig. 3 Typical HEVC encoder architecture with a pre-analysis stage (shared elements with the decoder are *shaded in grey*)

any partitioning or neighbour predictors. In spite of this, the amount of information that can be extracted and predicted by the pre-analysis algorithm is endless, including motion information, intra modes, prediction costs, coding pattern, partitioning, etc. This information is thus used in the many modules that constitute the encoder to control or to aid the encoding process.

In prior works, we focused our efforts on implementing a pre-analysis algorithm for inter prediction. The obtained motion information can be used to reduce the number of positions checked in the search area [6,7] or to limit the number of reference frames [7]. As a result, the total encoding time is notably reduced at the expense of negligible losses in terms of coding efficiency. Nevertheless, this pre-analysis algorithm is only valid for those coding configurations that make use of inter prediction, such as Random Access. For this reason, in [5] we proposed an alternative but complementary algorithm for the intra prediction module, especially targeted to All Intra configurations. This section will show how both techniques can be joined into a single pre-analysis algorithm in a very flexible way. The former can provide good speed-up results for configurations that make use of temporal prediction, while the latter can contribute to all kinds of configurations indistinctly.

4.1 Pre-analysis algorithm for intra prediction

The aim of the pre-analysis stage is to reduce the total encoding time. To achieve this reduction, one first step would be simplifying the intra module of the encoder, which can constitute more than 90 % of the encoding time for all intra configurations, as detailed in Sect. 1. In this regard, the pre-analysis stage has been designed to estimate the best intra mode and its corresponding cost for each PU in a fast way. This is achieved by carrying out a similar process to that of the intra module. The latter, however, involves large computation times, as it needs to test every possible mode and perform the transform operation. For this reason, this operation is simplified in the pre-analysis stage using less complex operations.

One way in which the intra operation can be performed in the pre-analysis stage is by dividing the input frames into fixed-size block partitions, resulting in a grid of square blocks. For each of these blocks, the best intra mode is calculated (directional, DC or planar), which in turn provides a prediction cost. However, we proved experimentally that extrapolating the best intra mode from the pre-analysis data for every PU is not really possible if only one block size is utilised, given the large range of partitioning possibilities that HEVC offers. For this reason, our proposal divides the input frames into fixed-size block partitions that vary from 4×4 to 64×64, including 8×8, 16×16 and 32×32. In this way, this preliminary information can be more easily adapted to the different PUs the encoder might test. In fact, for each PU there is a pre-analysis block that exactly matches in position and size.

To obtain the most reliable prediction possible, all the 33 directional modes are checked for every block, along with the DC and planar modes. With the aim of performing a fast estimation, the sum of absolute differences (SAD) measurement is used instead of the Hadamard transform. While the latter offers a better representation of the distortion caused by the prediction, the former also allows to compare several prediction modes, but involving considerably less computation.

It also has to be taken into account that a pre-analysis stage has some limitations compared to the encoding stage itself. Nevertheless, it is possible to address them so that the results obtained in the former are virtually comparable to those of the latter. As a result, the effects on the resulting coding efficiency are negligible. Some of these limitations are:

- As mentioned at the beginning of this section, the pre-analysis stage performs its operations on the original input frames. The encoder, however, makes use of the neighbouring reconstructed samples for the intra prediction. Nevertheless, considering that the reconstructed samples are fairly similar to the original ones, especially for low QP values, it is reasonable to assume that the prediction in both cases will be similar enough to determine which prediction modes are best.
- By the same token, there is no neighbour information other than the one that this stage itself calculates. As a result, it is not possible to know which samples might be or might not be available for prediction at the encoding stage. However, the standard defines some sample substitution techniques that ensure that there is always present the required neighbouring samples to perform intra prediction [3]. In this way, it is possible to assume that in many cases the original samples will be

similar to those that the encoder will create in the substitution process, and thus this is the approach that the proposed pre-analysis algorithm follows.

In addition to these two considerations, it also has to be taken into account that the encoder makes use of an R–D model such as:

$$J = \text{Distortion}_{\text{intra}} + \lambda \cdot \text{Rate}_{\text{intra}} \tag{1}$$

where J represents the cost function, $\text{Distortion}_{\text{intra}}$ the distortion function (typically Hadamard), λ is the Lagrangian multiplier which depends on the QP, and $\text{Rate}_{\text{intra}}$ represents the bits required to code the prediction mode. Nevertheless, while it is possible to calculate the λ value and estimate the $\text{Distortion}_{\text{intra}}$ in the pre-analysis stage, it is not feasible to predict the value of $\text{Rate}_{\text{intra}}$, given that the neighbouring blocks have not been entropy-coded at that moment. For this reason, only the distortion is estimated in this stage, postponing the R–D calculation until the moment in which the encoder performs intra prediction.

With regard to the integration of this algorithm, it has to be noted that the pre-analysis stage can be part of any HEVC-compliant encoder. However, to show how these encoders can benefit from this algorithm, the remainder of this section will detail its integration into the HEVC test model (HM) reference software [27]. This encoder performs a three step intra prediction algorithm to obtain the best intra mode for a given PU. In the first step, a rough prediction is carried out, in which all the 35 intra modes are tested. In the second step, a set of the best candidates found in the previous step is considered along with the most probable modes (MPM), performing a more thorough prediction and a transform of the resulting residual. Finally, the last step consists on choosing the best candidate out of the ones from the second step and performing the full RQT partitioning, which results in the definitive R–D cost for the corresponding intra mode.

As can be seen, the integration of our proposal in this case is fairly immediate. As the proposed pre-analysis algorithm already estimates the prediction cost of the 35 intra modes, the first step of the encoding stage in HM can be skipped. Therefore, the set of best candidates for the second step would be simply obtained from the pre-analysis scores. While it might seem there is no time reduction from this, actually the operations performed in the pre-analysis stage are much simpler than the ones in the encoding stage. For example, there is a lower number of conditions to check, and also the distortion algorithm being used is SAD and not Hadamard, as mentioned before.

4.2 Pre-analysis algorithm for inter prediction

While the algorithm described before allows to reduce the processing time of the intra module, it is the inter prediction module the one that involves larger encoding times for configurations other than All Intra. For this reason, the proposed pre-analysis stage has been also designed to estimate preliminary information from the input pictures to later use it in the ME stage as shown in Fig. 4. As can be seen, the pre-analysis stage performs a fast motion estimation, whereas the ME module makes use of the

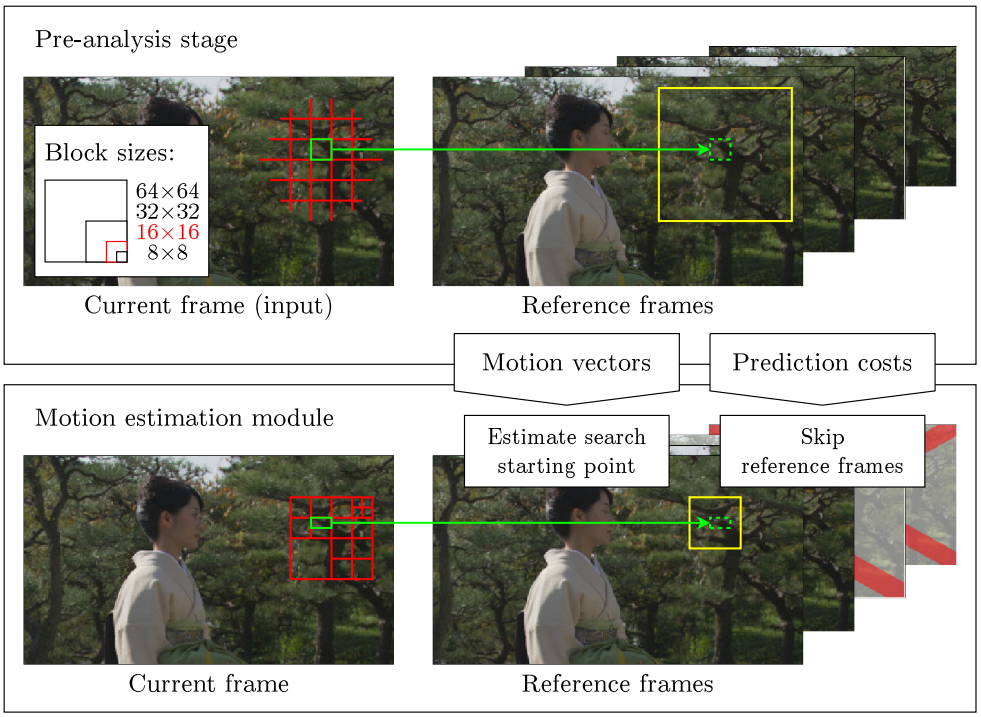


Fig. 4 Interaction between the pre-analysis stage and the ME algorithm in the proposed approach for inter prediction

estimated information to reduce the number of reference frames and the size of the search range. Both steps will be detailed in the following paragraphs.

With regard to the inter prediction module, the main aim of the pre-analysis stage is to estimate the motion of a frame in a fast way. This is achieved by carrying out a similar process to that of the ME module. The latter, however, involves large computation times, as it needs to test every possible PU partition for every depth level in the partitioning tree. For this reason, as described in Sect. 4.1 this operation is simplified in the pre-analysis stage using fixed-size block partitions, resulting in a grid of motion vectors (MV) and estimated costs for every reference picture of the current frame. Experimental results showed that performing an estimation for all the possible 2N × 2N PUs (64 × 64, 32 × 32, 16 × 16 and 8 × 8) provides much more useful information to the encoder than only one size, as these blocks and their associated information can be adapted to the PUs in a more flexible way. Accordingly, our proposed approach splits the current frame into blocks of all the aforementioned sizes, and performs a fast motion estimation on every reference frame. Therefore, this stage results in the estimation of the MVs and their cost for every block and reference picture.

Similar to the intra-based pre-analysis algorithm, it is not possible to know the exact neighbouring information for each block. However, it is possible to estimate it from previous pre-analysis blocks. In fact, to obtain the most reliable MV map possible, the following considerations are taken into account:

– The ME algorithm is the same as the one used in the encoding stage, although limiting its precision to integer-pel samples, given that only an approximated estimation is required.
– The neighbour MVs of the current block are considered as predictors similar to the advanced motion vector prediction (AMVP) feature of the HEVC standard [3].
– When estimating the motion of 8 × 8 blocks, a pattern of 16 × 16 pixels is used instead, as capturing the context of the block has been proved to provide more accurate results.
– The same Lagrangian multiplier R–D optimization model as in (1) is used, but adapting the corresponding functions to the inter prediction module. In other words, Dist_{MV} would be the distortion function in terms of the sum of absolute differences (SAD), λ is the Lagrangian multiplier for inter prediction, and $Rate_{MV}$ would represent the bits required to code the MV.

Once the pre-analysis stage has been carried out, the ME module has at its disposal the information of the motion of the current frame. While this information can be used in several ways, our aim is to reduce the encoding time of this module. In this regard, the estimated values will be used to reduce the number of tested reference frames, and to predict the starting point of the motion search.

First, it is necessary to determine how to adapt the estimated information from the pre-analysis blocks to each PU. If the PU has a 2N × 2N size, then it exists a pre-analysis block that directly matches the PU. For the remaining sizes, the corresponding PU can be formed from the two co-located pre-analysis blocks, except for 8 × 4 and 4 × 8 sizes, in which case the 8 × 8 pre-analysis block is used. In the former case, the estimated MV and prediction cost can be re-used from the pre-analysis block, but in the latter, both MVs are tested, taking only the one that provides the lowest cost. Consequently, the result of this first step for each PU is one MV and the corresponding prediction cost for each reference frame.

As anticipated before, the estimated prediction costs brings the encoder the possibility to select those reference frames that should be tested for each PU. However, this decision becomes more complex when biprediction is taken into account, as the prediction is obtained combining two reference frames. It is also statistically true that the reference frames utilised for non-2N × 2N PUs are quite frequently the ones used for the 2N × 2N PU in the same depth level. For this reason, our proposal tests all the reference frames for 2N × 2N PUs, so that if the best reference frame estimated by the pre-analysis stage for a non-2N × 2N matches the one used in the 2N × 2N PU, it will only use that frame, skipping the rest. In the case of biprediction, if the reference picture chosen by the pre-analysis stage matches one of the two frames used in the 2N × 2N block, both are selected and the rest are skipped. If the reference indices do not match, then all the reference pictures are tested for that PU to prevent erroneous decisions.

Whereas the estimated prediction costs can be used to select the reference frames, the MVs can also help reducing the computational cost of the ME module. More specifically, the MVs obtained from the pre-analysis stage can be used as starting point for the motion search, notably reducing the size of the search area from the default 128 × 128 region to, for example, 8 × 8. Consequently, the default TZSearch

algorithm [28] can also be replaced with a local search algorithm. In our case, a hexagon-based plus refinement pattern has been used, which reduces the number of SAD operations performed by the ME module. Additionally, to make sure that the estimated MV is a good candidate for the search, it is checked whether this vector is close to one of the AMVP candidates. If it is in a 3-pixel radio of any AMVP candidate, the MV is chosen as starting point and the closest predictor is selected. If not, the prediction costs of all of them are calculated, choosing the starting point and the predictor according to the MV or predictor providing the lowest cost.

5 Performance evaluation

The experiments have been executed on the HM 16.6 reference software, whose choice was motivated by the fact that it is the most widespread encoder in the research community, establishing a point of comparison between contributions. Tests have been conducted following the guidelines and coding conditions provided by the document elaborated by the JCT-VC [29]. The QP values being tested are 22, 27, 32 and 37. The encodings have been carried out using the Main profile and 8-bit depth. Sequences of classes A to D according to the JCT-VC classification have been used, which include the following resolutions: 2560 $\times$ 1600 (A), 1920 $\times$ 1080 (B), 832 $\times$ 480 (C), and 416 $\times$ 240 (D). The rest of the configuration parameters have all been kept to their default values.

All the four default configurations have been tested: all intra, random access, low delay, and low delay P. While all of them make use of spatial prediction, only the last three are allowed to use temporal prediction. In those cases, the search range of the pre-analysis stage was kept to the default value used by the baseline encoder, i.e. 64 pixels (128 $\times$ 128 pixels search area), while the ME search range was reduced to 4 pixels (8 $\times$ 8 area) when the pre-analysis algorithm was enabled. In addition, in the case of low delay P, only P-slices are used in the encoding, which means that the ME algorithm is not allowed to perform biprediction.

The hardware platform used in the experiments is composed of an Intel® Xeon® E5-2630L v3 CPU running at 1.80 GHz and 16 GB of main memory. The encoder has been compiled with GCC 4.8.5-4 and executed on CentOS 7 (Linux 3.10.0-327). Turbo Boost has been disabled to achieve the reproducibility of the results.

The results will be shown in terms of time reduction (TR) and Bjøntegaard delta rate (BD-rate). The BD-rate is a measure of coding efficiency that represents the percentage of bit rate variation between two encodings with the same objective quality [30]. The arithmetic mean of the results of all the sequences will also be presented.

Table 1 shows the results obtained by the encoder when the pre-analysis algorithm is enabled for the All Intra configuration. It has to be noted that, in this case, the pre-analysis stage would not calculate any motion information, as there is no temporal prediction in this configuration. As can be seen, making use of the information gathered by the pre-analysis stage, the encoder is able to save between 5.57 and 9.15 % of the total encoding time (7.07 % average). As mentioned before, these time savings are derived from the fact that the pre-analysis stage performs less demanding operations compared to the baseline encoder to carry out intra prediction, more specifically

Table 1 Results of the proposed algorithm for the all intra configuration

Class	Video sequence	BD-rate (%)	Encoding TR (%)
A	Traffic	0.51	7.96
	PeopleOnStreet	0.59	7.87
	NebutaFestival	0.20	6.23
	SteamLocomotive	0.43	8.62
B	Kimono	0.53	9.15
	ParkScene	0.40	7.31
	Cactus	0.47	7.47
	BasketballDrive	0.59	6.90
	BQTerrace	0.28	6.39
C	BasketballDrill	0.46	7.29
	BQMall	0.53	6.65
	PartyScene	0.38	5.74
	RaceHorses	0.40	6.81
D	BasketballPass	0.60	7.82
	BQSquare	0.43	5.57
	BlowingBubbles	0.41	5.96
	RaceHorses	0.56	6.44
Mean values		0.46	7.07

because of the substitution of the Hadamard algorithm with the SAD operation. But even if Hadamard was used in the pre-analysis stage, it would still be possible to obtain time savings, which demonstrates the fact that this stage is less computationally expensive, as it needs to check less conditions than the encoder.

This table also shows, however, that the pre-analysis algorithm has an effect on the coding efficiency. In the experiments, the obtained average BD-rate is 0.46 %, which is considered negligible for most multimedia applications. This means that the bit rate is increased by less than 1 % for the same objective quality. This small penalization can be justified by the limitations of the pre-analysis stage stated in Sect. 4.1, which result in sub-optimal decisions compared to the ones the encoder would have made. It should also be mentioned that these divergences are larger for smaller PUs such as 4×4, as they are typically harder to predict than the larger ones. Finally, the numeric results also show that there is some correlation between the BD-rate and the QP value used in the encoding, so that the higher the former, the higher the latter. This is probably related to the fact that the pre-analysis stage makes use of the original samples instead of the reconstructed ones.

As a consequence of the pre-analysis algorithm, the encoding time is redistributed as shown in Fig. 5. These timing values have been extracted from the average profiles of all the tested sequences and QP values. As can be seen, a notable part of the time devoted to the intra module has been reduced, which is represented in the figure by the blue bar. The reduced part corresponds to the first step of the three step intra algorithm

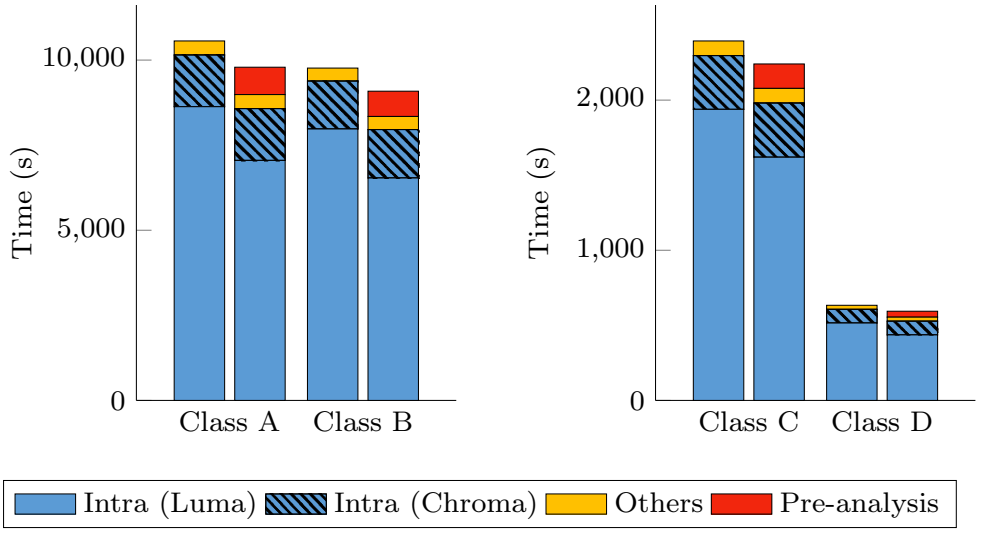


Fig. 5 Profiling of the baseline (*left side bar*) compared to the proposal (*right side bar*) for the All Intra configuration

implemented in HM, which is now performed in the pre-analysis algorithm. It can be seen that the decrement in time experienced by the encoder is larger than the increment introduced by the pre-analysis algorithm. It has to be noted that this small pre-analysis overhead does not involve any latency, as the achieved time reduction is larger than the time spent on this stage.

Likewise, Table 2 shows the results obtained by the pre-analysis algorithm when using the Random Access configuration. The results show that almost one third of the total number of reference frames are skipped thanks to the motion information estimated by the proposed algorithm. It can also be seen that the percentage of skipped frames does not vary substantially depending on the sequence. Regarding the number of SAD operations performed in the integer-pel ME (IME), it can be seen that more than 80 % of these operations have been saved, dramatically reducing the time spent in this step. Not only is this value influenced by skipping frames, but also because of reducing the search area and using a local search area. As a result, the total encoding time can be reduced between 12.77 and 18.32 % (16.19 % average) at the expense of a negligible increase of 0.44 % in coding efficiency on average, which is practically imperceptible in most scenarios.

Figure 6 shows the redistribution of time caused by the pre-analysis algorithm with the Random Access configuration. First of all, it can be seen that the processing time of the intra module has not been reduced to the same extent as in the All Intra configuration. This can be explained by the fact that the time devoted to this module is much more limited, and also because the intra-based pre-analysis algorithm has been restricted to I-slices, given that the encoder implements a fast intra skip algorithm for other types of slices, and it would not take advantage of the proposed stage. In turn, a notable part of the time devoted to the inter prediction module has been reduced. On the one hand, this is caused by the fact that one third of the reference frames is skipped. On the other hand, the number of SAD operations performed in the uni-

Table 2 Results of the proposed algorithm for the random access configuration

Class	Video sequence	BD-rate (%)	Encoding TR (%)	Ref. frames skipped (%)	IME SAD savings (%)
A	Traffic	0.46	16.44	27.46	73.30
	PeopleOnStreet	0.33	16.36	30.44	90.08
	NebutaFestival	0.26	12.77	26.00	82.52
	SteamLocomotive	−0.58	18.32	32.37	89.71
B	Kimono	0.27	17.23	26.26	85.84
	ParkScene	0.40	16.14	25.29	77.18
	Cactus	0.46	17.75	32.72	85.47
	BasketballDrive	0.33	17.95	27.99	89.89
	BQTerrace	0.74	14.93	25.78	73.65
C	BasketballDrill	0.23	17.37	33.35	88.22
	BQMall	0.40	16.27	28.38	82.94
	PartyScene	0.74	14.35	27.54	80.99
	RaceHorses	0.93	17.77	30.91	92.13
D	BasketballPass	0.23	16.73	30.99	89.67
	BQSquare	0.95	13.88	25.32	67.83
	BlowingBubbles	0.63	14.34	29.30	75.82
	RaceHorses	0.76	16.68	29.89	90.04
Mean values		0.44	16.19	28.82	83.25

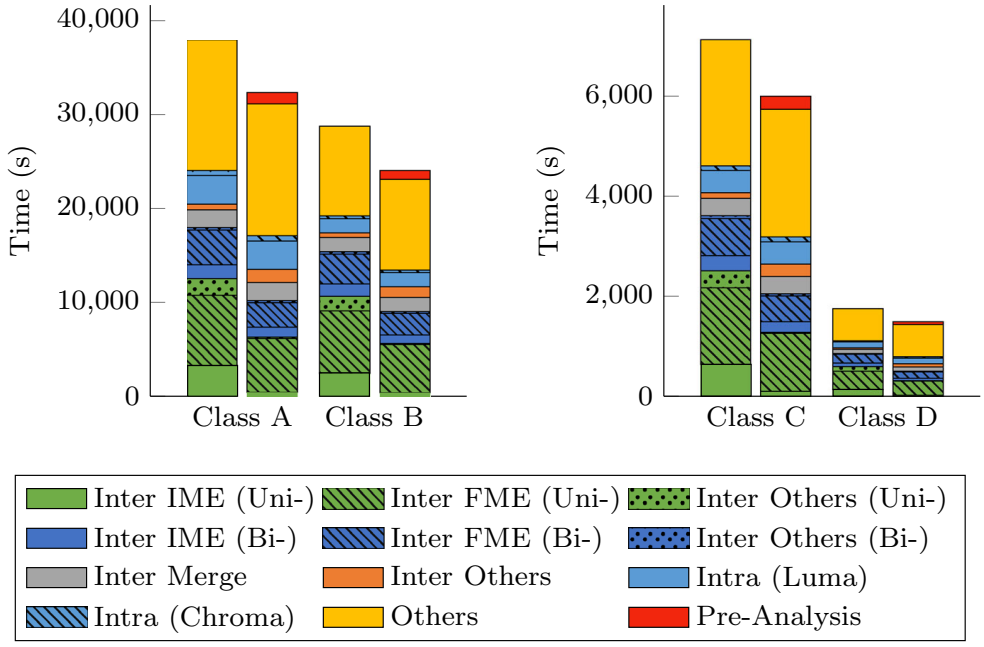


Fig. 6 Profiling of the baseline (*left side bar*) compared to the proposal (*right side bar*) for the random access configuration

Table 3 Results of the proposed algorithm for the low delay configuration

Class	Video sequence	BD-rate (%)	Encoding TR (%)	Ref. frames skipped (%)	IME SAD savings (%)
A	Traffic	0.90	18.49	26.21	78.37
	PeopleOnStreet	0.51	22.60	33.35	93.36
	NebutaFestival	0.38	15.64	23.44	87.47
	SteamLocomotive	0.21	21.31	30.93	91.93
B	Kimono	0.22	21.51	25.85	91.34
	ParkScene	0.49	18.95	23.92	84.47
	Cactus	0.52	22.00	35.77	89.54
	BasketballDrive	0.40	22.24	27.60	92.83
	BQTerrace	0.84	16.16	22.76	77.81
C	BasketballDrill	0.20	22.09	37.53	91.21
	BQMall	0.61	21.06	30.36	88.46
	PartyScene	0.77	18.27	29.55	84.44
	RaceHorses	0.83	23.65	36.24	94.67
D	BasketballPass	0.47	22.95	35.42	93.00
	BQSquare	1.47	15.08	22.69	70.49
	BlowingBubbles	0.73	19.54	33.00	81.03
	RaceHorses	0.77	23.43	36.19	93.79
Mean values		0.61	20.29	30.05	87.31

predictional integer ME is almost imperceptible. It can also be seen, however, that some more time is spent on the "ME Others" block, where the pre-analysis decisions and cost calculations take place, as well as the AMVP algorithm, which was placed in the "ME Others (Uni-)" in the baseline encoder. The pre-analysis itself introduces a small overhead that does not involve any latency, as the achieved time reduction is larger than the time spent on this stage.

Table 3 and Fig. 7 contain the corresponding results for the Low Delay configuration, which uses bipredictive temporal prediction as Random Access, but forming a different coding pattern. In this case, the table shows an increase of the percentage of skipped frames, which can be explained by the fact that the reference lists usually hold more frames than in Random Access, and as a consequence it is more likely to discard a reference frame. This has a direct influence on the SAD savings of the integer ME algorithm, rising up to almost 90 %. As a result, the obtained time saving ranges from 15.08 to 23.43 % (20.29 % average), which is higher than with other configurations, but it also involves a slightly greater BD-rate penalty of 0.61 % on average. The time redistribution follows the pattern of the previous tested configuration. It has to be noted also that the ME involves a larger amount of processing time compared with Random Access, which has a positive effect on the achieved speed-up.

Finally, Table 4 shows the results of the Low Delay P configuration, which is very similar to the Low Delay pattern, but ruling out B-slices, and hence biprediction.

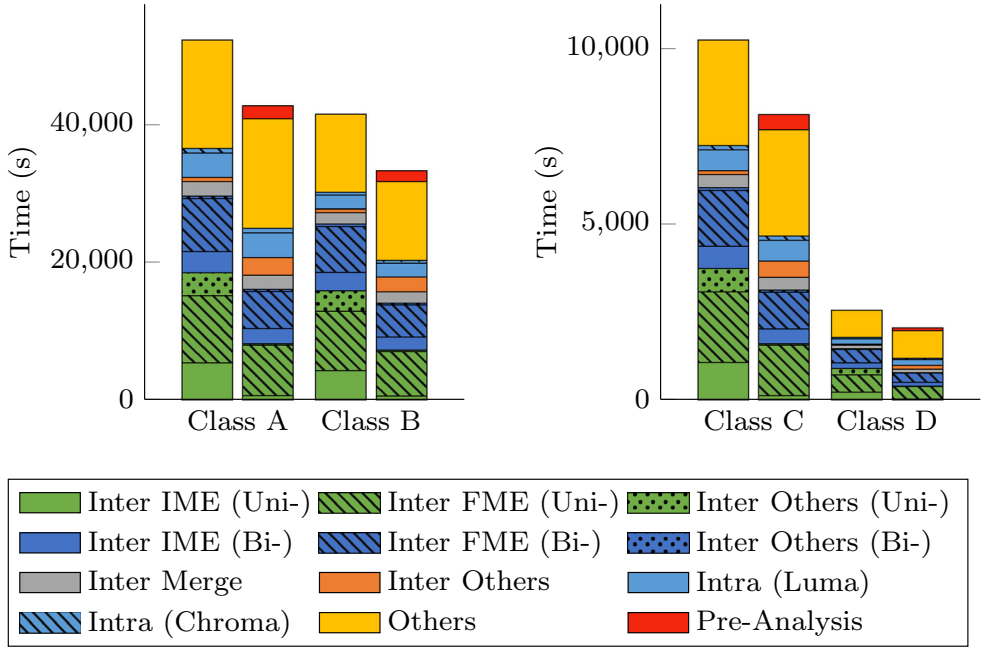


Fig. 7 Profiling of the baseline (*left side bar*) compared to the proposal (*right side bar*) for the low delay configuration

Table 4 Results of the proposed algorithm for the low delay P configuration

Class	Video sequence	BD-rate (%)	Encoding TR (%)	Ref. frames skipped (%)	IME SAD savings (%)
A	Traffic	0.42	17.23	29.33	79.83
	PeopleOnStreet	0.22	19.92	33.65	93.75
	NebutaFestival	0.05	14.97	27.92	88.50
	SteamLocomotive	0.04	20.13	32.94	92.31
B	Kimono	0.10	20.09	28.54	91.63
	ParkScene	0.34	17.18	26.40	85.06
	Cactus	0.27	18.46	35.10	89.98
	BasketballDrive	0.04	21.19	30.18	93.28
	BQTerrace	0.49	14.90	25.86	78.71
C	BasketballDrill	0.02	18.62	36.89	91.80
	BQMall	0.35	17.96	31.32	89.01
	PartyScene	0.57	15.85	31.47	85.68
	RaceHorses	0.42	20.92	36.89	95.08
D	BasketballPass	0.26	19.72	35.45	93.39
	BQSquare	0.94	12.97	24.30	71.90
	BlowingBubbles	0.75	16.06	35.83	82.78
	RaceHorses	0.40	20.17	35.76	94.19
Mean values		0.33	18.02	31.64	88.05

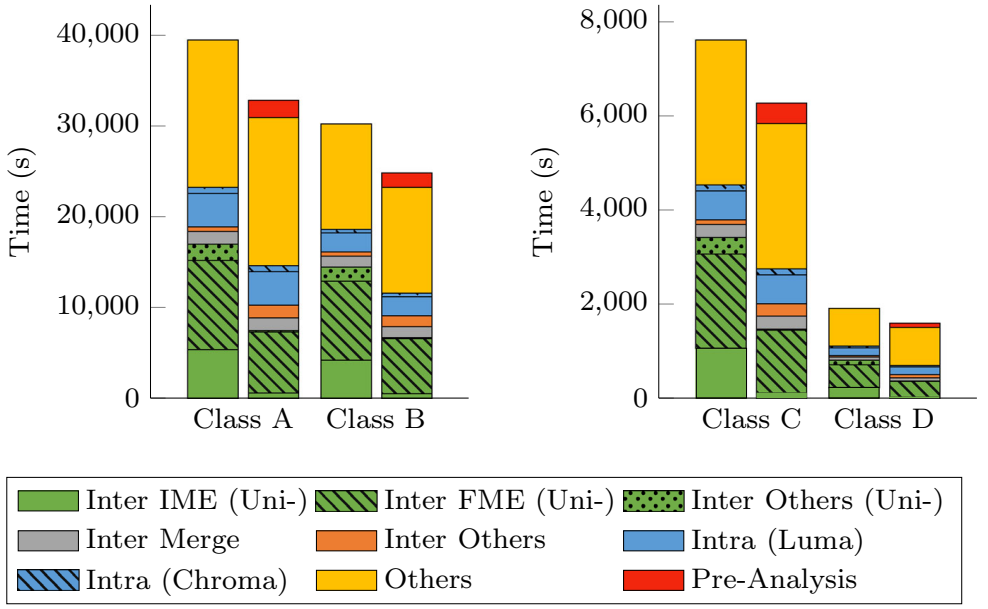


Fig. 8 Profiling of the baseline (*left side bar*) compared to the proposal (*right side bar*) for the low delay P configuration

Consequently, the amount of time devoted to the ME is smaller than with previous configurations. Nonetheless, the proposed algorithm is able to skip more reference frames than any other configuration, and save between 12.97 and 21.19 % of the total encoding time (18.02 % average). Helped by the absence of biprediction, the corresponding average BD-rate loss is 0.33 %, which is negligible for most applications. Once again, the redistribution of time for this configuration is shown in Fig. 8, in which it can be seen that there are no significant changes with respect to the previous ones.

6 Conclusions and future work

The advent of HEVC has opened the door to new applications and video formats such as UHD with the aim of fulfilling the demands for quality of experience of the market. However, the improved coding efficiency of HEVC has also led to a notable increase in the computational requirements of the encoder. As a consequence, HEVC-based encoders require efficient algorithms to achieve real-time encoding. Accordingly, this paper proposes a pre-analysis algorithm designed to estimate the intra prediction cost for every intra mode, and also some motion information for every PU in a fast way. This information is later used in the encoder to select the intra modes to be tested, to select the reference frames utilised in the ME stage, and to reduce the search area by performing a local search. As a consequence, the total encoding time is reduced. Additionally, this paper also details some considerations that have been taken into account in the design of the pre-analysis algorithm given the particularities of HEVC.

An experimental evaluation of the algorithm has shown that between 5.57 and 9.15 % of the encoding time can be saved for All Intra configurations. With respect to

other configurations that make use of temporal prediction, these savings range from 12.77 to 18.32 % for Random Access. In any of the cases, the impact of the algorithm in terms of BD-rate is typically much lower than 1 %, which is practically negligible for most multimedia applications.

Future works include considering the inter and intra costs to determine whether the encoder should check only one of the two prediction modes for a given CU. In a higher layer of abstraction, it could be also possible to determine the CTU partitioning of a frame based on these costs. Furthermore, this algorithm could be implemented in a heterogeneous platform, so that a coprocessor could assist the encoding process by providing preliminary information.

Acknowledgements This work was jointly supported by the Spanish Ministry of Economy and Competitiveness (MINECO) and the European Commission (FEDER funds) under the project TIN2015-66972-C5-2-R, and by the Spanish Ministry of Education, Culture and Sports under the Grant FPU13/04601.

References

1. ISO/IEC, ITU-T (2016) Advanced video coding for generic audiovisual services. ITU-T recommendation H.264 and ISO/IEC 14496-10 (version 10)
2. ISO/IEC, ITU-T (2015) High efficiency video coding (HEVC). ITU-T recommendation H.265 and ISO/IEC 23008-2 (version 3)
3. Sullivan GJ, Ohm JR, Han Woo-Jin, Wiegand T (2012) Overview of the high efficiency video coding (HEVC) standard. IEEE Trans Circuits Syst Video Technol 22(12):1649–1668. doi:10.1109/TCSVT.2012.2221191
4. Ohm JR, Sullivan GJ, 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. doi:10.1109/TCSVT.2012.2221192
5. Cebrián-Márquez G, Martínez JL, Cuenca P (2016) Two-stage intra prediction algorithm for HEVC. In: 16th International Conference on Mathematical Methods in Science and Engineering (CMMSE)
6. Cebrián-Márquez G, Chi Ching Chi, Martínez JL, Cuenca P, Álvarez-Mesa M, Sanz-Rodríguez S, Juurlink B (2015) Reducing HEVC encoding complexity using two-stage motion estimation. In: IEEE International Conference on Visual Communications and Image Processing (VCIP)
7. Cebrián-Márquez G, Martínez JL, Cuenca P (2016) A pre-analysis algorithm for fast motion estimation in HEVC. In: IEEE International Conference on Image Processing (ICIP)
8. Misra K, Segall A, Horowitz M, Shilin Xu, Fuldseth A, Zhou Minhua (2013) An overview of tiles in HEVC. IEEE J Sel Topics Signal Process 7(6):969–977. doi:10.1109/JSTSP.2013.2271451
9. Henry F, Pateux S (2011) Wavefront parallel processing. Tech. Rep. JCTVC-E196
10. Chi CC, Álvarez-Mesa M, Juurlink B, Clare G, Henry F, Pateux S, Schierl T (2012) Parallel Scalability and efficiency of HEVC parallelization approaches. IEEE Trans Circuits Syst Video Technol 22(12):1827–1838. doi:10.1109/TCSVT.2012.2223056
11. Piñol P, Migallón H, López-Granado O, Malumbres MP (2015) Slice-based parallel approach for HEVC encoder. J Supercomput 71(5):1882–1892. doi:10.1007/s11227-014-1371-y
12. Piñol P, Migallón H, López-Granado O, Malumbres MP (2014) Parallel strategies analysis over the HEVC encoder. J Supercomput 70(2):671–683. doi:10.1007/s11227-014-1121-1
13. Yan C, Zhang Y, Jizheng X, Dai F, Zhang J, Dai Q, Feng W (2014) Efficient parallel framework for HEVC motion estimation on many-core processors. IEEE Trans Circuits Syst Video Technol 24(12):2077–2089. doi:10.1109/TCSVT.2014.2335852
14. Cebrián-Márquez G, Hernández-Losada JL, Martínez JL, Cuenca P, Tang Minhao, Wen Jiangtao (2015) Accelerating HEVC using heterogeneous platforms. J Supercomput 71(2):613–628. doi:10.1007/s11227-014-1313-8
15. Radicke S, Hahn JU, Wang Qi, Grecos C (2014) Bi-predictive motion estimation for HEVC on a graphics processing unit (GPU). IEEE Trans Consum Electron 60(4):728–736. doi:10.1109/TCE.2014.7027349

16. Cho S, Kim M (2013) Fast CU splitting and pruning for suboptimal CU partitioning in HEVC intra coding. IEEE Trans Circuits Syst Video Technol 23(9):1555–1564. doi:10.1109/TCSVT.2013.2249017
17. Shen L, Zhang Z, An P (2013) Fast CU size decision and mode decision algorithm for HEVC intra coding. IEEE Trans Consum Electron 59(1):207–213. doi:10.1109/TCE.2013.6490261
18. Zhang H, Ma Z (2014) Fast intra mode decision for high efficiency video coding (HEVC). IEEE Trans Circuits Syst Video Technol 24(4):660–668. doi:10.1109/TCSVT.2013.2290578
19. Wie Jiang, Hanjie Ma, Yaowu Chen (2012) Gradient based fast mode decision algorithm for intra prediction in HEVC. In: 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), pp 1836–1840. doi:10.1109/CECNet.2012.6201851
20. Shen L, Zhang Z, Liu Z (2014) Adaptive inter-mode decision for HEVC jointly utilizing inter-level and spatiotemporal correlations. IEEE Trans Circuits Syst Video Technol 24(10):1709–1722. doi:10.1109/TCSVT.2014.2313892
21. Rhee CE, Lee K, Kim TS, Lee H-J (2012) A survey of fast mode decision algorithms for inter-prediction and their applications to high efficiency video coding. IEEE Trans Consum Electron 58(4):1375–1383
22. Garrett-Glaser J (2009) A novel macroblock-tree algorithm for high-performance optimization of dependent video coding in H.264/AVC. Tech. Rep., Department of Computer Science. Harvey Mudd College
23. VideoLAN (2016) x264 source code repository. http://git.videolan.org/git/x264.git
24. VideoLAN (2016) x265 source code repository. http://hg.videolan.org/x265
25. Lin S, Au OC, Cong Z, Huang FH (2014) Rate distortion modeling and adaptive rate control scheme for high efficiency video coding (HEVC). In: IEEE International Symposium on Circuits and Systems (ISCAS), pp 1933–1936. doi:10.1109/ISCAS.2014.6865539
26. Mallikarachchi T, Fernando A, Arachchi HK (2014) Efficient coding unit size selection based on texture analysis for HEVC intra prediction. In: IEEE International Conference on Multimedia and Expo (ICME), pp 1–6. doi:10.1109/ICME.2014.6890319
27. ISO/IEC, ITU-T (2016) HEVC test model (HM) reference software. https://hevc.hhi.fraunhofer.de/
28. Kibeya H, Belghith F, Loukil H, Ben Ayed MA, Masmoudi N (2014) TZSearch pattern search improvement for HEVC motion estimation modules. In: 1st International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), pp 95–99. doi:10.1109/ATSIP.2014.6834584
29. Bossen F (2013) Common test conditions and software reference configurations. Tech. Rep. JCTVC-L1100
30. Bjøntegaard G (2001) Calculation of average PSNR differences between RD-curves. Tech. Rep. VCEG-M33, ITU-T Video Coding Experts Group (VCEG)

CHAPTER 5

Look-Ahead Partitioning Based on Motion Information for HEVC

- **Title**: Look-Ahead Partitioning Based on Motion Information for HEVC.
- **Authors**: Gabriel Cebrián-Márquez, José Luis Martínez, and Pedro Cuenca.
- **Type**: journal paper.
- **Journal**: IEEE Transactions on Circuits and Systems for Video Technology.
- **Publisher**: Institute of Electrical and Electronics Engineers (IEEE).
- **ISSN**: 1051-8215.
- **Status**: under review.
- **Submission date**: November 2017.
- **JCR IF/ranking**: 3.599/Q1 (JCR2016).

Look-Ahead Partitioning Based on Motion Information for HEVC

Gabriel Cebrián-Márquez, José Luis Martínez, and Pedro Cuenca

***Abstract*—The High Efficiency Video Coding (HEVC) standard has shown large improvements in coding efficiency compared with previous standards. In particular, HEVC outperforms H.264/MPEG-4 Advanced Video Coding (AVC) by up to 50% in terms of bitrate reduction for similar perceptual quality. This improvement is the result of the introduction of new coding tools which enable the representation of data using fewer bits, but at the cost of long computation times. One of the most significant tools introduced by HEVC is the novel quadtree-based structure called the coding tree unit (CTU), which can be subsequently split into coding units (CUs), prediction units (PUs) and transform units (TUs), providing huge flexibility in the encoding. However, selecting the optimal tree partitioning requires the evaluation of a huge number of possibilities, which constitutes the most complex operation for the encoder. In order to tackle this task, this paper proposes a CTU/CU partitioning algorithm based on a look-ahead stage. This stage performs a fast motion estimation that provides preliminary information to the encoder, including estimate distortion costs, which enables the building of the least-cost quadtree. On the basis of this tree, the evaluation performed by the encoder is restricted to a subset of PUs. After a thorough statistical analysis of numerous CU/PU combinations, we propose two different configurations that prioritize each of the target variables: coding efficiency and time reduction. In the former case, 56.80% of the encoding time can be saved at the cost of a 2.38% increase in bitrate, while a larger reduction of 61.41% is achieved with a 3.11% increase in bitrate in the latter case.**

***Index Terms*—HEVC, fast encoding, pre-analysis, look-ahead, partitioning.**

I. Introduction

In early 2013, the High Efficiency Video Coding (HEVC) standard became the latest joint video project of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) organizations, as part of the Joint Collaborative Team on Video Coding (JCT-VC) [1]. Several revisions have been released since then, including the extensions to support additional application scenarios, such as range extensions with enhanced precision and color format support, multiview and 3D, scalable coding, and screen content coding. In this way, the fourth and latest version of this standard was released in December 2016, although there are still very active ongoing development efforts to define additional features and profiles.

The new HEVC standard was conceived as the evolution of its predecessor, namely H.264/MPEG-4 Advanced Video Coding (AVC) [2]. Over the last decade, content providers and manufacturers have widely adopted H.264/MPEG-4 AVC in countless scenarios because of its coding efficiency. Nevertheless, the increasing demands of the market for a better quality of experience, and the advent of new video formats such as ultra high definition (UHD) or improved color representations motivated the development of HEVC. In this way, the aim of this new standard is not only to provide support to these enhancements, but also to achieve good coding efficiency in order to enable the delivery of these contents over bandwidth-constrained networks.

Compared with H.264/MPEG-4 AVC, HEVC is able to achieve 35.4% bitrate savings on average for an equal peak signal-to-noise ratio (PSNR), or around a 50% bitrate saving based on subjective quality assessments [3]. This improvement is the result of all the novel encoding tools introduced in HEVC, among which it is worth mentioning the hierarchical block structure used to divide the pictures into partitions. This structure is called the coding tree unit (CTU), and can be subsequently divided into coding units (CUs), prediction units (PUs) and transform units (TUs). This wide range of possibilities provides great flexibility to the standard when adapting the partitions to the intrinsic features of the picture, leading to better coding efficiency.

The process of determining the optimal partitioning requires, however, testing numerous block combinations. As a consequence, the computational complexity of HEVC has increased exponentially compared with prior standards [4]. In fact, this complexity represents one of the most significant obstacles to successfully deploying HEVC in real-world scenarios. For this reason, the research community is making intense efforts to design fast encoding algorithms that enable a reduction in the total encoding time. In this regard, this paper proposes a fast CTU/CU partitioning algorithm for the HEVC encoder that omits the evaluation of CUs and PUs that are unlikely to provide optimal results. As a result, the total encoding time is reduced with imperceptible losses in terms of coding efficiency, enabling the feasibility of the standard in real-time scenarios.

The method proposed in this paper bases the partitioning decision on the motion information obtained in a preliminary stage. In order to do so, the input pictures are divided into square blocks of different sizes, which are later used by the preliminary stage to perform a fast motion estimation (ME). In this way, the result of this stage, also called look-ahead, is an estimate of the distortion cost for each block and reference frame. At this point, the idea behind the proposed method is that this motion information can be used to predict the partitioning quadtree of the CTUs into which the picture is

The authors are with the High-Performance Networks and Architectures group of the University of Castilla-La Mancha. Campus Universitario s/n 02071, Albacete, Spain. E-mail: {Gabriel.Cebrian, JoseLuis.Martinez, and Pedro.Cuenca}@uclm.es

Manuscript received October 12, 2017.

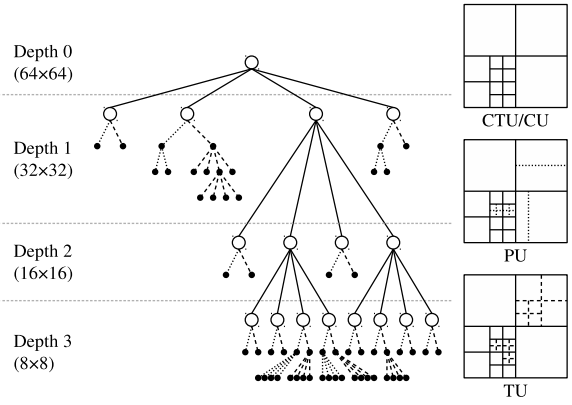


Fig. 1. Quadtree partitioning structure in HEVC, including PUs and TUs.

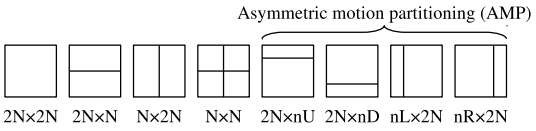


Fig. 2. PU partitions defined in HEVC.

divided. Nevertheless, this quadtree does not always match the one the encoder would have chosen after a rate-distortion optimization (RDO) process, resulting in coding efficiency penalties. As a preventive way to avoid these, not only the PUs of the quadtree defined by the look-ahead stage are checked, but also some additional PUs located in the parent and child CUs. In this way, an experimental evaluation of different PU configurations will be performed to study their effect in the resulting quality and compression ratio. As a result, the total encoding time is notably reduced, whereas the coding efficiency is maintained.

While the look-ahead stage has proven to be an effective way of providing support to the encoding process in current real-time encoders, this paper, to the best of the authors' knowledge, is the first scientific approach to CTU/CU partitioning using the motion information provided by this stage. Additionally, the information extracted by the proposed look-ahead can be utilized for many other purposes, such as rate control, bit allocation, or encoding pattern determination, among others, without requiring modifications to the algorithm. However, the analysis of these approaches is beyond the scope of this paper.

The rest of this paper is organized as follows. Section II presents a short summary of the main aspects of HEVC. Section III covers some of the most relevant related works in the area. The look-ahead algorithm and the proposed partitioning algorithm are described in Section IV. An experimental evaluation is presented in Section V, along with the selection of the optimal PU configuration. Finally, conclusions are drawn in Section VI.

II. Technical Background

The HEVC standard represents the evolution of all the previous standards and, in particular, of H.264/MPEG-4 AVC, since they are both based on the same block-based approach. In this way, not only does HEVC introduce new coding tools, but it also reformulates and improves other tools that were already present in prior standards. This leads to a notable increase in coding efficiency compared with H.264/MPEG-4 AVC. One of the most important enhancements is the introduction of a new partitioning scheme which completely dispenses with the term macroblock (MB). Instead, the new partitions are now called CTUs, whose size is typically 64×64 pixels, although 32×32 and 16×16 are also admitted. Each CTU can be recursively partitioned into four square sub-blocks called CUs, down to a minimum size of 8×8 pixels, thus forming a quadtree as shown in Fig. 1. With the aim of providing higher flexibility to the standard, CUs are subsequently composed of PUs and TUs.

The HEVC standard defines up to eight different possible inter and intra prediction partitions, which are depicted in Fig. 2 together with their nomenclature. The four on the right of the figure correspond to the asymmetric motion partitioning (AMP), and are new in this standard. This variety of sizes enables the encoder to determine the optimal trade-off between rate and distortion in a process called RDO, using a rate-distortion (R-D) model to minimize the cost [5]. In this regard, the encoding becomes more adaptable to edges and other particularities of the input pictures. Concerning intra prediction, 35 different intra coding modes have been defined in the standard. In this prediction mode, PUs inherit the same size (2N×2N) as the CU to which they belong. An exception to this rule occurs when the CU cannot be split any further, in which case it is allowed to be divided into four smaller N×N PUs. As a result, PUs can range from 64×64 to 4×4 pixels for intra prediction. With regard to inter prediction, the standard allows the usage of symmetric PU partitions for all CU sizes, except for N×N if the resulting PUs are 4×4. It also restricts the asymmetric partitions to those CUs whose size is larger than 8×8 pixels. Certain other restrictions apply for intra and inter prediction in very specific cases [6].

The two prediction modes enable the estimation of the picture information from temporal and spatial neighbors. As a result, some residual values are generated, which are transmitted to the decoder to reconstruct the original picture. This residual needs to be transformed using a tree structure into what is known as a residual quadtree (RQT). In this way, TU blocks can range from 32×32 to 4×4 samples. As can be seen, the standard makes a distinction between prediction and residual, so PUs and TUs may not necessarily match in a given CU, which provides extra flexibility in the encoding. Apart from all those mentioned above, other new features in HEVC include the advanced motion vector prediction (AMVP) to derive MV candidates from neighbors, and new in-loop filters such as the sample adaptive offset (SAO) filter.

The huge computational complexity of the coding tools introduced by HEVC was patent throughout the development of the standard. For this reason, HEVC defines two paral-

lelization schemes with the aim of reducing this complexity, namely tiles and wavefront parallel processing (WPP). The former consists of rectangular-shaped partitions named tiles, where dependencies are broken across boundaries, with the exception of the in-loop filters [7]. As a result, these partitions can be processed concurrently at the cost of some coding penalties. WPP, in turn, enables the creation of CTU rows that can also be processed in parallel [8]. In this case, entropy coding is allowed to cross boundaries to minimize coding losses, but at the expense of introducing a delay of at least two CTUs between consecutive rows. For this reason, this scheme can obtain better coding efficiency than tiles, albeit with less parallelism.

III. Related Work

Ever since the beginning of the development of HEVC, many works have attempted to address the huge computational complexity of this standard. A significant number of approaches found in the literature make use of parallel architectures to enable the concurrent encoding of video sequences. In fact, the standard itself includes some parallel tools to assist multiprocessing, namely tiles and WPP, as mentioned in the previous section. While this provides significant speed-ups, it also presents some issues, especially in terms of scalability and coding efficiency. Moreover, these techniques are not even enough to overcome the computational complexity of the encoder, so it remains necessary to develop fast encoding algorithms to reduce the processing time of the HEVC standard. With this aim, many works attempt to achieve this reduction by using techniques that make it possible to determine the best partitioning and coding structure without evaluating all the possibilities through a full RDO process.

The main way to achieve large time savings in HEVC is by determining the CTU/CU structure of the picture to be encoded, so that most of the blocks in the partitioning tree are omitted. In this way, L. Shen et al. proposed a fast CU size decision algorithm to reduce the number of candidate CUs to check in each quadtree [9]. This decision is taken according to adaptive thresholds based on the texture homogeneity of the picture. However, this method is only applicable to intra-predicted CUs, thus it is not adequate for inter-predicted frames, in which these CUs become a minority. For this reason, the same authors proposed a method that is valid for all types of prediction modes [10]. In this case, the decision is taken according to the neighboring CUs, but it becomes inadequate for highly complex sequences. Other proposals base their methods on statistics from the prediction residual, such as that proposed by H.L. Tan et al. [11]. Nevertheless, this involves evaluating several or all PUs in each depth level to determine whether to split a CU or not. Moreover, the order in which the quadtree is processed, the complexity of the picture, and the quantization parameter (QP) used in the encoding might have an impact on the speed-up obtained. Data mining techniques are also very often utilized to determine the CTU/CU splitting, such as the Bayesian rules proposed by X. Shen et al. [12]. However, the use of thresholds derived from off-line training can exhibit degraded R-D performance in some cases, such as in dynamic sequences or scene changes. For this reason, some authors have proposed on-line models, such as those of H.-S. Kim and R.-H. Park, which can be effective but extremely complex [13].

A different approach to reducing the computational complexity of the encoder consists in selecting the best PU mode, and omitting the rest. Many works achieve this by conditionally evaluating particular sets of partitions and modes depending on prior coding decisions. In fact, many of them focus on detecting the skip mode at an early stage. In this way, H. Lee et al. proposed an early skip mode decision based on the use of thresholds and the R-D cost of the 2N$\times$2N merge mode [14]. However, even if the thresholds are dynamically recalculated, they are highly dependent on the complexity of the video sequence. Alternative methods include the use of down-sampled pictures to analyze the texture of the original image, such as that proposed by G. Tian and S. Goto [15], or the use of the Sobel operator to detect the edges in a region and skip some evaluations, as proposed by X. Huang et al. [16]. Nevertheless, these two methods also make use of thresholds that need to be established beforehand.

All the aforementioned methods are not mutually exclusive. In fact, CTU/CU splitting algorithms and PU selection methods can be combined in such a way that only certain prediction modes of some depth levels are evaluated. In this way, higher speed-ups are obtained compared with the two methods separately. As has been mentioned before, one technique to predict either the best CU partitioning or the prediction mode is to infer the decision from the neighbor information. This is suggested by J.-H. Lee et al. in their work, in which the CTU/CU splitting algorithm is based on the R-D costs of the parent and current levels, and the PU selection is performed according to spatio-temporal and depth correlation values [17]. Similarly, S. Ahn et al. make use of parameters that are intrinsically calculated during the encoding process, such as filtering parameters, motion vectors (MVs) or TU sizes, among others [18]. However, the quadtree is still evaluated recursively from top to bottom to execute early termination algorithms, so it may not obtain large speed-ups in highly-detailed sequences or when using low QPs. The same occurs with the approach followed by J. Lu et al., in which the splitting decision is taken according to the homogeneity of the pixels with fixed thresholds [19]. J. Xiong et al., in turn, make use of a two-layered ME algorithm to base the splitting decision on the distortion costs of the child CUs and a modified version of the R-D model [20].

Regarding pre-analysis algorithms, the most relevant example is the look-ahead module of x264 [21], [22], which is an open-source H.264/MPEG-4 AVC encoder. This module operates with a subsampled version of the input frames, dividing them into fixed-sized blocks to calculate their intra/inter costs. This information is used at a later stage, mainly to establish the pattern of the output stream in conjunction with the rate-control algorithm. Nevertheless, this technique is aimed at H.264/MPEG-4 AVC, which defines a much simpler partitioning scheme than HEVC. For this reason, even though it has been ported to the newer standard in x265 [23], it does not fully meet its requirements. Other uses for the pre-

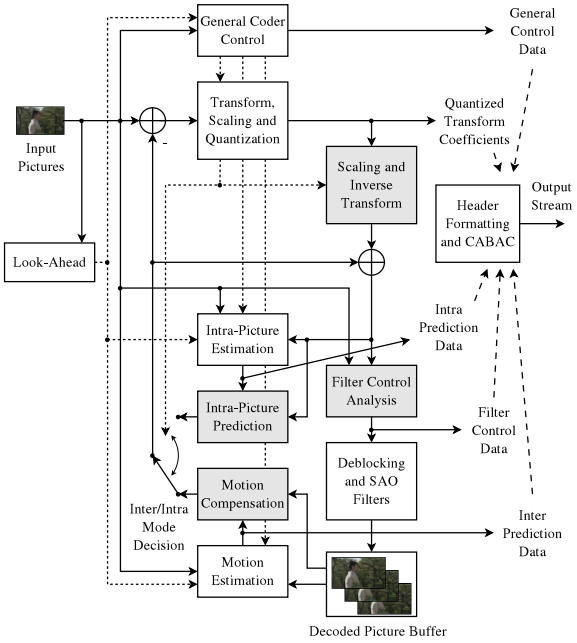


Fig. 3. Typical HEVC encoder architecture with a look-ahead stage.

analysis algorithm include, for example, the adjustment of the rate-control algorithm, as proposed by L. Sun et al. [24]. These applications of the pre-analysis algorithm, although not covered in this paper, are complementary to the proposed algorithm.

IV. CTU/CU Partitioning Algorithm Based on a Look-Ahead Stage

A look-ahead is an operation that is performed after the input pictures have been read or received, and before the actual encoding is performed, with the aim of obtaining valuable information from the images. This information is later used by the encoder for different purposes, e.g. rate control, bit allocation, scene change detection, or, as is the case for the proposed algorithm, acceleration of the encoding process. Consequently, it can be understood as a pre-analysis algorithm. Figure 3 shows the architecture of a typical HEVC-compliant encoder in which a look-ahead stage has been included. Blocks shaded in gray represent the processing units that both the encoder and the decoder have in common. As can be seen, the input pictures are analyzed by this stage before any encoding is performed, so the partitioning and neighbor information is not available for prediction. In spite of this, there is a large amount of information that can be extracted with the look-ahead algorithm, including motion information, intra modes, prediction costs, coding patterns, partitioning, etc. This information is later used in the many modules that constitute the encoder to guide the encoding process.

In prior works, we focused our efforts on implementing a look-ahead algorithm for accelerating both the inter and intra prediction modules of the encoder. Firstly, the information

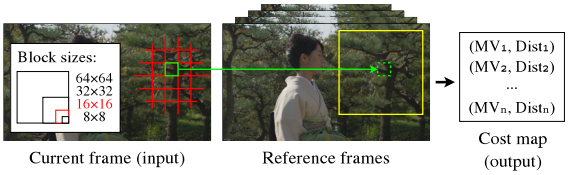


Fig. 4. Diagram of the ME performed by the look-ahead stage.

obtained in the look-ahead stage was used to reduce the number of positions checked in the search area, and to limit the number of reference frames [25], [26]. As a result, the total encoding time was notably reduced at the expense of negligible losses in terms of coding efficiency. However, this algorithm was only valid for those configurations that make use of inter prediction, but not for All Intra configurations. For this reason, a complementary algorithm was proposed in [27], in which the look-ahead stage is extended to estimate intra prediction costs as well. In the present paper, however, the perspective is different. Instead of accelerating the prediction modes, the proposed algorithm is aimed at accelerating the CTU/CU partitioning algorithm in order to obtain larger speed-ups.

The proposed scheme can be divided into two steps. First, a look-ahead stage performs a preliminary ME and obtains distortion costs from the input frames. Then, the proposed CTU/CU partitioning algorithm builds an optimal quadtree according to the look-ahead costs. As a result, a large number of partitions can be omitted, resulting in shorter computation times. Both steps will be described in detail in the following subsections.

A. Look-Ahead Stage

As mentioned before, a look-ahead is an operation performed before the actual encoding to obtain preliminary information that is later used for different purposes. In the case of the proposed algorithm, it is responsible for calculating the motion information of the picture to be processed. For this reason, it is possible to find certain similarities between this stage and the inter prediction module of the encoder, as they both perform a similar operation. Nevertheless, the former is intended to provide just an estimate of the motion in a frame, so its design can be much simpler, and it also has a different purpose, which is to collect the information that is later used to accelerate the encoding.

The ME performed by the look-ahead follows a block-based approach in a similar way to the ME module of the encoder. However, evaluating all the possible partitions defined in HEVC would be excessively time-consuming. For this reason, and given that only an estimate of the costs is needed, the look-ahead performs this estimation by using square blocks. In order to later predict a more accurate partitioning quadtree, and with the aim of adapting it to the picture in a more flexible way, the following block sizes are used: 8×8, 16×16, 32×32, and 64×64. Thus, the picture is divided into four different grids. Once the frame is split, the look-ahead performs an ME on every reference frame, as depicted in Fig. 4. Therefore, the

result of this stage is a data structure that stores the distortion value in terms of the sum of absolute differences (SAD) of each block and reference frame. This information is later used by the encoder to build an optimal partitioning quadtree and to accelerate the encoding process.

With the aim of obtaining the most reliable cost map possible, some considerations were taken into account in the design of the look-ahead stage:

- It uses the same ME algorithm as the encoding stage. This makes it possible to obtain similar MVs, and therefore similar distortion costs. Nevertheless, as the CTU/CU partitioning algorithm only requires an estimate of the costs, the precision of the search is limited to integer-pel samples instead of performing the full quarter-pel search, which results in a faster estimation.
- The MVs of the blocks adjacent to the current block are considered as predictors in a similar way to the AMVP feature of the HEVC standard [6]. In this way, two predictors are retrieved: the one above the current block and the one to the left. If either of these is not available, then a temporal candidate is interpolated and used as predictor. The list of predictors is filled with zero MVs, $(0, 0)$, until it contains a total of two predictors. At this point, both options are evaluated, and the one providing the lowest distortion is chosen as the predictor for the current block.
- The following Lagrangian multiplier R-D optimization model is used:

$$J\left(MV\right) = Dist\left(MV\right) + \lambda \cdot Rate\left(MV\right), \quad (1)$$

where J represents the cost function for a given MV, $Dist$ the distortion function in terms of SAD, λ is the Lagrangian multiplier, and $Rate$ represents the function to calculate the bits required to code the MV. It should be noted that the rate is dependent on the previously selected predictor.

B. CTU/CU Partitioning Algorithm

As mentioned in Section II, the new partitioning scheme introduced by HEVC represents the encoding tool that contributes most to the high coding efficiency of this standard. Structures such as CUs, PUs and TUs make it possible to adapt the encoding to the features of the video sequence in a flexible way. In this regard, most encoders perform an exhaustive search through the partitioning tree to determine the optimal partitioning in terms of R-D. However, this requires evaluating a large number of possibilities, given that every CU can be subsequently divided into four CUs, thus resulting in long computation times. For this reason, the main aim of the proposed method is to obtain a preliminary estimate of the partitioning quadtree prior to the encoding, so that the number of combinations that need to be evaluated is considerably smaller.

The proposed method is based on the idea that the distortion costs obtained in the look-ahead can be used to calculate an estimate of the partitioning for P and B slices. In this regard, a bottom-up approach is followed to build the corresponding quadtrees from the look-ahead information, so that the decision of splitting or not splitting a CU is determined according to the preliminary costs of smaller CUs. Beginning from the leaves at depth $D-1$ to the top at depth 0, where $D \in \{1, 2, 3, 4\}$ represents the maximum partitioning depth allowed by the encoder, costs are recursively grouped into fours. If the sum of the costs estimated for each sub-CU is lower than the cost of a given CU, then the CU will be split, as this provides better coding efficiency according to the look-ahead algorithm.

Firstly, the proposed bottom-up approach requires the definition of costs to support the splitting decision. In this regard, a regular encoder bases this decision not only on the distortion cost, but also on the rate, as shown in Eq. 1. In the case of the proposed algorithm, it would be possible to predict the rate from the look-ahead MVs and the estimated predictors, but this, however, would not produce accurate results. The main reason for this lies in the fact that it is not possible to anticipate the structure of the CUs and PUs that will finally be used in the encoding, thus it is not possible to know which predictors will be available for a certain PU. As a way to solve this, instead of just omitting the rate component, a fixed rate value is used for the proposed partitioning algorithm. Considering that a CU can be identified within a CTU as $CU_{d,i}$, where $d \in \{0..D-1\}$ is the depth of the CU, and $i \in \left\{0..4^d - 1\right\}$ is the index of the CU in Z-order for depth d, its associated look-ahead cost $C_{d,i}$ can be defined as:

$$C_{d,i} = Dist\left(MV_{d,i}\right) + \lambda \cdot Rate_{LA}, \quad (2)$$

where $MV_{d,i}$ is the MV of the co-located look-ahead block for $CU_{d,i}$, $Dist$ corresponds to the distortion function in terms of SAD, λ is the Lagrangian multiplier, and $Rate_{LA}$ represents the fixed rate component introduced. As can be seen, the calculation of $C_{d,i}$ is based on Eq. 1. The multiplying factor λ enables the adaptation of the rate component to the QP, so that high QP values lead to a higher weight on the number of bits required to encode a CU, and therefore to a lower fragmentation. Accordingly, $Rate_{LA}$, which has been estimated experimentally as the average difference between MVs and predictors, prevents the over-splitting of quadtrees.

Once the procedure to obtain the cost associated with a CU has been described, it is possible to define how the aforementioned bottom-up algorithm estimates the partitioning of a CTU. Starting from the smallest CUs at depth $D-1$ to the largest CU at depth 0, the following criterion is followed:

$$S\left(CU_{d,i}\right) = \begin{cases} \text{split}, & \text{if } \sum_{j=0}^{3} C^{\star}_{d+1,4i+j} < C_{d,i} \\ & \text{and } d < D-1 \\ \text{no split}, & \text{otherwise} \end{cases} \quad (3)$$

where S represents the partitioning decision, and $C^{\star}$ is the cost assigned to a CU once the splitting decision has been taken. This new cost is defined as follows:

$$C^{\star}_{d,i} = \begin{cases} \min\left\{\sum_{j=0}^{3} C^{\star}_{d+1,4i+j}, C_{d,i}\right\}, & \text{if } d < D-1 \\ C_{d,i}, & \text{otherwise} \end{cases} \quad (4)$$

The bottom-up algorithm described above, and defined by Eq. 3 and 4, enables the construction of the quadtree on the basis of the look-ahead costs. This method is depicted in

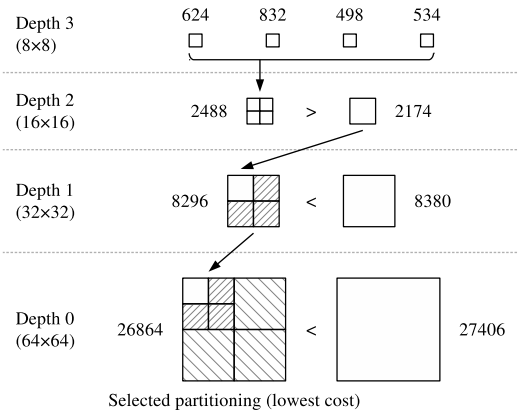


Fig. 5. Estimation of the CTU/CU partitioning quadtree following a bottom-up approach.

Fig. 5, which shows the estimation of the CTU/CU partitioning by grouping costs from the smallest blocks to the largest one. The numbers shown in the figure are examples of $C^{\star}$ and C costs for split and non-split blocks, respectively. Blocks shaded in gray represent CUs whose partitioning has been determined by following this procedure, but these become irrelevant for the understanding of the figure. As can be seen, the main aim of the algorithm is to find the partitioning quadtree that minimizes the total cost.

The result of the proposed algorithm is optimal from the point of view of the look-ahead costs. Nevertheless, the quadtree obtained might not necessarily match the one that the encoder would have selected after a full search through all the possible combinations. This implies that some coding inefficiencies might appear due to the increase in the number of bits needed to encode the CTUs. There are two main reasons for this mismatch. First, as discussed above, the predictors calculated in the look-ahead are used in the ME of this preliminary stage to avoid drifted MVs, but given that it is not possible to know which of these predictors will finally be available for a given CU, they cannot be used to calculate the rate component of the cost function. Secondly, P and B slices can contain intra-predicted PUs. However, the percentage of inter-predicted PUs is significantly greater than the intra-predicted ones in these slices, so it is possible to assume that it is feasible to estimate the partitioning from just the inter prediction costs.

Once the partitioning quadtree has been estimated, the encoder could omit the evaluation of all the CUs not included in the tree. Nevertheless, even though this would involve a large reduction in terms of encoding time, it would also lead to some coding efficiency losses caused by estimation misses. For this reason, a safer approach is followed by the proposed algorithm to prevent the encoder from skipping relevant partition combinations. In particular, the encoder checks not only the PUs of the CUs placed at the leaves of the quadtree, but also some additional PUs at upper and lower depth levels. As a result, the encoder is able to achieve better R-D values with low impact in terms of time reduction. In this regard, a study of different combinations will be shown in Section V, performing a thorough evaluation of the influence of different CU/PU configurations on the results.

V. CU/PU Configuration Selection and Experimental Evaluation

In this section, different CU/PU configurations will be evaluated to analyze the influence of the inclusion and exclusion of PUs in different depth levels. This implies that the encoder might not necessarily commit to the partitioning quadtree provided by the look-ahead. In this regard, a study comprising several configurations will be performed to analyze their resulting coding efficiency and time reduction. However, given the large number of possible configurations that may arise from the wide range of different PUs defined in HEVC, the study will be performed following a two-step approach. First, considering $CU_{d,i}$ as a leaf of the estimated quadtree, i.e. a non-split CU according to the look-ahead, several configurations will be analyzed varying which PUs are evaluated or omitted at the parent CU, i.e. $CU_{d-1,\lfloor i/4 \rfloor}$. A statistical analysis of the results will provide a subset of these, which will be used as a basis for the second stage. In this one, the same process of analysis will be repeated but for the corresponding child CUs, i.e. $CU_{d+1,4i+j}, j \in \{0,1,2,3\}$. The details of the study will be explained in greater detail in the following subsections.

The experiments were executed on the HEVC Test Model (HM-16.6) reference software [28], following the guidelines and coding conditions provided by the document elaborated by the JCT-VC [29]. In this regard, Random Access was the selected configuration, and the QP values tested were 22, 27, 32 and 37. The encodings were carried out using the Main profile and 8-bit depth. Sequences of classes A to D according to the JCT-VC classification were used, which include the following resolutions: 2560×1600 (A), 1920×1080 (B), 832×480 (C), and 416×240 (D). A search area of 128×128 pixels was used for the look-ahead stage, which is equivalent to the default search area for the baseline encoder.

The hardware platform used in the experiments is composed of an Intel® Xeon® E5-2630L v3 CPU running at 1.80 GHz. The encoder was compiled with GCC 4.8.5-4 and executed on CentOS 7 (Linux 3.10.0-327). Turbo Boost was disabled to achieve the reproducibility of the results.

The results are given in terms of time reduction (TR) and the Bjøntegaard delta rate (BD-rate) [30]. The BD-rate is a measure of coding efficiency that represents the percentage of bitrate variation between two sequences with the same objective quality. Higher time reduction and lower BD-rate values represent better results. Accordingly, the two variables will need to be evaluated for each CU/PU configuration.

With regard to intra prediction, the encoder used in the tests implemented an optimized algorithm that omits the evaluation of this type of prediction when the best inter prediction mode does not produce non-zero transform coefficients for a given CU [31]. This results in a shorter time devoted to intra prediction in P and B slices, while maintaining the coding

TABLE I
PU CONFIGURATIONS EVALUATED IN THE FIRST STAGE

	Parent CU				Selected ($CU_{d,i}$)				Child CUs			
	□	◫	◧	▨	□	◫	◧	▨	□	◫	◧	▨
1					●	●	●	●				
2	●				●	●	●	●				
3	●			●	●	●	●	●				
4	●	●		●	●	●	●	●				
5		●		●	●	●	●	●				
6		●	●	●	●	●	●	●				
7	●	●	●	●	●	●	●	●				
8	○				●	●	●	●				
9	○			○	●	●	●	●				
10	○	○		○	●	●	●	●				
11		○		○	●	●	●	●				
12		○	○	○	●	●	●	●				
13	○	○	○	○	●	●	●	●				

□ 2N×2N and skip
◫ 2N×N, N×2N and N×N
◧ Asymmetric PU sizes
▨ Intra prediction

● This prediction mode is tested.
○ This prediction mode is tested if all four child CUs are not split according to the proposed algorithm.

efficiency. Additionally, with regard to AMP, the encoder also omits the evaluation of certain asymmetric PUs on the basis of previous evaluations [31]. These two algorithms are enabled by default, and have been kept as is for the evaluation. This will be taken into consideration when interpreting the results.

A. Evaluation of PUs at the Parent CU

The first set of tests is intended to evaluate the influence of the PUs located at $CU_{d-1,\lfloor i/4 \rfloor}$, i.e. the parent CU. It should be noted that the parent CU might not be available if the proposed method decided to stop splitting at depth 0, or if it is out of bounds. In order to explore all the possibilities, a wide variety of configurations was chosen for evaluation, as detailed in Table I. Although some other marginal configurations are possible, the ones selected for this evaluation cover a large number of study cases, and make it possible to analyze the effect of different PU combinations at the parent CU. Different PU sizes were evaluated, including symmetric and asymmetric modes, as well as the particular cases in which none or all of the PUs are evaluated for this CU. Additionally, the following aspects were taken into account in their selection:

- All the different PUs at $CU_{d,i}$ are always checked, as it is the CU that provided the lowest cost according to the proposed method. Given that the estimated quadtree results in the lowest estimated cost, it is assumed that these PUs might also provide the lowest cost in the encoding process.
- All the PUs of the child CUs are omitted in this first stage of the evaluation. They will be evaluated once it is determined which configurations provide the best trade-off when using the PUs of the parent CU.
- PUs at the parent CU are significantly larger than those at $CU_{d,i}$. Accordingly, their evaluation involves longer computation times. For this reason, some of the proposed PU configurations were provided with a fast coding decision algorithm that enables the omitting of the evaluation of PUs at the parent CU in certain cases. In particular, these PUs will be taken into consideration only when the four child CUs are not split according to the look-ahead. In this event, it is assumed that the partitioning quadtree is simple enough, and that the parent CU may provide better R-D results if not split. Otherwise, it is omitted. Configurations with this algorithm enabled are shown in the table with empty dot symbols.

TABLE II
CODING EFFICIENCY AND TIME RESULTS OF EACH PU CONFIGURATION EVALUATED IN THE FIRST STAGE

	BD-rate (%)	TR (%)
1	8.73	72.10
2	6.33	64.76
3	5.39	62.73
4	4.60	56.33
5	5.40	62.93
6	4.74	57.70
7	4.11	51.38
8	6.81	68.81
9	5.90	68.04
10	5.21	65.18
11	6.11	68.10
12	5.55	65.31
13	4.83	62.61

The results obtained by this set of selected configurations are shown in Table II. In the interests of brevity, the average coding efficiency and TR for all the sequences and QPs is shown instead of the per-sequence results. First of all, it is worth mentioning the results obtained by Configuration 1, in which the PUs at the parent CU are not evaluated. It can be seen that roughly three-quarters of the encoding time can be saved, which represents a notable reduction, but it comes at the expense of high coding efficiency penalties. For this reason, it becomes necessary to evaluate some PUs at this CU with the aim of avoiding coding efficiency losses when the proposed method misses the prediction and decides to split the CU when it should not. In this respect, Configuration 7 deserves special attention, as it involves evaluating all the PUs at both the parent CU and at $CU_{d,i}$. In this case, the BD-rate can be reduced by more than half compared with Configuration 1, but in turn the TR drops to 51.38%. These time values may be directly linked to the fact that three and two depth levels out of a total of four are fully omitted for Configurations 1 and 7, respectively.

With regard to the other configurations, in particular those whose indices range from 2 to 6, it can be seen that they obtain values lying between those obtained by Configurations 1 and 7. This can be easily explained by the fact that they evaluate more partitions than Configuration 1, but fewer than Configuration 7. At this point, it is worth mentioning Configurations 2 and 3, which only differ in the inclusion of the intra prediction mode in the parent CU. As can be seen, the BD-rate is significantly reduced at the negligible cost of approximately 2% in terms of time reduction. This not only shows the benefits of the

evaluation of the intra mode, but also demonstrates that the encoder can still decide to choose encoding a CU using intra prediction even though the proposed algorithm operates with inter prediction costs.

Finally, Configurations 8–13 are closely related to Configurations 2–7, as they evaluate the same PUs, but using the fast decision algorithm described previously. In this way, they suffer from a small penalization in terms of coding efficiency, but they achieve a notable reduction in the encoding time, obtaining values that range from 62.61% to 68.81% on average.

Each of the configurations described above represents a trade-off between coding efficiency and time savings. In those cases in which more PUs are evaluated, the coding efficiency is improved, but there is also an impact on the encoding time. For this reason, it becomes very complex to decide whether one configuration is better than another, mainly because there are two variables of interest. In fact, in some real-world scenarios the coding efficiency might be of more relevance, whereas faster encoding might be a more important requirement in other situations. As a way to solve this, a statistical analysis can help find a subset of configurations from Table I that provides a good balance between the two variables. In this regard, it is necessary to first characterize the dataset of the problem, which in this particular case can be categorized as non-parametric. Taking this into consideration, the Friedman test seems the most suitable option for this study, as it is a non-parametric test used for repeated measures [32], [33]. This test enables the comparison of more than two methods given a set of problems for a particular output. In fact, the null hypothesis corresponds to the case in which all the methods are equivalent. In our case, the methods are the different CU/PU configurations, the problems correspond to the test sequences, and the outputs are the coding efficiency and the speed-up. The speed-up is a time measure equivalent to the TR, but it increases linearly, so it is more appropriate for the statistical test. It can be converted using the following equivalence:

$$\text{Speed-Up} = 1 - \frac{1}{\text{TR}} \tag{5}$$

The Friedman test can be performed with respect to one output variable. If the test is performed with respect to the coding efficiency, the p-value obtained is 1.9134e-29, and if it is calculated for the speed-up, the p-value is 2.5013e-31. Considering a significance level of $p < 0.05$, we can confirm that the null hypothesis is rejected, which means that the configurations are not statistically equivalent. As the null hypothesis of the Friedman test is rejected, we can proceed with a post-hoc test; in particular, with Holm's procedure [33], [34]. In this test, all the methods are compared against the one that obtained the best mean rank given the target output variable, which is also called the control method. To do this, it sorts all the p-values in increasing order and performs a correction. Then, the corresponding null hypotheses are rejected until there is one method whose hypothesis is accepted, in which case the algorithm stops. In this way, those methods whose null hypothesis is rejected are assumed to be significantly different to the control method.

TABLE III
P-VALUES OBTAINED USING HOLM'S PROCEDURE IN THE FIRST STAGE

	BD-rate		Speed-up	
	P-value	Rank	P-value	Rank
1	3.8562e-16	13.00	**(control)**	1.00
2	7.2005e-10	10.27	5.9874e-04	6.53
3	1.3922e-04	7.07	1.4782e-07	9.00
4	**3.3285e-01**	3.27	1.1346e-13	12.00
5	1.4706e-04	7.00	3.7683e-07	8.73
6	**3.3285e-01**	2.87	2.0344e-11	11.00
7	**(control)**	1.00	3.8562e-16	13.00
8	2.3607e-13	11.87	**4.8192e-01**	2.00
9	7.3629e-09	9.73	**2.2451e-01**	3.53
10	3.4922e-03	5.73	2.1901e-03	6.00
11	1.1761e-08	9.60	**2.2451e-01**	3.47
12	4.9895e-04	6.53	6.7347e-03	5.47
13	**3.3285e-01**	3.07	5.5158e-08	9.27

The ranking obtained by the configurations studied in this stage is shown in Table III, in which those configurations whose p-value is higher than the significance level $p < 0.05$ are listed in bold. As can be seen, if Holm's procedure is performed considering the BD-rate as the target output, then Configuration 7 is selected as the control method as it is the one providing the best coding efficiency, and Configurations 4, 6 and 13 are kept as significantly equal. On the contrary, if this test is carried out considering the speed-up as the target output, Configuration 1 is selected as the control method this time, and Configurations 8, 9 and 11 are selected as significantly equal using the same significance level as before.

As can be seen, depending on what the variable of interest is, different configurations are selected. If any of the configurations had been selected for both the speed-up and the BD-rate, these would have been considered to be the ones providing the best trade-off between both target variables, but this does not occur in this case. As a way to solve this, it is possible to repeat the statistical tests on the two subsets, but using the target variable not considered for the first test. In other words, the speed-up would be the target variable for Configurations 4, 6, 7 and 13, and the BD-rate would be the target variable for Configurations 1, 8, 9 and 11. In this case, as there is a small number of configurations per subset, the pairwise Wilcoxon signed-rank test can be used instead of the Friedman and Holm post-hoc combination to evaluate which configurations provide the best trade-offs [33], [35].

With regard to the first subset, the test shows that the four configurations are significantly different to each other. For this reason, provided that they are all equivalent in terms of BD-rate, it is possible to directly take the one providing the largest speed-up, which corresponds to Configuration 13. Concerning the second subset, the Wilcoxon test shows that Configurations 9 and 11 are significantly similar, reporting a p-value of 6.3867e-1, while the other two are different. It is possible to assume, then, that Configurations 9 and 11 provide an equivalent trade-off between the two target variables. Nevertheless, given the long processing time required for each configuration, and that this will grow exponentially in the next

stage of the study, only one of the two configurations will be selected. Given that Configuration 9 obtains slightly better BD-rate results, it was selected as baseline for the second stage of the analysis. In any case, as they are statistically equivalent, the results of the study would be comparable to those obtained if Configuration 11 were chosen instead.

As a conclusion, this stage has shown the evaluation of different PUs of the parent CU when the proposed algorithm decides to stop splitting at $CU_{d,i}$. The average results show that the speed-up values obtained range from 50% to 75% depending on the proportion of PUs checked at the parent CU. The BD-rate varies from 4.11% to 8.73%, becoming lower as the number of PUs evaluated increases. As a result, selecting a configuration that offers a good trade-off between the two variables can be complex and subjective. For this reason, a statistical test was performed to determine which configurations provide the best trade-off between the two variables from an objective point of view. This test showed that Configuration 13 provides a good balance between coding efficiency and speed-up if the former variable is given greater priority, whereas Configurations 9 and 11 are the ones selected by the test if the speed-up is of greater relevance, although only Configuration 9 will be used in the next stage. Consequently, these two configurations will be used as baseline for the next stage of the algorithm evaluation, in which it will be analyzed how different PUs of the child CUs affect the results in terms of both target variables.

B. Evaluation of PUs at Child CUs

Once it has been analyzed how different PUs at the parent CU affect the encoding in terms of coding efficiency and time reduction, the tests of this second stage will evaluate how these two variables are influenced by the PUs located at the four child CUs. The evaluation of these PUs is only applicable when the depth at which the proposed algorithm decided to stop splitting is not $D-1$. Taking this into account, Table IV shows the full set of configurations evaluated in the second stage. In the interest of clarity, it also contains the two configurations selected in the first stage, on which the rest of the configurations of this stage are based. As can be seen, a large subset of configurations has been selected to cover a wide variety of study cases. On this occasion, in addition to those of the first stage, the following aspects were taken into account for the selection of the configurations:

- In HEVC, the coded block flag (CBF) signals the significance of the entire TU, i.e. it signals whether a TU contains non-zero transform coefficients [36]. Accordingly, it is possible to assume that there is a direct relationship between the splitting decision and the CBF. In fact, this flag has already been utilized in the past to prune the partitioning tree [37]. For this reason, in a similar way to the first stage, some configurations have been provided with a fast encoding algorithm that omits the evaluation of PUs at the child CUs when the best partitioning at $CU_{d,i}$ does not trigger the flag, i.e. when the best PU does not result in non-zero coefficients.
- Configurations x.4 and x.9 differ from x.3 and x.8, respectively, in the fact that the former only evaluate the PUs marked with a circle if the child CUs are located at depth $D-1$. The main reason for the evaluation of these specific configurations is that it may improve the coding efficiency of sequences with a lot of detail and low QP, as the picture tends to be partitioned into smaller blocks.

TABLE IV
PU CONFIGURATIONS EVALUATED IN THE SECOND STAGE

	Parent CU				Selected ($CU_{d,i}$)				Child CUs			
	□	◫	◧	▨	□	◫	◧	▨	□	◫	◧	▨
9	○			○	●	●	●	●				
9.1	○			○	●	●	●	●	●			
9.2	○			○	●	●	●	●	●			●
9.3	○			○	●	●	●	●	●	●		●
9.4	○			○	●	●	●	●	●	◉		●
9.5	○			○	●	●	●	●	●	●	●	●
9.6	○			○	●	●	●	●	◇			
9.7	○			○	●	●	●	●	◇			◇
9.8	○			○	●	●	●	●	◇	◇		◇
9.9	○			○	●	●	●	●	◇	◎		◇
9.10	○			○	●	●	●	●	◇	◇	◇	◇
13	○	○	○	○	●	●	●	●				
13.1	○	○	○	○	●	●	●	●	●			
13.2	○	○	○	○	●	●	●	●	●			●
13.3	○	○	○	○	●	●	●	●	●	●		●
13.4	○	○	○	○	●	●	●	●	●	◉		●
13.5	○	○	○	○	●	●	●	●	●	●	●	●
13.6	○	○	○	○	●	●	●	●	◇			
13.7	○	○	○	○	●	●	●	●	◇			◇
13.8	○	○	○	○	●	●	●	●	◇	◇		◇
13.9	○	○	○	○	●	●	●	●	◇	◎		◇
13.10	○	○	○	○	●	●	●	●	◇	◇	◇	◇

□ 2N×2N and skip
◫ 2N×N, N×2N and N×N
◧ Asymmetric PU sizes
▨ Intra prediction

● This prediction mode is tested.
○ This prediction mode is tested if all four child CUs are not split according to the proposed algorithm.
◇ This prediction mode is tested if CBF was set in the parent CU.
◯ Only when $d+1=D-1$.

The results obtained by this second set of selected configurations are shown in Table V. As in the first stage, the average coding efficiency and TR for all the sequences and QPs is shown instead of the per-sequence results. As can be seen, introducing the evaluation of the 2N×2N PU and the Skip mode in the child CUs in Configurations x.1 involves a sharp reduction in the BD-rate, at the cost of a certain increase in the encoding time. These two modes become particularly important when the proposed algorithm decides not to split a CU when it should be split. This misprediction usually occurs when the costs of splitting and not splitting are very similar in the look-ahead. After that, Configurations x.2 show the influence of the intra prediction mode in child CUs. It can be seen that the coding efficiency is notably improved at the cost of a negligible increase in the encoding time. Additionally, this enables the intra prediction mode to be evaluated in these CUs, even if the proposed algorithm operates with inter prediction costs.

The rest of the configurations show that the BD-rate decreases as more partitions are evaluated. In this way, for example, 40.68% of the encoding time can be saved at the

TABLE V
CODING EFFICIENCY AND TIME RESULTS OF EACH PU CONFIGURATION EVALUATED IN THE SECOND STAGE

	BD-rate (%)	TR (%)
9	5.90	68.04
9.1	4.16	59.99
9.2	3.68	58.72
9.3	2.56	47.67
9.4	3.34	55.83
9.5	2.23	46.05
9.6	4.64	65.44
9.7	4.20	64.34
9.8	3.35	62.14
9.9	3.90	63.30
9.10	3.11	61.41
13	4.83	62.61
13.1	3.30	54.56
13.2	2.86	53.28
13.3	1.71	42.24
13.4	2.51	50.41
13.5	1.40	40.68
13.6	3.67	60.04
13.7	3.25	58.92
13.8	2.38	56.80
13.9	2.96	57.83
13.10	2.15	55.98

TABLE VI
P-VALUES OBTAINED USING HOLM'S PROCEDURE IN THE SECOND STAGE

	BD-rate		Speed-up	
	P-value	Rank	P-value	Rank
9	1.7329e-17	22.00	**(control)**	1.00
9.1	8.4667e-12	18.13	2.2941e-03	9.60
9.2	3.7695e-08	15.13	1.2694e-04	11.40
9.3	**5.3569e-02**	7.20	1.3493e-12	18.73
9.4	2.7818e-05	12.20	1.8918e-07	14.47
9.5	**3.0884e-01**	4.87	4.2064e-14	19.80
9.6	2.1856e-15	20.67	**7.6685e-01**	2.00
9.7	1.3493e-12	18.73	**7.6685e-01**	3.07
9.8	3.4733e-06	13.20	**2.2954e-01**	5.73
9.9	2.9127e-10	16.93	**5.0490e-01**	4.27
9.10	1.9627e-04	11.07	**6.3063e-02**	7.07
13	8.2665e-14	19.60	**2.7047e-01**	5.33
13.1	9.7884e-05	11.53	6.4897e-09	15.80
13.2	2.0157e-02	8.07	2.4007e-10	17.00
13.3	**6.3267e-01**	2.13	6.6341e-16	21.00
13.4	**2.4527e-01**	5.53	2.9798e-12	18.47
13.5	**(control)**	1.00	1.7329e-17	22.00
13.6	4.9558e-08	15.00	1.1468e-02	8.47
13.7	1.3126e-04	11.33	1.6599e-03	9.87
13.8	**2.4527e-01**	5.67	8.9778e-06	12.73
13.9	4.3491e-03	9.20	1.6945e-04	11.20
13.10	**4.7530e-01**	3.80	5.4475e-07	14.00

cost of only 1.40% in terms of BD-rate for Configuration 13.5. Particular attention should be paid to Configurations x.4, whose coding efficiency and TR results are halfway between those of Configurations x.2 and x.3, as they evaluate more partitions than the former, but fewer than the latter.

Finally, the results obtained by Configurations x.6–10 show the effect of the fast encoding algorithm proposed for this second evaluation stage. As could be expected, the BD-rate increases compared with the one obtained by Configurations x.1–5, but the time savings derived from the use of the CBF are notably higher. In fact, it is worth noting how the coding efficiency obtained by Configurations x.6 is closer to that of Configurations x.1 than to the baseline, whereas the opposite happens with the TR.

At this point, the 22 configurations offer different trade-offs between coding efficiency and encoding time; those configurations providing higher time savings also result in higher coding efficiency losses, and vice versa. For this reason, it is necessary to determine which of these represents the best trade-off and provides a good balance between the two variables, as in the first stage of the study. In this regard, we can proceed in the same way: performing a statistical analysis with the Friedman test. Once again, the methods of this test are the different PU configurations, the problems correspond to the test sequences, and the outputs are the coding efficiency and the speed-up. If the test is performed with respect to the coding efficiency, the p-value obtained is 1.2579e-49, and if it is calculated for the speed-up, the p-value is 7.2310e-53. Considering a significance level of $p < 0.05$, we can see that the null hypothesis is rejected, which means that the configurations are not statistically equivalent.

In view of the results, the next step is to perform Holm's procedure to select the configurations that are statistically equivalent to the control method with respect to a given output variable. In this regard, Table VI shows the output p-values and ranks of Holm's procedure, where those configurations whose p-value is higher than the significance level $p < 0.05$ are shown in bold. As can be seen, if the post-hoc procedure is performed when considering the BD-rate as the target variable, then Configuration 13.5 is selected as the control method, as it provides the lowest BD-rate, and Configurations 9.3, 9.5, 13.3, 13.4, 13.8 and 13.10 are kept as significantly equivalent. Conversely, if the post-hoc is performed with respect to the speed-up, then Configuration 9 is selected as the control method, given that it provides the highest speed-up in the subset, and Configurations 9.6, 9.7, 9.8, 9.9, 9.10 and 13 are maintained as significantly equivalent.

As in the first stage, the configurations chosen by Holm's procedure depend on the target variable and do not intersect, so it is not possible to choose just one configuration. For this reason, the statistical test is repeated, but evaluating the correlation between CU/PU configurations with respect to the opposite target variable in each subset. As there is a small number of configurations in each subset, the pairwise Wilcoxon signed-rank test is used instead of the Friedman test. With regard to the first subset, which was obtained using the BD-rate as target variable, the Wilcoxon test shows that the five configurations are statistically different from the point of view of the speed-up. Consequently, given that all of them were equivalent in terms of coding efficiency, it is possible to directly choose the one providing the best time reduction, which in this case is Configuration 13.8. The same procedure

can be applied to the second subset. In this case, the configuration providing the best BD-rate is Configuration 9.10, and the Wilcoxon test shows that no other configuration in the subset is statistically equivalent to it. Given that the five configurations in the subset are statistically equivalent with respect to the speed-up, it is possible to choose Configuration 9.10 as the one providing the best trade-off if priority is given to the speed-up.

As a conclusion, this second and last stage of the analysis has shown the effect of evaluating different PUs at the four child CUs on the basis of the configurations found in the first stage. The average results show that the speed-up obtained largely depends on the number of PUs evaluated, ranging from 40.68% to 68.04%. The BD-rate varies from a negligible 1.40% to 5.90%. In order to determine which configuration provides the best trade-off between the two variables, a statistical analysis was performed again with this new set of configurations. This test showed that Configuration 13.8 provides a good balance between coding efficiency and speed-up if the former variable is given greater priority, providing average time savings of 56.80% at the cost of only 2.38% in terms of BD-rate. Conversely, if the speed-up is given greater relevance, then Configuration 9.10 is chosen by the statistical test, which results in a 61.41% time reduction at the cost of 3.11% in terms of coding efficiency losses. Consequently, these two configurations are the ones that provide the best trade-offs between the two variables depending on which of them is given greater relevance, so that one or the other might be more appropriate depending on the scenario in which it is applied.

C. Analysis of the Selected Configurations

In the previous subsections, only the average coding efficiency and TR results for all the sequences and QPs are shown in the interests of brevity, although the different per-sequence results were used in the statistical tests. In this subsection, however, the full set of results will be detailed and analyzed for the two configurations selected. With the aim of simplifying the nomenclature, Configuration 13.8 will be referred to as the BD-rate-oriented configuration, and Configuration 9.10 as the speed-up-oriented configuration.

Firstly, Table VII shows the coding efficiency and TR results obtained by the BD-rate-oriented configuration. As can be seen, the results are not dependent on the test sequences. In other words, the design of the proposed algorithm enables its adaptation to sequences of different characteristics, sizes and textures, giving approximately the same results. This implies that high-resolution sequences can benefit from the proposed algorithm in the same way as those with lower resolutions. The coding efficiency obtained by the algorithm depends mainly on the hit rate achieved, i.e. the number of times the estimated quadtree matches the best partitioning of the picture. The time savings, in turn, are related to the number of partitions evaluated by the encoder in the partitioning process, which depends on whether the fast decision algorithms implemented for the PUs at the parent and child CUs decide to skip the partitions in these blocks or not. It might be worth noting the BD-rate obtained by the Kimono sequence, which is slightly higher than the average. This sequence contains motion that can be complex to predict, especially with the absence of predictors, as is the case in the look-ahead stage. Nevertheless, the time saving obtained is not influenced by this fact.

TABLE VII
PER-SEQUENCE RESULTS OF THE BD-RATE-ORIENTED CONFIGURATION

Class	Video sequence	BD-rate (%)	TR (%)
A	Traffic	1.75	67.35
	PeopleOnStreet	1.75	50.22
B	Kimono	4.11	58.55
	ParkScene	1.69	64.77
	Cactus	2.54	59.38
	BasketballDrive	2.53	56.39
	BQTerrace	2.92	65.87
C	BasketballDrill	2.22	53.96
	BQMall	2.58	60.39
	PartyScene	2.40	55.56
	RaceHorses	2.35	48.35
D	BasketballPass	2.06	49.43
	BQSquare	2.52	60.40
	BlowingBubbles	2.27	55.03
	RaceHorses	1.97	46.38
	Mean values	2.38	56.80

TABLE VIII
PER-SEQUENCE RESULTS OF THE SPEED-UP-ORIENTED CONFIGURATION

Class	Video sequence	BD-rate (%)	TR (%)
A	Traffic	2.58	69.96
	PeopleOnStreet	2.95	58.03
B	Kimono	4.99	62.96
	ParkScene	2.18	67.41
	Cactus	3.43	63.30
	BasketballDrive	3.57	60.68
	BQTerrace	3.30	67.28
C	BasketballDrill	3.05	58.59
	BQMall	3.51	64.68
	PartyScene	2.72	60.21
	RaceHorses	3.30	54.91
D	BasketballPass	2.88	56.44
	BQSquare	2.64	63.49
	BlowingBubbles	2.60	59.10
	RaceHorses	3.03	54.17
	Mean values	3.11	61.41

Similarly, Table VIII displays the results obtained by the speed-up-oriented configuration for the same sequence set. Compared with the results of the previous configuration, which focuses on coding efficiency, it can be seen that the BD-rate increases by different proportions (around 0.73% on average). Nevertheless, the TR also undergoes a notable increase, which can reach values of up to 69.96% in some cases. It should be noted that even though the increase might not seem significant in appearance, small gains in TR represent a large increase in terms of speed-up when the TR is higher than a certain value. Additionally, it should be noted that the target variables do not present the same variation for each sequence. This, again, is dependent on the hit rate of the algorithm. As some

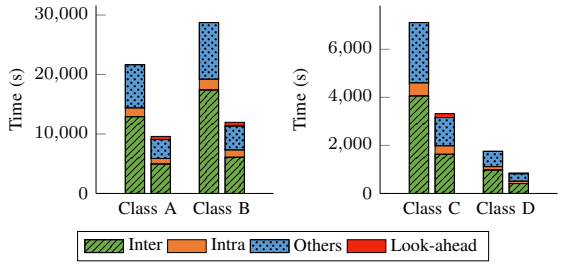


Fig. 6. Profiling of the baseline (left side bar) compared with the proposal (right side bar) using the BD-rate-oriented configuration.

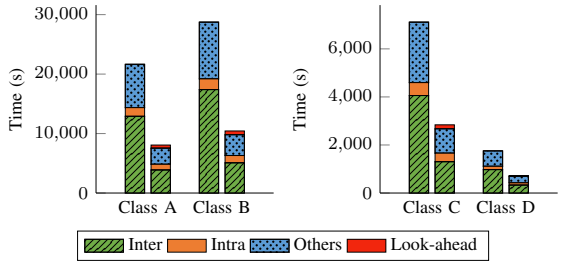


Fig. 7. Profiling of the baseline (left side bar) compared with the proposal (right side bar) using the speed-up-oriented configuration.

PUs are not evaluated in comparison with the BD-rate-oriented configuration, the coding efficiency could be penalized if the proposed algorithm misses the prediction and the PU which offered a better result is omitted. In any case, the results show significant time savings at the cost of some coding efficiency losses that prove marginal in many scenarios.

In order to show the effect of the proposed algorithm on the encoder, Fig. 6 shows the per-class profiling results obtained by the BD-rate-oriented configuration compared with the baseline encoder, while Fig. 7 displays the corresponding results for the speed-up-oriented configuration. These timing values have been extracted from the average profiles of all the tested sequences and QP values. It should be noted that the total encoding time for class A is lower on average than class B in spite of the greater resolution. This is caused by the lower number of frames in these sequences.

With regard to the BD-rate-oriented configuration, it can be seen that both the "Intra" and "Inter" modules are equally reduced, which can be justified by the fact that this configuration either evaluates or omits all the PUs in a given depth level. This differs from the behavior of the speed-up-oriented configuration, which omits some inter-predicted PUs of the parent and child CUs. As a result, it can be seen in Fig. 7 that the "Inter" module is given less encoding time in the speed-up-oriented configuration than in the BD-rate-oriented one. Finally, it should be noted that the look-ahead stage represents only around 4% of the total encoding time, which proves negligible compared with the time savings achieved.

TABLE IX
AVERAGE RESULTS OBTAINED BY THE RELATED WORKS

Type	Related Work	BD-rate (%)	TR (%)
CU	L. Shen et al. [10]	1.2	28.5
	X. Shen et al. [12]	1.9	44.7
	S. Ahn et al. [18]	1.0	35.3
	J. Xiong et al. [20]	1.1	41.9
PU	X. Huang et al. [16]	2.0	39.5
CU+Skip	L. Shen et al. [10]	1.5	40.5
	S. Ahn et al. [18]	1.4	49.6
	J. Lu et al. [19]	4.4	35.1
	J. Xiong et al. [20]	2.2	54.8
CU+PU	J. Lee et al. [17]	2.7	51.7
Proposed (BD-rate-oriented)		**2.4**	**56.8**
Proposed (speed-up-oriented)		**3.1**	**61.4**

D. Comparison with Related Works

In this subsection, the results obtained by the proposed algorithm are compared with those of some of the related works presented in Section III, which have a strong relationship with the algorithm proposed in this paper. In order to perform a fair comparison, only those works that reported BD-rate and time reduction results following the coding conditions established by the JCT-VC in [29] were selected. As an additional requirement, these results must have been obtained using the Random Access configuration under the Main profile. In this regard, Table IX shows the average results obtained by the related works as reported by the authors in their documents. These have been classified according to the type of algorithm the authors propose, which basically depends on whether it reduces the computational complexity of the CTU/CU partitioning (group "CU"), if it is aimed at reducing the number of PUs evaluated in each CU (group "PU"), or both approaches at the same time (group "CU+Skip" or "CU+PU"). It should be noted that the works [10], [18] and [20] appear in two of the groups because the authors provided the results of the approaches separately.

The first point that should be noted is that the time reduction obtained by the proposed algorithm using any of the configurations is considerably larger than that of the related works shown in the table. This could be explained by the fact that some of these only have an effect on either CU or PU selection ("CU" and "PU" groups, respectively), and not both at the same time. Nevertheless, a theoretical reduction of around 75% could be obtained by only selecting the optimal quadtree, as was demonstrated by Configuration 1 in Section V-A. This fact suggests that the aforementioned algorithms adopt a conservative behavior to avoid coding losses. In other words, the proposed algorithm is able to skip CUs and PUs more frequently, which results in slightly higher coding efficiency penalties, but also in notably superior speed-ups. This statement is also applicable for the joint schemes shown in the table, which obtain lower time reductions that range from 35.1% to 54.8%.

From the point of view of the trade-off between coding efficiency and speed-up, the works from the groups "CU"

TABLE X
AVERAGE RESULTS OBTAINED BY PREVIOUS PROPOSALS

Proposal	BD-rate (%)	TR (%)
Integer ME [25]	1.0	14.7
Integer ME and reference selection [26]	0.4	16.2
Integer ME, reference selection and intra [27]	0.5	16.3
Proposed (BD-rate-oriented)	**2.4**	**56.8**
Proposed (speed-up-oriented)	**3.1**	**61.4**

and "PU" obtain better coding efficiency results on average than the joint algorithms. This is caused by the fact that these algorithms still evaluate all the PUs in some CUs, or some PUs of the full quadtree, respectively, which also limits the speed-up obtained by the algorithms. In this way, it can clearly be seen that the proposed algorithm outperforms [10], [18] and [20] from group "CU", and [16], given that the time reduction obtained is far greater. With regard to [12], the authors obtain reasonable speed-ups, but lower than our proposal. Given that the BD-rate obtained is similar to the one of the proposed BD-rate-oriented configuration, it can be concluded that our proposed algorithm is better than the Bayesian-based algorithm proposed by the authors.

Concerning the rest of the state-of-the-art works, [17] and [19] are clearly outperformed by our algorithm in terms of both coding efficiency and time reduction. Additionally, the joint version of [10] is in turn outperformed by [18]. Considering that the hit rate of the partitioning algorithms does not increase linearly, and thus neither does the BD-rate, it seems reasonable to assume that if the joint algorithms in [18] and [20] obtained similar speed-ups to the proposed algorithm, the coding efficiency would also rise to an equivalent BD-rate. Taking this into consideration, we can consider that both algorithms obtain roughly equivalent results, while our proposal achieves a higher speed-up.

In summary, our proposed algorithm achieves higher time reduction than all the related works analyzed in this section. While some of them present slightly better coding efficiency, our algorithm displays a better trade-off between the two target variables. Moreover, whereas some of these works propose complex algorithms to accelerate CTU/CU partitioning, such as data mining solutions, our scheme is able to outperform them by using much simpler methods. Additionally, unlike other algorithms whose applicability is limited to quadtree partitioning, the proposed algorithm obtains valuable information such as MVs and distortion costs, which can be used for other purposes, e.g. rate control, or bit allocation. However, these uses are beyond the scope of this paper.

Finally, Table X summarizes the results obtained by some of our previous proposals. In a similar way to this paper, these proposals calculate preliminary information that is later used to accelerate the encoder. In these works, however, this information is used to speed up the ME, but not the splitting decision algorithm. For this reason, while there is a difference in coding efficiency, the time savings achieved by the proposal in this paper are notably higher, given that the ME represents a smaller fraction of the encoding time.

VI. CONCLUSIONS

The introduction of new coding tools in HEVC, and also the enhancement of existing ones, has led to a large increase in coding efficiency compared with prior standards, but also to a huge increment in the computational complexity of the encoder. With this in mind, the main focus of this paper is on tackling this complexity and adapt it for real-world scenarios, in which fast encoding is a requirement. In this regard, we propose a CTU/CU partitioning algorithm for P and B slices that is based on a look-ahead stage. In this stage, a preliminary ME is performed to obtain the distortion costs of the square blocks into which the picture is divided, which are later used by the algorithm to determine the corresponding partitioning quadtree. As a result, a large number of CUs which are unlikely to provide good R-D results can be omitted, resulting in large speed-ups.

In order to avoid costly misses, the encoder not only tests the PUs included in the CU at which the proposed algorithm decides to stop splitting, but also some PUs at the next and previous depth levels. In order to determine which PUs in each depth provide the best trade-off between coding efficiency and time reduction, a thorough statistical analysis was carried out to evaluate many different CU/PU configurations. The analysis showed the inability to select only one configuration, so two were proposed in this paper in accordance with the priority given to the target variables. On the one hand, the BD-rate-oriented configuration is able to achieve a 56.80% time reduction at the cost of a 2.38% BD-rate. On the other hand, the speed-up-oriented configuration is able to deliver a greater time reduction, which rises up to 61.41% at the expense of a 3.11% BD-rate. Compared with the methods in related works, the proposed algorithm is able to outperform them in terms of time reduction, while offering a good balance between coding efficiency and speed-up.

ACKNOWLEDGMENTS

This work was jointly supported by the Spanish Ministry of Economy and Competitiveness and the European Commission (FEDER funds) under the project TIN2015-66972-C5-2-R, and by the Spanish Ministry of Education, Culture and Sports under the grant FPU13/04601.

REFERENCES

[1] ISO/IEC and ITU-T, "High Efficiency Video Coding (HEVC). ITU-T Recommendation H.265 and ISO/IEC 23008-2 (version 4)," Dec. 2016.
[2] ——, "Advanced Video Coding for Generic Audiovisual Services. ITU-T Recommendation H.264 and ISO/IEC 14496-10 (version 11)," Feb. 2016.
[3] J.-R. Ohm, G. J. Sullivan, H. Schwarz, Thiow Keng 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.
[4] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC Complexity and Implementation Analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.
[5] 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.
[6] G. J. Sullivan, J.-R. Ohm, Woo-Jin 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.

[7] K. Misra, A. Segall, M. Horowitz, S. Xu, A. Fuldseth, and M. Zhou, "An Overview of Tiles in HEVC," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 969–977, Dec. 2013.
[8] F. Henry and S. Pateux, "Wavefront Parallel Processing," Tech. Rep. JCTVC-E196, Mar. 2011.
[9] L. Shen, Z. Zhang, and Z. Liu, "Effective CU Size Decision for HEVC Intracoding," *IEEE Trans. Image Process.*, vol. 23, no. 10, pp. 4232–4241, Oct. 2014.
[10] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An Effective CU Size Decision Method for HEVC Encoders," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 465–470, Feb. 2013.
[11] H. L. Tan, C. C. Ko, and S. Rahardja, "Fast Coding Quad-Tree Decisions Using Prediction Residuals Statistics for High Efficiency Video Coding (HEVC)," *IEEE Trans. Broadcast.*, vol. 62, no. 1, pp. 128–133, Mar. 2016.
[12] X. Shen, L. Yu, and J. Chen, "Fast Coding Unit Size Selection for HEVC Based on Bayesian Decision Rule," in *Picture Coding Symposium*, May 2012, pp. 453–456.
[13] H.-S. Kim and R.-H. Park, "Fast CU Partitioning Algorithm for HEVC Using an Online-Learning-Based Bayesian Decision Rule," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 130–138, Jan. 2016.
[14] H. Lee, H. J. Shim, Y. Park, and B. Jeon, "Early Skip Mode Decision for HEVC Encoder With Emphasis on Coding Quality," *IEEE Trans. Broadcast.*, vol. 61, no. 3, pp. 388–397, Sep. 2015.
[15] G. Tian and S. Goto, "Content Adaptive Prediction Unit Size Decision Algorithm for HEVC Intra Coding," in *Picture Coding Symposium*, May 2012, pp. 405–408.
[16] X. Huang, Q. Zhang, X. Zhao, W. Zhang, Y. Zhang, and Y. Gan, "Fast Inter-Prediction Mode Decision Algorithm for HEVC," *Signal, Image Video Process.*, vol. 11, no. 1, pp. 33–40, Jan. 2017.
[17] J.-H. Lee, K. Goswami, B.-G. Kim, S. Jeong, and J. S. Choi, "Fast Encoding Algorithm for High-Efficiency Video Coding (HEVC) System Based on Spatio-Temporal Correlation," *J. Real-Time Image Proc.*, vol. 12, no. 2, pp. 407–418, Aug. 2016.
[18] S. Ahn, B. Lee, and M. Kim, "A Novel Fast CU Encoding Scheme Based on Spatiotemporal Encoding Parameters for HEVC Inter Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 422–435, Mar. 2015.
[19] J. Lu, F. Liang, L. Xie, and Y. Luo, "A Fast Block Partition Algorithm for HEVC," in *9th International Conference on Information, Communications Signal Processing*, Dec. 2013, pp. 1–5.
[20] J. Xiong, H. Li, F. Meng, Q. Wu, and K. N. Ngan, "Fast HEVC Inter CU Decision Based on Latent SAD Estimation," *IEEE Trans. Multimedia*, vol. 17, no. 12, pp. 2147–2159, Dec. 2015.
[21] "x264 Source Code Repository," http://git.videolan.org/git/x264.git.
[22] J. Garrett-Glaser, "A Novel Macroblock-Tree Algorithm for High-Performance Optimization of Dependent Video Coding in H.264/AVC," Department of Computer Science. Harvey Mudd College, Tech. Rep., 2009.
[23] "x265 Source Code Repository," http://hg.videolan.org/x265.
[24] L. Sun, O. C. Au, C. Zhao, and F. H. Huang, "Rate Distortion Modeling and Adaptive Rate Control Scheme for High Efficiency Video Coding (HEVC)," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, Jun. 2014, pp. 1933–1936.
[25] G. Cebrián-Márquez, Chi Ching Chi, J. L. Martínez, P. Cuenca, M. Álvarez-Mesa, S. Sanz-Rodríguez, and B. Juurlink, "Reducing HEVC Encoding Complexity Using Two-Stage Motion Estimation," in *IEEE International Conference on Visual Communications and Image Processing (VCIP)*, Dec. 2015.
[26] G. Cebrián-Márquez, J. L. Martínez, and P. Cuenca, "A Pre-Analysis Algorithm for Fast Motion Estimation in HEVC," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 2013–2017.
[27] ——, "Inter and Intra Pre-Analysis Algorithm for HEVC," *J. Supercomput.*, vol. 73, no. 1, pp. 414–432, Jan. 2017.
[28] "HEVC Test Model (HM) Reference Software," https://hevc.hhi.fraunhofer.de/.
[29] F. Bossen, "Common Test Conditions and Software Reference Configurations," Tech. Rep. JCTVC-L1100, Jan. 2013.
[30] G. Bjøntegaard, "Calculation of Average PSNR Differences Between RD-Curves," ITU-T Video Coding Experts Group (VCEG), Tech. Rep. VCEG-M33, Mar. 2001.
[31] C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. J. Sullivan, "High Efficiency Video Coding (HEVC) Test Model 16. Improved Encoder Description Update 7," Tech. Rep. JCTVC-Y1002, Jan. 2017.
[32] M. Friedman, "The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance," *J. Am. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, 1937.
[33] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.
[34] S. Holm, "A Simple Sequentially Rejective Multiple Test Procedure," *Scand. J. Stat.*, vol. 6, no. 2, pp. 65–70, 1979.
[35] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics*, vol. 1, no. 6, pp. 80–83, 1945.
[36] J. Sole, R. Joshi, Nguyen Nguyen, Tianying Ji, M. Karczewicz, G. Clare, F. Henry, and A. Dueñas, "Transform Coefficient Coding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1765–1777, Dec. 2012.
[37] R. Hee Gweon, Y.-L. Lee, and J. Lim, "Early Termination of CU Encoding to Reduce HEVC Complexity," Tech. Rep. JCTVC-F045, Jul. 2011.

Gabriel Cebrián-Márquez received the B.Sc. and the M.Sc. degrees in Computer Science and Engineering from the University of Castilla-La Mancha (Spain) in 2013 and 2014, respectively. He is currently studying a Ph.D. in Advanced Computing Technologies at the same university. In 2013, he joined the Laboratory of High-Performance Networks and Architectures (RAAP) as a research assistant at the Albacete Research Institute of Informatics (I3A). He has also been a visiting researcher at Technische Universität Berlin (Germany) and Ghent University (Belgium). His current research interests and areas of publication include video processing and coding, parallel and heterogeneous architectures, and high-performance computing.

José Luis Martínez received his M.Sc. and Ph.D. degrees in Computer Science and Engineering from the University of Castilla-La Mancha (Spain) in 2007 and 2009, respectively. In 2005, he joined the Department of Computer Engineering at the University of Castilla-La Mancha, where he was a researcher in the Computer Architecture and Technology group at the Albacete Research Institute of Informatics (I3A). In 2010, he joined the department of Computer Architecture at the Complutense University in Madrid where he was an assistant lecturer. In 2011, he rejoined the Department of Computer Engineering of the University of Castilla-La Mancha, where he is currently a lecturer. His research interests include video coding, video standards, video transcoding, and parallel video processing. He has also been a visiting researcher at the Florida Atlantic University, Boca Raton (USA) and Centre for Communication System Research (CCSR), at the University of Surrey, Guildford (UK). He has over 100 publications in these areas in international refereed journals and conference proceedings.

Pedro Cuenca received his M.Sc. degree in Physics (Electronics and Computer Science, extraordinary award) 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 (Spain), 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. He has served in the organization of International Conferences as Chair and Technical Program Chair. He was the Chair of the IFIP 6.8 Working Group during the 2006–2012 period. He has also been the Dean of the Faculty of Computer Engineering at the University of Castilla-La Mancha during the 2008–2016 period.

CHAPTER 6

Fast Inter CU Partitioning Based on a Look-Ahead Stage for HEVC

- **Title**: Fast Inter CU Partitioning Based on a Look-Ahead Stage for HEVC.
- **Authors**: Gabriel Cebrián-Márquez, José Luis Martínez, and Pedro Cuenca.
- **Type**: journal paper.
- **Journal**: IEEE Transactions on Multimedia.
- **Publisher**: Institute of Electrical and Electronics Engineers (IEEE).
- **ISSN**: 1520-9210.
- **Status**: under review.
- **Submission date**: December 2017.
- **JCR IF/ranking**: 3.509/Q1 (JCR2016).

Fast Inter CU Partitioning Based on a Look-Ahead Stage for HEVC

Gabriel Cebrián-Márquez, José Luis Martínez, and Pedro Cuenca

***Abstract*—High Efficiency Video Coding (HEVC) has become the state-of-the-art video coding standard. It outperforms its predecessor, H.264/MPEG-4 Advanced Video Coding (AVC), by up to 50% in terms of bitrate reduction for similar perceptual quality. This improvement stems from the introduction of new coding tools in the standard, such as the new quadtree-based partitioning scheme called the coding tree unit (CTU), which enables significant encoding flexibility by being recursively split into coding units (CUs), prediction units (PUs) and transform units (TUs). However, selecting the optimal partitioning requires the evaluation of numerous possibilities, which involves long computing times that hinder the applicability of the standard in real-world scenarios. With this in mind, the main focus of this paper is on tackling this complexity by means of a fast CTU/CU partitioning and mode decision algorithm based on a look-ahead stage. This stage performs a preliminary motion estimation that provides the motion costs used to build the least-cost quadtree, which is in turn utilized to conduct the partitioning. On the basis of this quadtree, the encoder may decide to terminate the partitioning early, or to evaluate additional depth levels. A thorough experimental evaluation of the algorithm shows that it can reduce the encoding time by 65.33%, at the expense of only a 1.35% BD-rate for the random access configuration. Combined with a fast inter prediction algorithm, this reduction can rise to 70.55%, while the coding efficiency is maintained at a 1.83% BD-rate. When compared with other related works, these results display an excellent trade-off between the two variables.**

***Index Terms*—HEVC, fast encoding, pre-analysis, partitioning, mode decision, inter prediction.**

I. Introduction

OVER the last decade, there has been a marked increase in the demand for a better quality of experience with multimedia contents. The growing popularity of high-definition video, and the advent of new formats such as 4k and 8k has motivated research into novel coding tools that improve the coding efficiency of state-of-the-art standards. Moreover, the traffic generated by video-on-demand services is imposing severe strain on communication networks. As a way to address these demands, the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) organizations developed the High Efficiency Video Coding (HEVC) standard, as part of the Joint Collaborative Team on Video Coding (JCT-VC) [1]. The first version of the standard was released in early 2013, but several revisions have been published since then to support additional application scenarios, such as range extensions with enhanced precision and color format support, multiview and 3D, scalable coding, and screen content coding.

The new HEVC standard is usually considered as the evolution of its predecessor, namely H.264/MPEG-4 Advanced Video Coding (AVC) [2]. Content providers and manufacturers adopted H.264/MPEG-4 AVC in countless scenarios because of its good coding efficiency, which turned it into the most widely used video coding standard. Nevertheless, all the aforementioned demands for improved coding efficiency, higher quality and new video formats motivated the development of HEVC. In this way, in addition to addressing these demands, this standard is able to achieve a 35.4% bitrate saving on average for an equal peak signal-to-noise ratio (PSNR), or around a 50% bitrate saving based on subjective quality assessments compared with H.264/MPEG-4 AVC [3]. This improved performance has attracted the attention of a large number of companies, which already implement this standard in many of their products.

Among all the encoding tools introduced in HEVC, it is worth highlighting the new block-based hierarchical structure into which pictures are partitioned. This structure, called the coding tree unit (CTU), can be subsequently divided into coding units (CUs), and these into prediction units (PUs) and transform units (TUs). As a result, the standard is highly adaptable to the features of the picture, which leads to better coding efficiency. Nevertheless, the process of determining the best frame partitioning requires evaluating a large number of possibilities. Consequently, the computational complexity of HEVC has increased considerably with respect to prior standards, becoming one of the most significant constraints on the implementation of this standard on current devices [4].

The optimal partitioning of a frame is determined via two main processes: CTU/CU splitting, and the mode decision. In the former, the encoder needs to identify the partitioning quadtree that provides the highest coding efficiency. This is usually accomplished by following a top-down strategy in which at each depth level it is necessary to decide whether to split the corresponding CU. In the latter process, once the CTU structure has been determined, each CU is assigned a prediction mode and a PU scheme, minimizing the residual generated and the bits required to signal the block. The process of finding the optimal partitioning and prediction mode is usually called rate-distortion (R-D) optimization.

Given the increased complexity of the encoder, the reduction of the encoding time has become one of the most significant objectives in the research community. Many authors have proposed fast encoding algorithms that focus on CTU/CU splitting or the mode decision. However, in spite of the

The authors are with the High-Performance Networks and Architectures group of the University of Castilla-La Mancha. Campus Universitario s/n 02071, Albacete, Spain. E-mail: {Gabriel.Cebrian, JoseLuis.Martinez, and Pedro.Cuenca}@uclm.es

Manuscript received December 22, 2017.

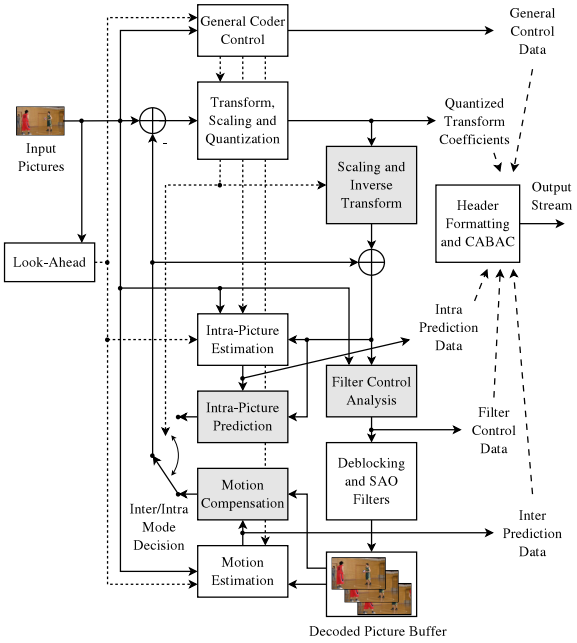


Fig. 1. Typical HEVC encoder architecture with a look-ahead stage.

complex design of some of these methods, many of them are unable to achieve significant time savings or to maintain the coding efficiency of the standard. In this regard, in this paper we propose a novel CTU/CU partitioning and mode decision algorithm that achieves a significant reduction in the total encoding time without a noticeable impact on coding efficiency, outperforming many other related proposals in the literature. Hence, the main aim pursued by this algorithm is to enable the feasibility of HEVC in real-time scenarios.

The proposed scheme is based on a preliminary stage named look-ahead, which pre-processes the input frames to obtain valuable information for the encoding, as shown in Fig. 1. In particular, the proposed look-ahead stage performs a fast motion estimation (ME), and in the process it obtains an estimate of the distortion cost and a motion vector (MV) for each reference frame and each square block into which the pictures are divided. This information enables the building of the least-cost partitioning tree, which is utilized to conduct CTU/CU splitting. In this way, the splitting process begins with the evaluation of the estimated quadtree, omitting all the CUs at higher and lower depth levels. As a result, the partitioning that is most likely to provide the optimal R-D results is evaluated first. From this point forward, the proposed algorithm can terminate the CTU early, or it can adaptively evaluate additional CUs according to certain rules. In addition to these splitting capabilities, the proposed algorithm can also omit the evaluation of certain PUs that are unlikely to provide better R-D results than the current best. As a result, the total encoding time is significantly reduced at the expense of imperceptible coding efficiency losses.

The rest of this paper is organized as follows. Section II contains a thorough analysis of the most relevant related works in this field. The proposed splitting and mode decision algorithm, which is based on a look-ahead stage, is described in Section III. An experimental evaluation is presented in Section IV, and Section V provides an in-depth comparison of the proposal with respect to the related works. Finally, conclusions are drawn in Section VI.

II. Related Work on Fast Encoding

Fast encoding has always been a popular issue in the field of HEVC. Many approaches have been adopted to reduce the total encoding time of the standard. In particular, CTU/CU splitting and the mode decision are the processes that have most frequently been addressed due to the large time savings that can be obtained as a result. This section will highlight the most relevant works related to fast encoding methods. For this purpose, these have been classified according to the method proposed, beginning with those that only address the CTU/CU splitting decision, and followed by others that also take into consideration fast prediction methods, early skip detection schemes, and mode selection algorithms.

A. CTU/CU Splitting

According to statistical analyses, the partitionings of neighboring CTUs usually correlate. For this reason, many authors have studied the way to estimate the partitioning from previously coded blocks. For example, *Q. Zhang et al.* based their method on spatial and temporal neighboring information to classify the CUs into simple, normal, and complex, which enables the determination of the minimum and maximum depth levels for a CTU [5]. Similarly, the algorithm proposed by *Z. Pan et al.* limits the partitioning to depths 0 or 2 in accordance with statistics obtained from the neighbors [6]. *Y. Gao et al.* established the relationship between the least frequently used CU sizes and the quantization parameter (QP), building a model which enables the omission of certain partitions based on the encoding information for previous groups of pictures (GOPs) [7].

On the basis of observation, many authors found a relationship between the features of the pictures and the coding parameters, and the splitting decisions. In this way, many works exploit this relationship to find the optimal CTU partitioning. *T.-L. Lin et al.* proposed an early CU termination algorithm based on the residual statistics of the current CU and the skip information of the neighbors [8]. The residual is also exploited by *H.-L. Tan et al.* in their work [9]. With regard to inter prediction, *J. Xiong et al.* proposed a splitting decision method based on the existing MV divergence in a CU [10]. Later, the same authors modeled this as a Markov random field inference problem, optimized by a graph cut algorithm [11]. *J. Liu et al.* based the splitting decision on the existence of non-zero coefficients after the transform, and also on spatial and temporal depth information, or gradients when the neighbors are not available or do not correlate [12]. Similarly, the method proposed by *Y. Li et al.* also bases the splitting decision on the existence of non-zero coefficients after the transform, and the

maximum depth of temporally co-located CTUs [13]. Finally, some authors perform an analysis of the input pictures, such as *D. G. Fernández et al.*, who proposed an early termination algorithm based on the spatial and temporal homogeneity of the images [14].

The search for the optimal partitioning has also been considered as a classification problem. In this regard, the use of data mining techniques is present in many fast encoding methods. For instance, the early CU termination algorithm proposed by *G. Corrêa et al.* is based on classification trees [15]. However, this type of algorithms requires evaluating unnecessary depth levels until it is decided to stop splitting. As a way to solve this, the algorithm proposed by *Y. Zhang et al.* enables the selection of a specific depth level by means of a joint classifier composed of multiple binary classifiers with different parameters for each depth, together with an R-D model to determine these parameters [16]. To overcome the disadvantages of the static models, *H.-S. Kim and R.-H. Park* proposed the use of Bayesian rules for on-line and off-line training with scene change detection [17]. *S. Momcilovic et al.* in turn made use of machine learning techniques, in particular those based on neural networks, to avoid the requirement of prior training [18].

In contrast to these works, different approaches include, for example, the complexity control algorithm proposed by *A. Jiménez-Moreno et al.* and which is based on a hierarchical approach, which makes it possible to parametrize the encoding to choose the desired complexity [19].

B. CTU/CU Splitting and Inter/Intra Prediction

Pruning the partitioning tree results in significant time savings. Nevertheless, the encoder still needs to perform both the inter and the intra prediction to select the optimal prediction mode. As a consequence, many authors complemented CTU/CU splitting with fast prediction algorithms. For instance, the algorithm proposed by *S. Wang et al.* performs the splitting decision based on thresholds with respect to the R-D cost of each CU [20]. It reduces the number of intra candidates by means of gradients, and also reduces the number of reference frames to be checked. The method proposed by *H. Kibeya et al.* is based on the early detection of the residual matrices that after transformation and quantization result in zero coefficients, so that CTU/CU splitting and the ME are terminated early [21]. Also related to inter prediction, *T. Mallikarachchi et al.* introduced two classification models based on the inter $N \times N$ costs to estimate the partitioning, enabling the reuse of motion vectors [22].

C. CTU/CU Splitting and Early Skip Detection

The use of fast prediction algorithms has numerous advantages in terms of reduction of the encoding time, but it does not solve the problem of evaluating unnecessary prediction modes. Taking into consideration the frequent use of the skip mode, in particular for low rate encodings, a different approach is to detect the use of this mode in an early stage, enabling the omission of the remaining modes for a CU. In this way, *S. Ahn et al.* made use of spatial and temporal parameters in their skip detection method and CTU/CU splitting decision algorithm, namely the sample adaptive offset (SAO) parameters, MVs, TU sizes, and the existence of non-zero coefficients [23]. The method defined by *Q. Hu et al.* employs the Neyman-Pearson decision rule to determine whether to use the skip mode, and to stop splitting a CTU based on the statistical parameters of previous partitioning decisions [24]. As a different approach, *M. Tang et al.* proposed the reordering of the prediction modes, so first the merge/skip and the $2N \times 2N$ modes are evaluated, then the child CUs, and finally the remaining prediction modes of the current CU [25]. In this way, it is possible to prune the quadtree or perform early skip detection.

As mentioned above, the use of machine learning techniques is always present in this type of algorithms. For example, *J. Lee et al.* defined a set of Bayesian rules based on thresholds with three main objectives: as a way to omit large CUs, to early terminate CTU/CU partitioning, and to define an early skip detection method [26]. Similarly, *J. Zhang et al.* classified the CUs into three categories with the use of Bayesian rules and the merge R-D cost, enabling their early pruning or an early detection of the skip mode [27]. For those CUs in which it is not possible to make a decision, and for I slices, the authors used conditional random fields.

Finally, it may be worth mentioning the extension made by *J. Xiong et al.* to their prior works in CTU/CU partitioning. In this regard, they moved towards the use of sum of absolute difference (SAD) costs in a two-layer ME to determine whether to split a CU or not, which shares some similarities with the algorithm proposed in this paper [28]. They also provided this algorithm with an early skip detection method based on the R-D costs obtained by the n previous CUs coded with this mode.

D. CTU/CU Splitting and Mode Decision

In the preceding subsections, it has been possible to see a progression towards greater time savings by integrating more elements in the fast encoding algorithms, such as fast prediction methods, or skip detection schemes. While the latter enable the omission of unnecessary evaluations, if the skip mode is not selected, then the encoder is required to check all the prediction modes. For this reason, many authors have proposed mode decision algorithms, which limit the number of PUs to be evaluated for a given CU.

In a similar way to the CTU/CU splitting algorithms of the previous subsection, the use of neighboring information is also very frequent. As an example, the algorithm proposed by *L. Shen et al.* carries out the splitting decision on the basis of spatial neighboring CTUs [29]. Additionally, they propose an early termination method based on motion homogeneity, the R-D costs and the skip mode. Based on this work, *Z. Liu et al.* proposed alternative uses of the neighboring information to predict the partitioning and the prediction mode [30], [31]. *J.-H. Lee et al.* introduced a scheme in which the splitting decision is based on the R-D cost of the parent CU and the cost of the CUs in the current depth, and also a PU decision algorithm driven by spatial and temporal parameters, and depth correlations [32]. Finally, the method proposed by *W. Zhao et*

al. classifies the CTUs into simple, complex or medial, so that the first or the last depth levels can be skipped according to the information of temporal neighbors [33]. The selection of PUs is conducted based on the residual of the $2N \times 2N$ inter prediction, and a fast intra mode decision algorithm.

With regard to machine learning techniques, and continuing on from their previous work, *G. Corrêa et al.* proposed a method based on decision trees to early terminate CTU/CU splitting, PU selection and residual quadtree (RQT) generation [34]. Similarly, *I. Zupancic et al.* introduced a variation of the visiting order of the CUs, so that depending on the prediction of a Bayesian classification model, the processing is performed top-down or bottom-up, using early termination and inter mode decision algorithms [35].

Alternative approaches include the algorithm proposed by *Y.-J. Ahn and D. Sim*, which is based on the costs of the square PUs [36]. First, in a top-down approach, the R-D cost of these blocks is calculated, and then, in a bottom-up fashion it is decided whether it is necessary to check the remaining sizes according to the accumulated cost of the child CUs. *X. Wu et al.* addressed CU, PU and TU selection as an optimal stopping problem driven by reward functions [37]. More recently, *Z. Yang et al.* based the early CU splitting algorithm and the PU decision on edge information, namely edge similarity and edge movement [38].

III. Proposed CTU/CU Splitting and Mode Decision Algorithm

As shown in the previous section, the existing works on fast encoding make use of diverse techniques to reduce the encoding time, such as machine learning, reuse of neighboring information, and feature extraction, among others. The method proposed in this paper, however, is based on a look-ahead stage, which enables the obtaining of valuable information from the input sequence prior to the encoding. In particular, the proposed preliminary stage calculates the motion information of the input sequence in a similar way to the normal encoding process. The resulting inter prediction information is then used to estimate the optimal quadtree partitioning, and to guide the encoding through the mode decision process. As a consequence, the total encoding time is notably reduced.

This model offers two major advantages in comparison with other schemes. Firstly, the similarities between the operations performed in the two stages result in a more optimal decision-making process when predicting the encoding decisions from the information obtained in the look-ahead stage. Secondly, the estimated motion information can be used, at no additional cost, for different purposes, such as rate control, scene change detection, bit allocation, perceptual coding or, as is the case for the proposed algorithm, acceleration of the encoding process.

The proposed scheme is thus divided into two stages: the look-ahead and the encoding itself. Both steps are described in detail in the following subsections.

A. Look-Ahead Stage

The main aim of the look-ahead stage is to obtain the inter prediction information used to guide the CTU/CU splitting and mode decision processes. To this end, this stage needs to perform an ME that is similar to the inter prediction module of the encoder. Nevertheless, given that the look-ahead stage only needs to provide an estimate of the motion in a frame, its design can be notably simpler than the normal ME.

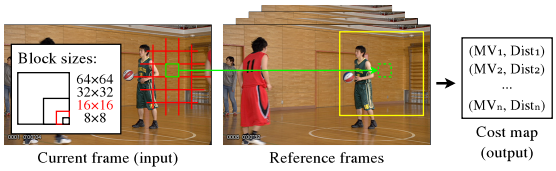


Fig. 2. Diagram of the ME performed by the look-ahead stage.

Evaluating all the possible partitions defined in HEVC would be excessively time-consuming. For this reason, the look-ahead stage is restricted to square blocks: 8×8, 16×16, 32×32, and 64×64. This provides great flexibility when estimating the motion information in a frame. In this regard, the ME is carried out for each block size and and on every reference frame, as depicted in Fig. 2, resulting in a data structure that stores the distortion value in terms of the SAD of each tuple. This same information is the one used later by the encoder to aid the splitting and mode decisions.

With the aim of obtaining the most reliable cost map possible, certain considerations were taken into account in the design of the look-ahead stage:

- The same search algorithm is used in both the look-ahead stage and the inter prediction module of the encoder. In this way, it is more likely that the two stages predict similar motion information. Nevertheless, given that the look-ahead stage is required to provide only an estimate of the costs, the precision of the search is limited to integer-pel samples instead of performing the full quarter-pel search, resulting in a faster estimation.
- The advanced motion vector prediction (AMVP) feature of the HEVC standard is assimilated in the look-ahead stage in such a way that the MVs of adjacent blocks are used as predictors. In this way, the MVs of the blocks above the current one and the one to the left are retrieved and used as predictors. If any of these are not available, a temporal candidate is used instead. If the number of available predictors after this selection process is lower than two, the list is filled with zero MVs, $(0, 0)$. To determine which predictor is finally used in the ME, both candidates are evaluated to select the one providing the lowest distortion.
- The following Lagrangian multiplier R-D optimization model is used:

$$J\left(MV\right) = Dist\left(MV\right) + \lambda \cdot Rate\left(MV\right), \qquad (1)$$

where J represents the cost function for a given MV, $Dist$ the distortion function in terms of SAD, λ is the Lagrangian multiplier, and $Rate$ represents the function to calculate the bits required to code the MV. It should be noted that the rate is dependent on the previously selected predictor.

B. CTU/CU Partitioning and Mode Decision Algorithm

With the motion information obtained in the look-ahead stage, the idea behind the proposed method is to calculate an estimate of the partitioning for P and B slices. In this way, the processing of CUs and PUs is started by evaluating the corresponding blocks at the leaves of the estimated partitioning tree. From this point, the evaluation is then conducted using a top-down approach, and then bottom-up, enabling the early termination of both the CTU/CU splitting and mode decision processes. In essence, the proposed approach is to first evaluate the partitioning that is most likely to provide the best R-D outcome, and then perform small adjustments according to the results obtained.

The estimated partitioning tree is built from the motion information of the look-ahead stage. From bottom to top, the decision of splitting or not splitting a CU is determined according to its corresponding estimated cost, and the sum of accumulated costs of lower depths. Beginning from the leaves at depth $D-1$ to the top at depth 0, where $D \in \{1,2,3,4\}$ represents the maximum partitioning depth allowed by the encoder, costs are recursively grouped into fours. If the sum of the costs estimated for each child CU is lower than the cost of a given CU, then the CU will be split, as this provides better coding efficiency according to the look-ahead algorithm.

In order to obtain the most accurate estimation of the partitioning, it is necessary to define the cost function that will determine the splitting decision. Encoders typically take the splitting decision on the basis of both the rate component and the distortion, as shown in Eq. 1. In a similar way, the proposed algorithm makes use of the motion information provided by the look-ahead stage to predict such parameters. Considering that a CU can be identified within a CTU as $CU_{d,i}$, where $d \in [0, D-1]$ is the depth of the CU, and $i \in \left[0, 4^d - 1\right]$ is the index of the CU in Z-order for depth d, its associated estimated cost $C_{d,i}$ can be defined as:

$$C_{d,i} = \min_{r \in [0,R-1]} \{Dist\left(MV_{d,i,r}\right) + \lambda \cdot Rate\left(MV_{d,i,r}\right)\}, \tag{2}$$

where R is the number of reference frames for the CU, $MV_{d,i,r}$ is the MV of the co-located look-ahead block for $CU_{d,i}$ and reference frame r, $Dist$ corresponds to the distortion function in terms of SAD, λ is the Lagrangian multiplier, and $Rate$ represents the rate component calculated as the number of bits needed to code the MV with respect to the predictor used. As can be seen, the calculation of $C_{d,i}$ is modeled on Eq. 1. It should also be noted that the QP takes part in the cost function through λ, so that high QP values lead to a higher weight for the rate component, and therefore to a lower fragmentation.

Having established the estimated cost function for a given CU, it is also necessary to define the procedure to determine the optimal partitioning quadtree according to these costs. In this regard, starting from the smallest CUs at depth $D-1$ to the largest CU at depth 0, the following criterion is followed:

$$S\left(CU_{d,i}\right) = \begin{cases} \text{split}, & \text{if } \sum_{j=0}^{3} C^{\star}_{d+1,4i+j} < C_{d,i} \\ & \text{and } d < D-1 \\ \text{no split}, & \text{otherwise} \end{cases} \tag{3}$$

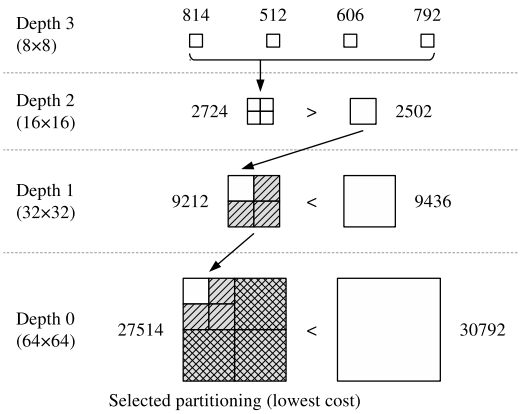


Fig. 3. Estimation of the CTU/CU partitioning quadtree following a bottom-up approach.

where S represents the partitioning decision, and $C^{\star}$ is the cost assigned to a CU once the splitting decision has been taken. This new cost is defined as follows:

$$C^{\star}_{d,i} = \begin{cases} \min\left\{\sum_{j=0}^{3} C^{\star}_{d+1,4i+j}, C_{d,i}\right\}, & \text{if } d < D-1 \\ C_{d,i}, & \text{otherwise} \end{cases} \tag{4}$$

In conclusion, the procedure defined by Eq. 3 and 4 enables the construction of the optimal partitioning quadtree on the basis of the look-ahead costs. An example of its application is shown in Fig. 3. As can be seen, the costs are recursively grouped into fours, from the smallest blocks to the largest one. The numbers shown in the figure are examples of the sum of the $C^{\star}$ costs assigned to the child CUs, and the C costs associated with the blocks, for the split and non-split CUs respectively. Blocks shaded in gray represent CUs whose partitioning has been determined by following this procedure, but these become irrelevant for understanding the figure.

The result of the proposed tree building algorithm is optimal from the point of view of the look-ahead costs. Nevertheless, the quadtree obtained might not necessarily match the one that the encoder would have selected after a full R-D optimization process on all the possible combinations. This may involve coding inefficiencies due to the increase in the number of bits needed to encode the CTUs. There are three main reasons for this mismatch. Firstly, the predictors calculated in the look-ahead stage are used to avoid drifted MVs, and to calculate the rate component in the cost function. However, these may differ from the ones finally used in the encoding, producing alternative MVs, and thus different distortion costs and rate values. Secondly, P and B slices can contain intra-predicted PUs. Nevertheless, the percentage of inter-predicted PUs is significantly greater than the intra-predicted ones in these types of slices, so it is assumed that the partitioning can be accurately estimated from only the inter prediction costs. And finally, the proposed scheme makes use of $2N \times 2N$ blocks to predict the partitioning, but HEVC defines additional PU sizes.

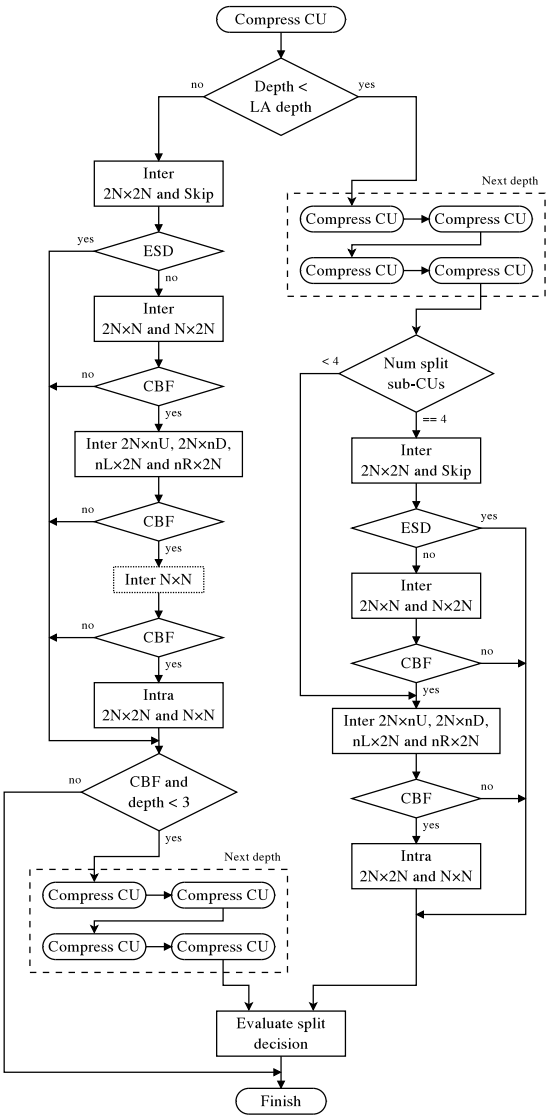


Fig. 4. Proposed guided evaluation performed by the encoder on the basis of the estimated quadtree.

One approach to reduce the total processing time would be restricting the encoding to the partitioning defined by the estimated quadtree. Nevertheless, this would also lead to coding inefficiencies due to the mismatching issues enumerated above. For this reason, a safer approach has been considered for the proposed algorithm. In particular, the encoder adaptively evaluates CUs and PUs in higher and lower depth levels if certain conditions are met, which helps find the optimal prediction, and thus improve the coding efficiency.

The proposed scheme performs a recursive evaluation of the CTU, starting from depth 0 and subsequently splitting the CUs until it is decided to terminate the process early, or the maximum depth is reached. However, the evaluation of some CUs is temporarily omitted in this first top-down pass. In particular, the evaluation of all the CUs above the leaves of the estimated quadtree is postponed until a second bottom-up pass is performed. In this way, the first pass processes the CUs that were predicted as non-split by the proposed algorithm, allowing the evaluation of their child CUs only if the early termination condition is not met, whereas the second pass processes the corresponding parent CUs on the basis of the encoding information from the first pass. As a result, this processing order enables the evaluation of the partitioning and prediction modes that are most likely to provide the best coding efficiency at an early stage, while it also provides the encoder with the adaptivity to check additional modes. This scheme is depicted in Fig. 4.

The splitting and mode decisions are taken by the coded block flag (CBF). This flag accounts for the significance of the entire TU, i.e. it signals whether a TU contains non-zero transform coefficients [39]. Accordingly, it is possible to assume that there is a direct relationship between the splitting of a CU and the CBF. In this regard, assuming that the optimal prediction mode has a similar size to the leaf CU of the estimated quadtree, the approach followed in the top-down evaluation is to check modes from larger to smaller sizes, where the CBF acts as the stopping condition for both the mode decision and CTU/CU splitting, as shown in the left-side branch of Fig. 4. As a particular case, the first modes to be checked are the $2N \times 2N$ inter prediction and skip, for which an early skip detection (ESD) condition is defined instead. The ESD condition not only relies on the CBF, but it also checks whether the motion vector difference (MVD) for these modes is zero, as proposed by *J. Yang et al.* [40]. If the CBF is set, and the skip mode is selected or the MVD is zero, then the remaining modes are omitted and the splitting is terminated early.

The approach followed in the bottom-up evaluation is also based on the use of the CBF and the ESD condition. Nevertheless, the most significant difference with respect to the top-down pass is that the encoding information of the child CUs is available. As a result, it is possible to omit certain PUs on the basis of their encoding decisions. In particular, the $2N \times 2N$, $2N \times N$ and $N \times 2N$ inter prediction modes cover the same area as all or some of these child CUs, and they also match their edges. For this reason, it is straightforward to consider that if the child CUs were split, then this region of the picture would not be homogeneous and then these modes would not be adequate, as they would increase the number of bits necessary to encode the residual. In this way, Fig. 4 shows in its right-side branch how these three symmetric modes are skipped if at least one of the child CUs is split.

Finally, it is worth pointing out the fast encoding methods that the encoder implements with regard to the asymmetric motion partitioning (AMP) modes: $2N \times nU$, $2N \times nD$, $nL \times 2N$ and $nR \times 2N$. To reduce the complexity of the encoding, some of these modes are omitted. In particular, only the horizontal or vertical modes are checked in the top-down evaluation if the current best mode for a given CU is either $2N \times N$ or $N \times 2N$, respectively, while both are evaluated if

$2N \times 2N$ provides the lowest R-D result [41]. Additionally, the prediction mode selected for the parent CU can also determine the evaluation of only the merge mode for the AMP sizes. In the bottom-up pass, a combination of both the current best mode for the CU and the splitting decision of the child CUs is taken into consideration. For example, the $nR \times 2N$ will be evaluated if the current best is unknown or $N \times 2N$, and the two left-most child CUs are not split. A similar approach is followed for the remaining three modes. If the current best is $2N \times 2N$, then the four AMP modes are evaluated, unless the skip mode is used, in which case the four sizes are restricted to the merge mode.

C. Integration With Other Fast Encoding Methods Based on a Look-Ahead Stage

As mentioned above, one of the most significant advantages of the proposed method is that the information calculated in the look-ahead stage can be used in the encoder for other purposes. In previous works, we utilized the motion information of this preliminary stage to accelerate inter prediction [42], [43]. The aim of the algorithm in those works was thus the same as in this paper, i.e. to reduce the total encoding time. However, that approach is completely different from the splitting and mode decision algorithm proposed in this paper. Firstly, the distortion costs were used to delimit the set of reference frames used for prediction for each PU. Secondly, the estimated MVs were used as an accurate starting point for the ME, leading to a reduction of the search area, and thus of the number of SAD calculations. In the end, the time devoted to inter prediction was notably reduced, while coding efficiency was maintained.

In contrast to these earlier works, the algorithm proposed in this paper functions at the CTU/CU level. Therefore, the two approaches can work together with the aim of reducing the total encoding time to a greater extent by accelerating the splitting, the mode decision, and inter prediction. In this regard, the performance of the proposed method will be evaluated with, and without integration with the aforementioned works to assess the contribution of both approaches.

IV. Experimental Evaluation

The experiments were executed on the HEVC Test Model (HM-16.6) reference software [44], following the guidelines and coding conditions provided by the document elaborated by the JCT-VC [45]. In this regard, the QP values tested were 22, 27, 32 and 37, and the selected configurations were random access (RA), low delay P (LDP), and low delay B (LDB). The encodings were carried out using the Main profile and 8-bit depth. Sequences of classes A to F according to the JCT-VC classification were used, which include the following resolutions: 2560×1600 (A), 1920×1080 (B), 832×480 (C and F), 416×240 (D), 1280×720 (E and F), and 1024×768 (F). Classes A to D comprise camera-view scene content, class E is composed of videoconferencing content, and class F consists of screen content (non-camera-captured and mixed imagery). A search area of 128×128 pixels was used for the look-ahead stage, which is equivalent to the default search area for the baseline encoder.

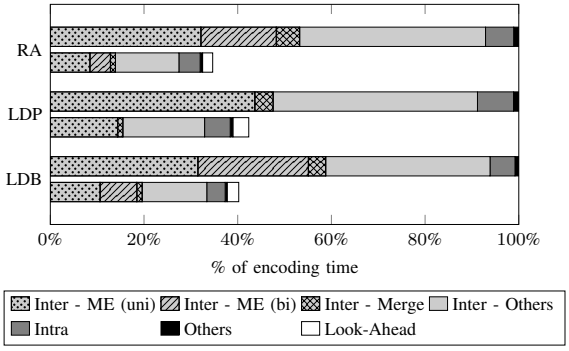


Fig. 5. Profiling of the baseline (upper bar) compared with the proposed algorithm (lower bar) for each configuration.

The hardware platform used in the experiments was composed of an Intel® Xeon® E5-2630L v3 CPU running at 1.80 GHz. The encoder was compiled with GCC 4.8.5-4 and executed on CentOS 7 (Linux 3.10.0-327). Turbo Boost was disabled to ensure the reproducibility of the results.

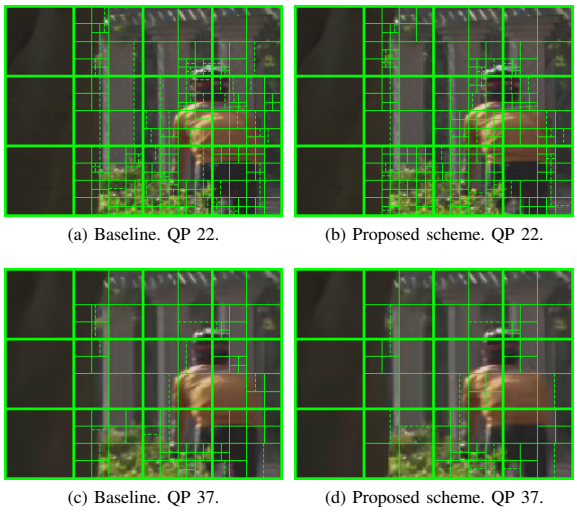
The results are given in terms of time reduction (TR) and the Bjøntegaard delta rate (BD-rate). The BD-rate is a measure of coding efficiency that represents the percentage of bitrate variation between two sequences with the same objective quality [46]. Higher time reduction and lower BD-rate values represent better results.

Table I shows the average TR for the four QP values and the BD-rate for each test sequence and coding configuration. As can be seen, the RA configuration obtains better coding efficiency results and higher time savings on average than the other two configurations. This can be attributed to the coding pattern used in RA compared with the low-delay configurations, which potentially results in more distinct R-D values, and thus the splitting and mode decisions become easier to take. The results also show that the proposed algorithm obtains significantly positive results for the three sequences of class E and *SlideEditing*. This is caused by the low motion displayed in these sequences, which enables the selection of the skip mode more frequently, added to the ability of the algorithm to adapt the visiting order of the CUs. As a result, the TR can reach values of more than 86% at the expense of a very small increase in bit rate. With respect to the rest of the sequences, it is possible to observe a slightly higher TR for high-resolution sequences compared with those of lower resolution. In this case, the reason lies in the fact that the partitions are usually larger, and thus the proposed algorithm can omit a larger number of partitions.

In order to show the effect of the proposed algorithm on the encoder, Fig. 5 shows the profiling results obtained by the proposed algorithm compared with the baseline encoder. These timing values have been extracted from the average profiles of all the tested sequences and QP values for each encoding configuration. In this regard, the most important information to be extracted from the figure is the amount of time devoted to the look-ahead stage. As can be seen, this stage takes

TABLE I
PER-SEQUENCE RESULTS OF THE PROPOSED CTU/CU SPLITTING AND MODE DECISION ALGORITHM

Class	Video sequence	Random Access		Low Delay P		Low Delay B	
		BD-rate (%)	TR (%)	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
A	Traffic	1.20	73.41	1.66	66.15	1.90	68.22
	PeopleOnStreet	1.76	55.66	1.29	47.86	1.50	50.00
B	Kimono	1.54	60.24	1.55	49.56	1.50	51.64
	ParkScene	1.06	70.53	1.63	61.38	1.55	63.38
	Cactus	1.52	63.07	2.18	53.68	1.87	56.53
	BasketballDrive	1.70	58.87	1.33	48.82	1.45	51.74
	BQTerrace	1.91	70.15	2.76	62.86	1.99	63.91
C	BasketballDrill	1.10	56.71	1.05	47.19	1.07	49.84
	BQMall	1.55	65.44	1.61	56.32	1.72	59.04
	PartyScene	1.14	59.75	1.47	52.05	1.34	54.02
	RaceHorses	1.90	51.93	1.18	43.23	1.26	45.40
D	BasketballPass	1.50	55.19	1.48	47.27	1.48	50.06
	BQSquare	1.22	65.92	2.21	59.39	1.77	60.59
	BlowingBubbles	1.05	61.00	1.67	51.97	1.64	53.88
	RaceHorses	1.87	52.83	1.41	44.61	1.63	46.58
E	FourPeople	0.74	79.94	1.51	73.62	1.90	75.63
	Johnny	0.94	82.60	2.52	78.29	2.35	79.68
	KristenAndSara	0.91	79.42	1.45	72.68	1.63	74.40
F	BasketballDrillText	1.08	57.60	0.96	48.67	1.10	50.82
	ChinaSpeed	2.31	51.65	1.88	43.40	1.91	46.64
	SlideEditing	0.21	86.79	0.75	85.69	0.25	86.89
	SlideShow	1.55	78.48	1.61	73.24	1.64	76.44
	Average (all sequences)	1.35	65.33	1.60	57.63	1.57	59.79



(a) Baseline. QP 22. (b) Proposed scheme. QP 22.

(c) Baseline. QP 37. (d) Proposed scheme. QP 37.

Fig. 6. Comparison of the CTU/CU and PU partitioning between the baseline encoder and the proposed approach.

approximately 2–3% of the total encoding time (6–8% with respect to the encoding time of the proposal). Considering the time savings achieved, this amount of time proves negligible. It is worth noting also that the information calculated in the look-ahead stage can be reused for other purposes without increasing the complexity of this stage.

An illustrative comparison between the baseline encoding and the one performed by the proposed algorithm is shown in Fig. 6. This figure displays the partitioning performed by the two alternatives of the 104th frame of the *ParkScene* sequence for the RA configuration with two different QPs. As can be seen, the two methods are able to adapt the partitioning to the features of the picture. In this regard, it should be noted that even though the difference in the splitting and mode decisions can be caused by the omission of some of the CUs and PUs by the proposed method, they can also differ because these new decisions become optimal in terms of R-D.

On average, the TR obtained by the proposed CTU/CU algorithm reaches 65.33%, which is almost two thirds of the total encoding time, while the low-delay configurations range from 57.63% to 59.79%. The increase in bit rate is, in turn, negligible, not exceeding 1.60% on average. As analyzed above, the proposed scheme obtains significant TRs for low-motion sequences, and also high-resolution video, which is the main target of HEVC, thus representing an excellent solution for this new generation of video coding.

As mentioned in Section III-C, the proposed scheme can be combined with other fast encoding algorithms. In particular, the motion information calculated in the look-ahead stage can be used to further accelerate the encoding, such as when using the fast inter prediction algorithm proposed in our previous works [42], [43]. As a way to show the effects of this integration, the proposed splitting and mode decision algorithm has been combined with this complementary algorithm with the aim of providing greater time savings. The average results of this integration are shown in Table II. As can be seen, the fast

TABLE II
RESULTS OF THE INTEGRATION OF THE PROPOSED ALGORITHM WITH OUR PREVIOUS WORK

Configuration	Prior Work [42], [43]		Proposed + [42], [43]	
	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
Random Access	0.55	18.36	1.83	70.55
Low Delay P	0.55	21.17	2.09	65.93
Low Delay B	0.79	22.95	2.32	68.20

inter prediction algorithm enables an additional TR of around 8% for the RA configuration, and approximately 14% for the low-delay configurations, with respect to that offered by the proposed algorithm, which is reflected in a very significant TR that is larger than 70% for RA. The impact in terms of BD-rate is still very low, not exceeding 2% for RA, which represents an excellent trade-off between time savings and coding efficiency.

V. COMPARISON WITH RELATED WORKS

In this subsection, the results obtained by the proposed algorithm are compared with those of the related works presented in Section II. All of these works aim to omit the evaluation of CUs that are unlikely to provide optimal R-D results, and thus have a strong relationship with the method proposed in this paper. The results will be compared with respect to the average BD-rate and TR values reported by the authors in their works. For this reason, only those works that followed the coding conditions established by the JCT-VC in [45] were selected. Additionally, the results were rounded to the highest common precision of the values provided. The comparison is made for the three coding configurations: RA, LDP and LDB. Consequently, the same work may appear in more than one configuration.

Table III summarizes the results obtained by the related works using the RA configuration. These have been sorted according to their type and in ascending order of TR. The most significant conclusion that can be drawn from the table is that the proposed algorithm outperforms the vast majority of the works in terms of TR. Moreover, the integration of our proposal with the fast inter prediction algorithm represents the fastest scheme among all the related works. In terms of coding efficiency, some of the algorithms provide a slightly lower BD-rate, but at the expense of a notably lower TR. As a way to facilitate understanding of the values in the table, Fig. 7a depicts the performance achieved by these works. Points located in the bottom-right corner represent higher TR values and a lower BD-rate, and thus more optimal results. The splitting and mode decision algorithm proposed in this paper has been denoted as *A*, whereas its integration with the fast inter prediction algorithm proposed in [42] and [43] has been called *B*. Related works have been grouped by colors according to their type.

As can be seen, the algorithms proposed by *Q. Hu et al.* [24] and *G. Corrêa et al.* [34] obtain very similar results to the proposed algorithm. Some of the other algorithms either improve the coding efficiency or obtain larger time savings, but at the expense of worse trade-offs between the two variables.

TABLE III
COMPARISON WITH THE RELATED WORKS (RA CONFIGURATION)

Type	Related Work	BD-rate (%)	TR (%)
CU	Y. Gao et al. [7]	1.0	24
	G. Corrêa et al. [15]	0.3	37
	H.-L. Tan et al. [9]	1.2	45
	H.-S. Kim and R.-H. Park [17]	0.7	54
	T.-L. Lin et al. [8]	0.5	55
	Y. Li et al. [13]	1.3	56
	J. Xiong et al. [11]	2.7	63
CU+Pred	H. Kibeya et al. [21]	1.6	23
CU+Skip	M. Tang et al. [25]	0.9	48
	Z. Liu et al. [30]	0.8	49
	S. Ahn et al. [23]	1.4	50
	J. Zhang et al. [27]	1.2	55
	J. Xiong et al. [28]	2.0	58
	Q. Hu et al. [24]	1.3	65
	J. Lee et al. [26]	3.0	69
CU+PU	I. Zupancic et al. [35]	1.6	33
	L. Shen et al. [29]	1.5	42
	X. Wu et al. [37]	1.4	51
	J.-H. Lee et al. [32]	2.7	52
	Z. Yang et al. [38]	1.5	54
	Y.-J. Ahn and D. Sim [36]	1.4	58
	Z. Liu et al. [31]	1.0	60
	G. Corrêa et al. [34]	1.4	65
	W. Zhao et al. [33]	1.9	68
	Proposed algorithm	**1.4**	**65**
	Proposed algorithm + [42], [43]	**1.8**	**71**

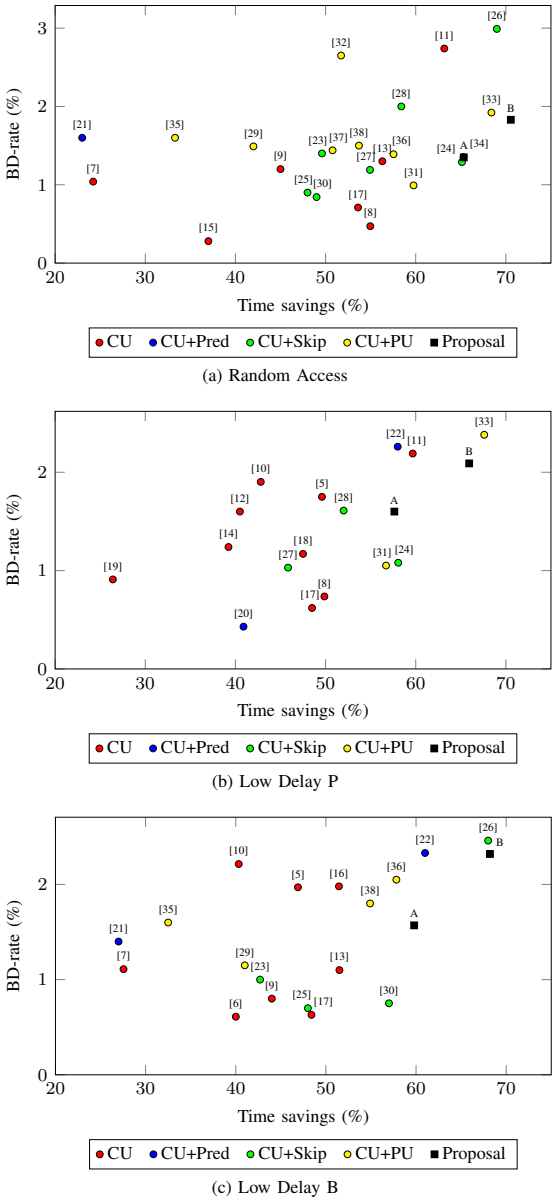
TABLE IV
COMPARISON WITH THE RELATED WORKS (LDP CONFIGURATION)

Type	Related Work	BD-rate (%)	TR (%)
CU	A. Jiménez-Moreno et al. [19]	0.9	26
	D. G. Fernández et al. [14]	1.2	39
	J. Liu et al. [12]	1.6	41
	J. Xiong et al. [10]	1.9	43
	S. Momcilovic et al. [18]	1.2	48
	H.-S. Kim and R.-H. Park [17]	0.6	49
	Q. Zhang et al. [5]	1.8	50
	T.-L. Lin et al. [8]	0.7	50
	J. Xiong et al. [11]	2.2	60
CU+Pred	S. Wang et al. [20]	0.4	41
	T. Mallikarachchi et al. [22]	2.3	58
CU+Skip	J. Zhang et al. [27]	1.0	46
	J. Xiong et al. [28]	1.6	52
	Q. Hu et al. [24]	1.1	58
CU+PU	Z. Liu et al. [31]	1.1	57
	W. Zhao et al. [33]	2.4	68
	Proposed algorithm	**1.6**	**58**
	Proposed algorithm + [42], [43]	**2.1**	**66**

The former work is based on the use of statistical information from the encoding, and thus requires a periodical training stage. As a consequence, frequent scene changes may have a negative impact on its performance. By contrast, the algorithm proposed in this paper does not require training periods, given that both the look-ahead stage and the encoder will adapt their decisions to the input sequence. The latter work makes use of data mining techniques and, in particular, decision trees.

TABLE V
COMPARISON WITH THE RELATED WORKS (LDB CONFIGURATION)

Type	Related Work	BD-rate (%)	TR (%)
CU	Y. Gao et al. [7]	1.1	28
	Z. Pan et al. [6]	0.6	40
	J. Xiong et al. [10]	2.2	40
	H.-L. Tan et al. [9]	0.8	44
	Q. Zhang et al. [5]	2.0	47
	H.-S. Kim and R.-H. Park [17]	0.6	48
	Y. Zhang et al. [16]	2.0	51
	Y. Li et al. [13]	1.1	52
CU+Pred	H. Kibeya et al. [21]	1.4	27
	T. Mallikarachchi et al. [22]	2.3	61
CU+Skip	S. Ahn et al. [23]	1.0	43
	M. Tang et al. [25]	0.7	48
	Z. Liu et al. [30]	0.8	57
	J. Lee et al. [26]	2.5	68
CU+PU	I. Zupancic et al. [35]	1.6	33
	L. Shen et al. [29]	1.2	41
	Z. Yang et al. [38]	1.8	55
	Y.-J. Ahn and D. Sim [36]	2.1	58
	Proposed algorithm	**1.6**	**60**
	Proposed algorithm + [42], [43]	**2.3**	**68**



(a) Random Access

(b) Low Delay P

(c) Low Delay B

Fig. 7. Related works compared with the proposed algorithm.

However, the method proposed by the authors is based on off-line training, and thus its major shortcoming is the lack of adaptability to sequences that might differ from those used in the training phase. In fact, *T. Mallikarachchi et al.* analyzed this algorithm in detail, and showed that the penalty in terms of coding efficiency can even exceed a 10% BD-rate for some sequences [22]. Therefore, although the two aforementioned works obtain similar average results, the adaptivity of our proposed method makes it more suitable for general use.

The results of the LDP configuration are presented in Table IV. Accordingly, Fig. 7b depicts these values on a plot. Once again, the proposed algorithm outperforms the majority of the related works in this configuration, with the exception of two of them. Firstly, the scheme proposed by *Q. Hu et al.* [24] achieves lower coding efficiency penalties, but involves some issues with scene changes as discussed before. Secondly, *Z. Liu et al.* [31] proposed another algorithm based on statistical analysis that also achieves better coding efficiency. Nevertheless, it requires the use of thresholds, and thus it is not possible to ensure that all of the sequences that are different from the training set will perform adequately. Additionally, while it performs better in this configuration, the algorithm proposed in this paper obtains a substantially better TR than this related work in the RA configuration.

Finally, Table V and Fig. 7c show the corresponding results for the LDB configuration. In this case, as in the previous ones, the proposed scheme offers large time savings and good coding efficiency compared with the related works. Nevertheless, it may be of interest to analyze the work proposed by *Z. Liu et al.* [30], which, although it does not achieve the same degree of TR, results in a lower BD-rate. This method takes the decisions on the basis of neighboring information. While this information enables the prediction of the maximum CU depth in a quadtree, it also requires the use of weights that need to be tuned beforehand. As a result, these static values might have a negative influence on certain sequences that are dissimilar to the sequences used for their calculation. Additionally, it should be noted that the algorithm proposed in this paper visibly outperforms this related work under the RA configuration.

In summary, our proposed algorithm is among the fastest methods in the literature. In fact, its integration with the fast inter prediction algorithm achieves the highest TR for the RA and LDB configurations. While some of the related works present slightly better coding efficiency, our algorithm

displays a better trade-off between time savings and coding efficiency. As shown in Section II, many of the works propose complex algorithms to accelerate CTU/CU splitting and the mode decision, such as machine learning techniques. Nevertheless, our method is able to outperform them by using much simpler methods. Additionally, unlike other algorithms whose applicability is limited to quadtree partitioning, the proposed algorithm obtains valuable information such as MVs and distortion costs, which can be used for other purposes, e.g. rate control and bit allocation.

VI. Conclusions

Among all the coding tools introduced by HEVC, the new CTU-based partitioning scheme provides the flexibility that enables the greatest adaptability of the encoding to the intrinsic features of the sequence compared with its predecessors. Nevertheless, it also involves a huge increment in the computational complexity of the encoder, making it difficult to successfully implement this standard in real-world scenarios. For this reason, the main focus of this paper is on tackling this complexity. In this regard, we propose a CTU/CU partitioning and mode decision algorithm for P and B slices that is based on a look-ahead stage. In this stage, a preliminary ME is performed to obtain the motion information that is used as a basis for the estimation of the partitioning quadtree. This estimation enables the reordering of the evaluation in such a way that the CUs that are predicted as non-split are checked first. Later, the encoder may decide to evaluate CUs at higher or lower depth levels, or to early terminate the CTU. Moreover, some modes may be omitted on the basis of previous encoding decisions and results.

The proposed algorithm exploits the similarity between the look-ahead stage and the inter prediction to perform accurate predictions. In this regard, the experimental evaluation of the proposed scheme shows that it is able to reduce the encoding time by 65.33% for the RA configuration while maintaining the coding efficiency of the standard, with a negligible impact of a 1.35% BD-rate. Apart from the excellent trade-off provided, the proposed algorithm has two key advantages. Firstly, its simplicity compared with other methods, and secondly, the possibility of extending the algorithm by reusing the motion information from the look-ahead stage. As a way to show this, we also evaluated the integration of the proposed method with a fast inter prediction algorithm, showing that this aggregation can achieve a TR of 70.55% at the expense of only a 1.83% BD-rate for the RA configuration. The two methods were compared against the most relevant related works in the field, outperforming them in terms of TR, and thus offering a better trade-off between time savings and coding efficiency.

Acknowledgments

This work was jointly supported by the Spanish Ministry of Economy and Competitiveness and the European Commission (FEDER funds) under the project TIN2015-66972-C5-2-R, and by the Spanish Ministry of Education, Culture and Sports under the grant FPU13/04601.

References

[1] ISO/IEC and ITU-T, "High Efficiency Video Coding (HEVC). ITU-T Recommendation H.265 and ISO/IEC 23008-2 (version 4)," Dec. 2016.
[2] ——, "Advanced Video Coding for Generic Audiovisual Services. ITU-T Recommendation H.264 and ISO/IEC 14496-10 (version 11)," Feb. 2016.
[3] 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.
[4] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC Complexity and Implementation Analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.
[5] Q. Zhang, J. Zhao, X. Huang, and Y. Gan, "A Fast and Efficient Coding Unit Size Decision Algorithm Based on Temporal and Spatial Correlation," *Optik - Int. J. Light Electronic Optics*, vol. 126, no. 21, pp. 2793–2798, Jul. 2015.
[6] Z. Pan, S. Kwong, Y. Zhang, J. Lei, and H. Yuan, "Fast Coding Tree Unit Depth Decision for High Efficiency Video Coding," in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct. 2014, pp. 3214–3218.
[7] Y. Gao, P. Liu, Y. Wu, and K. Jia, "Quadtree Degeneration for HEVC," *IEEE Trans. Multimedia*, vol. 18, no. 12, pp. 2321–2330, Dec. 2016.
[8] T.-L. Lin, C.-C. Chou, Z. Liu, and K.-H. Tung, "HEVC Early Termination Methods for Optimal CU Decision Utilizing Encoding Residual Information," *J. Real-Time Image Process.*, Jun. 2016.
[9] H. L. Tan, C. C. Ko, and S. Rahardja, "Fast Coding Quad-Tree Decisions Using Prediction Residuals Statistics for High Efficiency Video Coding (HEVC)," *IEEE Trans. Broadcast.*, vol. 62, no. 1, pp. 128–133, Mar. 2016.
[10] J. Xiong, H. Li, Q. Wu, and F. Meng, "A Fast HEVC Inter CU Selection Method Based on Pyramid Motion Divergence," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 559–564, Feb. 2014.
[11] J. Xiong, H. Li, F. Meng, S. Zhu, Q. Wu, and B. Zeng, "MRF-Based Fast HEVC Inter CU Decision With the Variance of Absolute Differences," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2141–2153, Dec. 2014.
[12] J. Liu, H. Jia, G. Xiang, X. Huang, B. Cai, C. Zhu, and D. Xie, "An Adaptive Inter CU Depth Decision Algorithm for HEVC," in *2015 Visual Communications and Image Processing (VCIP)*, Dec. 2015, pp. 1–4.
[13] Y. Li, G. Yang, Y. Zhu, X. Ding, and X. Sun, "Adaptive Inter CU Depth Decision for HEVC Using Optimal Selection Model and Encoding Parameters," *IEEE Trans. Broadcast.*, vol. 63, no. 3, pp. 535–546, Sep. 2017.
[14] D. G. Fernández, A. A. Del Barrio, G. Botella, and C. García, "Fast and Effective CU Size Decision Based on Spatial and Temporal Homogeneity Detection," *Multimed. Tools Appl.*, Feb. 2017.
[15] G. Corrêa, P. Assunção, L. Agostini, and L. A. da Silva Cruz, "Fast Coding Tree Structure Decision for HEVC Based on Classification Trees," *Analog Integr. Circ. Sig. Process.*, vol. 87, no. 2, pp. 129–139, May 2016.
[16] Y. Zhang, S. Kwong, X. Wang, H. Yuan, Z. Pan, and L. Xu, "Machine Learning-Based Coding Unit Depth Decisions for Flexible Complexity Allocation in High Efficiency Video Coding," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2225–2238, Jul. 2015.
[17] H.-S. Kim and R.-H. Park, "Fast CU Partitioning Algorithm for HEVC Using an Online-Learning-Based Bayesian Decision Rule," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 130–138, Jan. 2016.
[18] S. Momcilovic, N. Roma, L. Sousa, and I. Milentijevic, "Run-Time Machine Learning for HEVC/H.265 Fast Partitioning Decision," in *2015 IEEE International Symposium on Multimedia (ISM)*, Dec. 2015, pp. 347–350.
[19] A. Jiménez-Moreno, E. Martínez-Enríquez, and F. Díaz-de María, "Complexity Control Based on a Fast Coding Unit Decision Method in the HEVC Video Coding Standard," *IEEE Trans. Multimedia*, vol. 18, no. 4, pp. 563–575, Apr. 2016.
[20] S. Wang, F. Luo, S. Ma, X. Zhang, S. Wang, D. Zhao, and W. Gao, "Low Complexity Encoder Optimization for HEVC," *J. Visual Commun. Image Representation*, vol. 35, no. Supplement C, pp. 120–131, Feb. 2016.
[21] H. Kibeya, F. Belghith, M. A. Ben Ayed, and N. Masmoudi, "Fast Coding Unit Selection and Motion Estimation Algorithm Based on Early Detection of Zero Block Quantified Transform Coefficients for High-Efficiency Video Coding Standard," *IET Image Process.*, vol. 10, no. 5, pp. 371–380, Apr. 2016.
[22] T. Mallikarachchi, D. S. Talagala, H. K. Arachchi, and A. Fernando, "Content-Adaptive Feature-Based CU Size Prediction for Fast Low-

Delay Video Encoding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, 2017.
[23] S. Ahn, B. Lee, and M. Kim, "A Novel Fast CU Encoding Scheme Based on Spatiotemporal Encoding Parameters for HEVC Inter Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 422–435, Mar. 2015.
[24] Q. Hu, X. Zhang, Z. Shi, and Z. Gao, "Neyman-Pearson-Based Early Mode Decision for HEVC Encoding," *IEEE Trans. Multimedia*, vol. 18, no. 3, pp. 379–391, Mar. 2016.
[25] M. Tang, X. Chen, J. Gu, Y. Han, J. Wen, and S. Yang, "Accelerating HEVC Encoding Using Early-Split," *IEEE Signal Process. Lett.*, 2017.
[26] J. Lee, S. Kim, K. Lim, and S. Lee, "A Fast CU Size Decision Algorithm for HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 411–421, Mar. 2015.
[27] J. Zhang, S. Kwong, and X. Wang, "Two-Stage Fast Inter CU Decision for HEVC Based on Bayesian Method and Conditional Random Fields," *IEEE Trans. Circuits Syst. Video Technol.*, 2017.
[28] J. Xiong, H. Li, F. Meng, Q. Wu, and K. N. Ngan, "Fast HEVC Inter CU Decision Based on Latent SAD Estimation," *IEEE Trans. Multimedia*, vol. 17, no. 12, pp. 2147–2159, Dec. 2015.
[29] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An Effective CU Size Decision Method for HEVC Encoders," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 465–470, Feb. 2013.
[30] Z. Liu, T.-L. Lin, and C.-C. Chou, "HEVC Coding-Unit Decision Algorithm Using Tree-Block Classification and Statistical Data Analysis," *Multimed. Tools Appl.*, vol. 76, no. 6, pp. 9051–9072, Mar. 2017.
[31] ——, "Efficient prediction of CU depth and PU mode for fast HEVC encoding using statistical analysis," *J. Visual Commun. Image Representation*, vol. 38, no. Supplement C, pp. 474–486, Jul. 2016.
[32] J.-H. Lee, K. Goswami, B.-G. Kim, S. Jeong, and J. S. Choi, "Fast Encoding Algorithm for High-Efficiency Video Coding (HEVC) System Based on Spatio-Temporal Correlation," *J. Real-Time Image Process.*, vol. 12, no. 2, pp. 407–418, Aug. 2016.
[33] W. Zhao, T. Onoye, and T. Song, "Hierarchical Structure-Based Fast Mode Decision for H.265/HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 10, pp. 1651–1664, Oct. 2015.
[34] G. Corrêa, P. Assunção, L. V. Agostini, and L. A. da Silva Cruz, "Fast HEVC Encoding Decisions Using Data Mining," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 4, pp. 660–673, Apr. 2015.
[35] I. Zupancic, S. G. Blasi, E. Peixoto, and E. Izquierdo, "Inter-Prediction Optimizations for Video Coding Using Adaptive Coding Unit Visiting Order," *IEEE Trans. Multimedia*, vol. 18, no. 9, pp. 1677–1690, Sep. 2016.
[36] Y.-J. Ahn and D. Sim, "Square-Type-First Inter-CU Tree Search Algorithm for Acceleration of HEVC Encoder," *J. Real-Time Image Process.*, vol. 12, no. 2, pp. 419–432, Aug 2016.
[37] X. Wu, H. Wang, and Z. Wei, "Optimal Stopping Theory Based Fast Coding Tree Unit Decision for High Efficiency Video Coding," in *2016 Visual Communications and Image Processing (VCIP)*, Nov. 2016.
[38] Z. Yang, S. Guo, and Q. Shao, "A Fast Inter-Frame Encoding Scheme Using the Edge Information and the Spatiotemporal Encoding Parameters for HEVC," *Multimed. Tools Appl.*, vol. 76, no. 22, pp. 24 125–24 142, Nov. 2017.
[39] J. Sole, R. Joshi, N. Nguyen, T. Ji, M. Karczewicz, G. Clare, F. Henry, and A. Dueñas, "Transform Coefficient Coding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1765–1777, Dec. 2012.
[40] J. Yang, J. Kim, K. Won, H. Lee, and B. Jeon, "Early SKIP Detection for HEVC," Tech. Rep. JCTVC-G543, Nov. 2011.
[41] C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. J. Sullivan, "High Efficiency Video Coding (HEVC) Test Model 16. Improved Encoder Description Update 6," Tech. Rep. JCTVC-X1002, Jun. 2016.
[42] G. Cebrián-Márquez, J. L. Martínez, and P. Cuenca, "A Pre-Analysis Algorithm for Fast Motion Estimation in HEVC," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 2013–2017.
[43] ——, "Inter and Intra Pre-Analysis Algorithm for HEVC," *J. Supercomput.*, vol. 73, no. 1, pp. 414–432, Jan. 2017.
[44] "HEVC Test Model (HM) Reference Software," https://hevc.hhi.fraunhofer.de/.
[45] F. Bossen, "Common Test Conditions and Software Reference Configurations," Tech. Rep. JCTVC-L1100, Jan. 2013.
[46] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," ITU-T Video Coding Experts Group (VCEG), Tech. Rep. VCEG-M33, Mar. 2001.

Gabriel Cebrián-Márquez received his B.Sc. and M.Sc. degrees in Computer Science and Engineering from the University of Castilla-La Mancha (Spain) in 2013 and 2014, respectively. He is currently studying for a Ph.D. in Advanced Computing Technologies at the same university. In 2013, he joined the Laboratory of High-Performance Networks and Architectures (RAAP) as a research assistant at the Albacete Research Institute of Informatics (I3A). He has also been a visiting researcher at Technische Universität Berlin (Germany) and Ghent University (Belgium). His current research interests and areas of publication include video processing and coding, parallel and heterogeneous architectures, and high-performance computing.

José Luis Martínez received his M.Sc. and Ph.D. degrees in Computer Science and Engineering from the University of Castilla-La Mancha (Spain) in 2007 and 2009, respectively. In 2005, he joined the Department of Computer Engineering at the University of Castilla-La Mancha, where he was a researcher in the Computer Architecture and Technology group at the Albacete Research Institute of Informatics (I3A). In 2010, he joined the department of Computer Architecture at the Complutense University in Madrid, where he was an assistant lecturer. In 2011, he rejoined the Department of Computer Engineering of the University of Castilla-La Mancha, where he is currently a lecturer. His research interests include video coding, video standards, video transcoding, and parallel video processing. He has also been a visiting researcher at the Florida Atlantic University, Boca Raton (USA) and Centre for Communication System Research (CCSR), at the University of Surrey, Guildford (UK). He has over 100 publications in these areas in international refereed journals and conference proceedings.

Pedro Cuenca received his M.Sc. degree in Physics (Electronics and Computer Science, extraordinary award) 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 (Spain), 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. He has served in the organization of International Conferences as Chair and Technical Program Chair. He was the Chair of the IFIP 6.8 Working Group during the period 2006–2012. He has also been the Dean of the Faculty of Computer Engineering at the University of Castilla-La Mancha (2008–2016).

CHAPTER 7

GPU-Based Fast Encoding Algorithm for HEVC

In this chapter, we will discuss some of the considerations taken into account in the design of the proposed GPU-based fast encoding algorithm, which represents the result of the integration of the methods proposed in previous works.

The proposed GPU-based ME algorithm is able to calculate the optimal MV for each PU and reference frame in terms of distortion. The look-ahead stage, in turn, calculates some preliminary motion information to accelerate the inter prediction, and the splitting and mode decisions. In view of this, the idea behind the proposed integration is that the look-ahead stage can make use of the GPU-based algorithm not only to offload the ME to the device, but also to obtain more fine-grained motion information. In this way, the look-ahead stage can omit the evaluation of a greater number of CUs and PUs.

The integration will be performed in two steps. First, the GPU-based ME will be used to replace the preliminary ME performed by the look-ahead stage. Additionally, the approach followed to accelerate the inter prediction will be adapted to make use of the motion information provided by the GPU. Nevertheless, the adaptive partitioning model proposed in Chapter 6 will be maintained. In the second step, this model will be adapted on the grounds that the motion information obtained from the GPU is calculated at PU level, and thus the look-ahead stage has more information at its disposal to restrict the evaluation of CUs and prediction modes to a smaller subset.

7.1 Integration of the GPU-Based ME into the Look-Ahead Stage

First, before carrying out the integration of the two approaches, it is necessary to discuss the suitability of the GPU-based ME for the look-ahead stage. The similarities between

the ME processes of the two approaches are clear. They both provide the same type of information as output, that is, MVs and their corresponding distortion costs. However, they also display some differences. For instance, as mentioned above, the GPU-based ME calculates the information at PU level, whereas the one carried out in the look-ahead stage is performed on a square-block basis. While this particular aspect does not hinder the integration, there are other differences that need to be resolved.

The following subsections will describe how these differences are handled, and will also present some preliminary results for this initial integration.

7.1.1 Adaptation of the GPU-Based ME and the Look-Ahead Stage

The first aspect that needs to be examined is the way in which the CPU and the GPU work together. In the original GPU-based ME algorithm, the device was able to calculate the motion information of a CTU. Nevertheless, the look-ahead stage is designed to carry out the pre-analysis on a frame basis. A solution to this situation would be for the GPU to calculate the motion information of the entire frame, and then for the look-ahead stage to perform the remaining operations, or alternatively, for the look-ahead stage to carry out the pre-analysis of only one CTU at a time. The two approaches are equivalent, but they have different advantages and disadvantages, such as the latency introduced at the beginning of a frame. For the particular case of fast encoding, we opted for the reduction of latency, and thus selected the second option, in which the estimation is performed on a CTU basis. Therefore, the communication between the CPU and the GPU is maintained as described in Chapters 2 and 3.

Another important aspect is the use of R-D models in the ME. Given that the original look-ahead stage is sequential, its ME operation implements an R-D model and performs the estimation taking into consideration the use of predictors. The GPU architecture, however, is massively parallel, and thus the introduction of dependencies such as predictors may have a negative effect on the performance. In view of this, and given that the look-ahead stage only requires an estimate of the motion information, we dismissed the idea of implementing an R-D model on the GPU, and assumed that the MVs provided by the device based on the distortion are accurate approximations of the optimal ones. Despite this consideration, it is still possible to estimate the full R-D cost of a PU at a later stage. The way in which this is carried out is, once the look-ahead stage has retrieved the motion information from the GPU, which was calculated taking into account only the distortion, the corresponding rate cost is added to the distortion cost.

These two considerations alone are enough to define the integration of the GPU-based ME algorithm with the look-ahead stage. The remaining aspects to be considered fall within the scope of the algorithms in which the estimated motion information is used. A diagram of the organization of the encoder after the integration of the GPU is shown in Fig. 7.1, which is in turn based on the initial architecture displayed in Fig. 1.3. The new modules

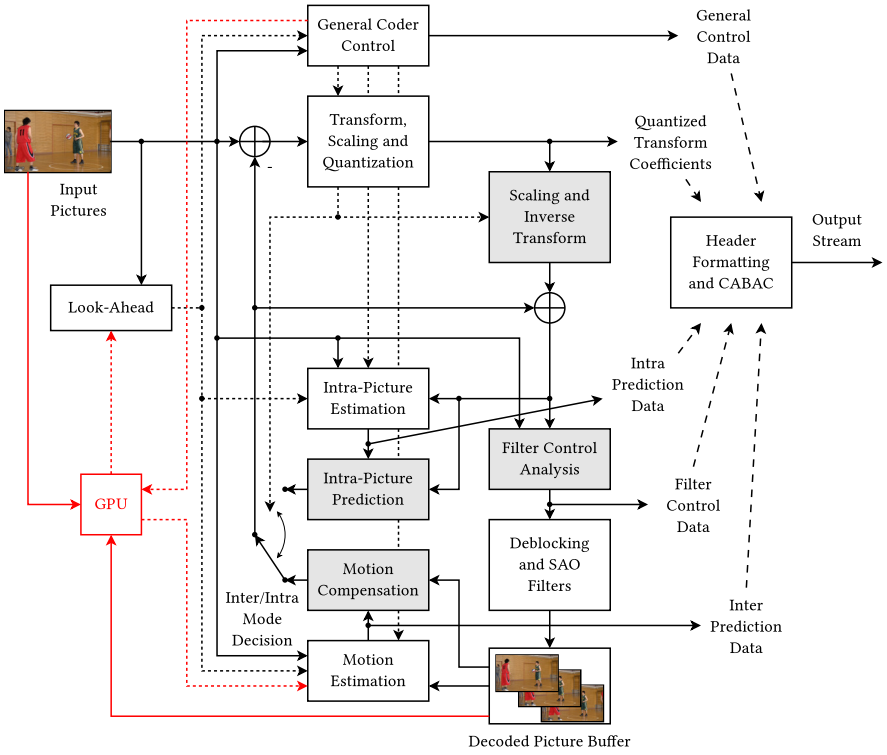


Figure 7.1: Encoding architecture after the integration of the GPU-based algorithm with the look-ahead stage (blocks shaded in gray are shared with the decoder).

introduced in the scheme are marked in red. As can be seen, the GPU needs to receive the input pictures and their reconstructed version to perform the ME. The motion information calculated on the device, which is copied to the CPU, is used to guide the estimation of the partitioning carried out in the look-ahead stage, and to accelerate the inter prediction, and the CTU/CU splitting and mode decisions.

7.1.2 Adaptation of the Inter Prediction, and the Splitting and Mode Decision Algorithm

With regard to the inter prediction, the use of the estimated motion information when carrying out the GPU-based ME is different to that of when the look-ahead stage is present. In the former case, the estimated MVs are used to replace the integer ME carried out in the inter prediction. In the latter case, however, given that this information is calculated on a block basis, it is not possible to use it directly without prior refinement. For this reason, as described in Chapters 4, 5 and 6, the MV of each PU is interpolated with respect to the MVs of the corresponding look-ahead blocks, and later refined by performing a local search.

After having integrated the GPU-based ME into the look-ahead stage, the aforementioned interpolation is no longer required. It can be considered that the MVs retrieved from the GPU are the ones that would have been obtained after interpolating the MVs of the

corresponding look-ahead blocks. In this way, the reference frame selection approach proposed in previous works can proceed as usual, that is, for the 2N×2N PUs, all the reference frames are evaluated, while for the remaining sizes, the set of reference frames taken into consideration will depend on the minimum cost obtained by the estimated MVs and the references used for the 2N×2N PUs.

As far as the ME is concerned, the MVs obtained from the GPU could be used directly to omit the integer ME, as proposed for the original algorithm. Nevertheless, in the case of the integrated algorithm, we decided to take a more conservative approach towards the preservation of the coding efficiency. In this regard, we maintained the local search implemented for the inter prediction in the design of the look-ahead stage. As a result, the MVs gathered from the GPU, which are optimal from the point of view of the SAD, are refined locally using an R-D model.

Finally, with respect to the dynamic CTU/CU splitting and mode decision algorithm implemented in combination with the look-ahead stage, it remained intact in this first step of the integration. Despite the larger amount of information provided by the GPU, we decided not to modify the behavior of this algorithm with the aim of assessing the performance of the changes already introduced. The only adaptation performed in the algorithm was the use of the 2N×2N distortion costs obtained from the GPU instead of the look-ahead costs. The R-D model utilized to calculate the full costs and to simulate the cost of splitting a CU was left unchanged. In this way, the rate component, R, is calculated from the difference between the current MV and the one selected from the neighboring blocks, as defined in the original look-ahead algorithm.

7.1.3 Experimental Evaluation

The scheme described in this section represents the first approximation of the integration of the GPU-based ME into the look-ahead stage. The performance of the proposal is assessed in this subsection in terms of both time savings and coding efficiency.

The experiments were executed on the HM-16.6 reference software [43], following the guidelines and coding conditions provided by the document drawn up by the JCT-VC [44]. In this regard, the QP values tested were 22, 27, 32 and 37, and the selected configurations was RA. The encodings were carried out using the Main profile and 8-bit depth. Sequences of classes A to F according to the JCT-VC classification were used, which include the following resolutions: 2560×1600 (A), 1920×1080 (B), 832×480 (C and F), 416×240 (D), 1280×720 (E and F), and 1024×768 (F). Classes A to D comprise camera-view scene content, class E is composed of videoconferencing content, and class F consists of screen content (non-camera-captured and mixed imagery). A search area of 128×128 pixels was used on the GPU, which is equivalent to the default search area for the baseline encoder.

The hardware platform used in the experiments was composed of an Intel® Xeon® E5-2650 v2 CPU running at 2.60 GHz, and an NVIDIA® Tesla® K40 GPU. The encoder was

Table 7.1: Results of the integration of the GPU-based ME and the look-ahead stage.

Class	Video sequence	BD-rate (%)	TR (%)
A	Traffic	1.51	76.16
	PeopleOnStreet	1.79	66.22
B	Kimono	1.93	68.30
	ParkScene	1.33	74.48
	Cactus	1.90	69.95
	BasketballDrive	2.93	69.07
	BQTerrace	1.96	72.29
C	BasketballDrill	1.16	64.97
	BQMall	2.26	71.57
	PartyScene	1.04	65.58
	RaceHorses	2.44	64.04
D	BasketballPass	1.88	64.73
	BQSquare	1.71	68.74
	BlowingBubbles	1.02	65.06
	RaceHorses	1.70	62.95
E	FourPeople	0.78	80.48
	Johnny	0.99	82.63
	KristenAndSara	0.96	80.94
F	BasketballDrillText	0.70	65.39
	ChinaSpeed	0.70	64.50
	SlideEditing	2.73	84.90
	SlideShow	5.75	79.41
	Average (all sequences)	1.78	71.02

compiled with GCC 4.8.5 and NVIDIA® CUDA 7.0, and executed on CentOS 7. Turbo Boost was disabled to ensure the reproducibility of the results.

The results are given in terms of time reduction (TR) and the BD-rate. Higher TR and lower BD-rate values represent better results.

The results of the proposed GPU-based fast encoding algorithm are shown in Table 7.1. The first conclusion to be highlighted from these values is that the average results obtained are very similar to the ones of the dynamic model presented for the look-ahead stage in Chapter 6. The dynamic model is able to save 70.55% of the encoding time at the cost of a 1.83% BD-rate, while the proposed integration achieves a TR of 71.02% at the expense of a 1.78% BD-rate, which represents a small improvement with respect to the two metrics.

With regard to the BD-rate, it is worth pointing out the results obtained by class E. As can be seen, the increase in BD-rate is negligible, not exceeding 1% in any of the three sequences. This can be explained by the small amount of motion displayed in them, which leads to an easier prediction, and thus to a more accurate partitioning. In fact, this is also

reflected in the TR obtained by this class, which surpasses 80%. Similarly, it is also worth explaining the coding efficiency losses generated by the *SlideShow* sequence. This particular test sequence consists of a series of slides being presented on screen, which results in a very low bit rate. Consequently, small prediction misses can involve a large increase in bit rate in relative terms.

As a closing comment to this section, the similarity of the results obtained by the proposed schemes prior to integration, and the ones achieved after integration was expected. The main change introduced in the look-ahead stage in this first approach is the substitution of the GPU-based ME for the block-based ME. Some adaptations were performed in the inter prediction to assimilate the new information, but the splitting and mode decision algorithm remained untouched. In the next section, this algorithm will be modified to take into consideration the motion information estimated by the GPU for all the PUs in the estimation of the partitioning.

7.2 Design of a New Splitting and Mode Decision Model

In the previous section, the GPU-based ME was successfully integrated into the look-ahead stage to replace the existing preliminary ME. The fast inter prediction techniques implemented in previous proposals were adapted to take advantage of the motion information provided by the device. The adaptive splitting and mode decision approach used together with the look-ahead stage underwent very few modifications, though.

This adaptive approach enables the evaluation of not only the CUs designated by the look-ahead stage, but also some of the corresponding parent and child CUs depending on the intermediate results of the encoding process. Likewise, the evaluation of prediction modes is carried out in accordance with prior encoding decisions. After the integration of the GPU-based ME into the look-ahead stage, however, there is a greater amount of motion information available for the estimation of the CTU/CU partitioning and the prediction modes. The idea behind the proposed alternative splitting and mode decision algorithm is that this fine-grained motion information can lead to a reduction in the number of evaluated CUs and PUs, and in turn to larger speed-ups.

The first aspect that needs to be modified is the partitioning estimation performed in the look-ahead stage. In particular, the cost composition carried out to determine the optimal partitioning from the estimated costs needs to be adapted to use all the PU sizes, and not only the 2N×2N size. With this aim, the same cost function used to estimate the R-D cost of the 2N×2N size can be used for the remaining sizes. In this regard, in order to simplify the calculation of the rate component, R, if a prediction mode is composed of two PUs, the first one is assigned one of the MVs of the neighboring blocks as a predictor, and the second PU is assigned the MV estimated for the first PU. This assignation is always performed by taking into consideration the same reference frame.

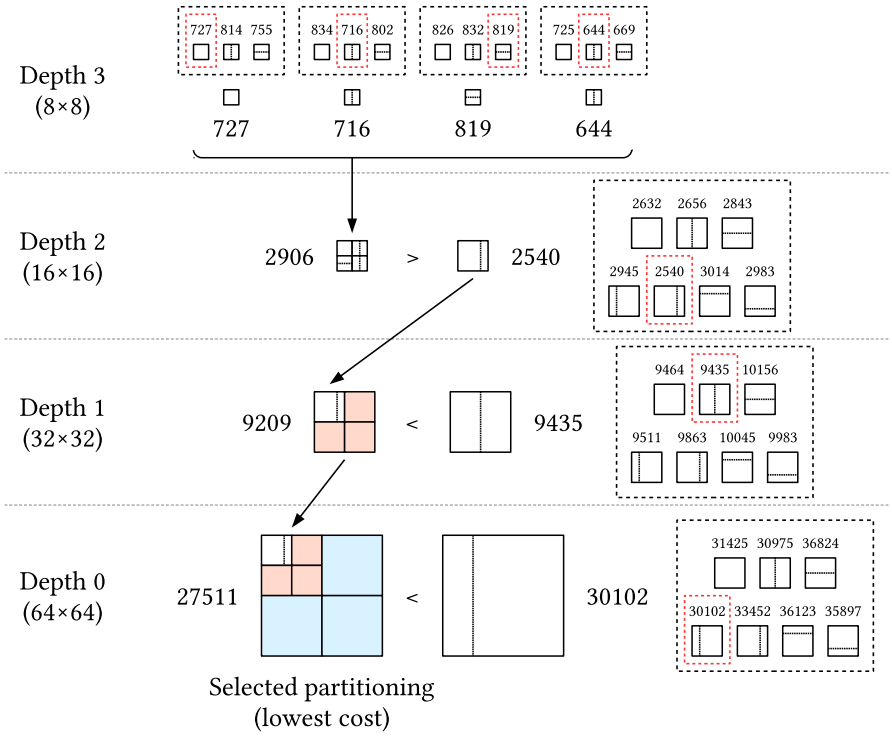


Figure 7.2: Estimation of the CTU/CU partitioning following a bottom-up approach using the estimated PU costs.

An example of the cost composition process used to determine the optimal partitioning on the basis of the look-ahead costs is depicted in Fig. 7.2, which is in turn based on the process shown in Fig. 1.7. As in the original algorithm, the estimated costs are recursively grouped into fours, starting from the leaves and moving to the root of the tree. In this case, however, it is necessary to consider all the PU sizes to determine the optimal partitioning. For this reason, the look-ahead stage does not only store the CTU/CU partitioning, but also the best prediction mode for future reference by the splitting and mode decision algorithm. Colored blocks represent CUs whose partitioning has been determined by following this procedure, but these become irrelevant in understanding the figure.

Once the optimal partitioning and prediction modes have been calculated by the look-ahead stage on the basis of the distortion values and MVs provided by the GPU, it is possible to use them in the encoding process to omit those CU and PU combinations that are unlikely to provide good coding efficiency results. Given that the partitioning is estimated by taking into consideration all the different prediction modes, it is assumed that it is no longer necessary to evaluate other CUs than those designated by the look-ahead stage. With regard to the mode evaluation, the set of PUs evaluated by the encoder is established in accordance with the optimal mode selected by the look-ahead stage. In this regard, Table 7.2 shows the inter prediction modes that are evaluated for the current CU on the basis of the mode selected by the look-ahead stage. White dots represent modes for which only the merge mode is evaluated. As far as the intra prediction is concerned, the default fast encoding

Table 7.2: Inter prediction modes evaluated in accordance with the mode estimated by the look-ahead stage.

		Estimated mode							
		2N×2N	2N×N	N×2N	2N×nU	2N×nD	nL×2N	nR×2N	N×N
Evaluated mode	2N×2N and skip	●	●	●	●	●	●	●	●
	2N×N		●		●	●			
	N×2N			●			●	●	
	2N×nU	○	○		●				
	2N×nD	○	○			●			
	nL×2N	○		○			●		
	nR×2N	○		○				●	
	N×N								●

technique implemented in HM is used, which consists in evaluating this mode if the CBF is set after encoding the CU using inter prediction [67].

As can be seen, the 2N×2N and the skip modes are always evaluated regardless of the mode selected by the look-ahead stage, given that these modes can provide better coding efficiency in homogeneous regions of the picture, or when the direction of the motion is constant. With regard to the remaining modes, they are evaluated on the basis of the PU sizes of the estimated mode, i.e. the horizontal modes are checked if a horizontal mode was selected by the look-ahead stage, and similarly for the vertical modes. However, the symmetrical modes are given higher priority than the asymmetrical modes, for which only the merge mode is checked unless they match the mode selected by the look-ahead stage. As an exception to this, the asymmetrical modes are fully evaluated if the corresponding directional symmetrical mode was previously evaluated and selected as best. For instance, the 2N×nU mode would be fully evaluated if the 2N×N was evaluated and selected as a better alternative than 2N×2N and the skip mode.

In conclusion, the new proposed model takes the splitting and mode decisions on the basis of the estimation performed by the look-ahead stage. This estimation is in turn carried out in accordance with the costs calculated by the GPU. As a result, the encoder is able to omit the evaluation of a greater number of CUs and PUs, which leads to a more significant time reduction compared with the original model.

7.2.1 Experimental Evaluation

This section will show the results of the experimental evaluation performed for the splitting and mode decision model proposed above. In this regard, the same software and hardware platform and the same coding conditions as in Section 7.1 were used in the evaluation. It

Table 7.3: Results of the integration of the GPU-based ME and the look-ahead stage using the alternative splitting and mode decision model.

Class	Video sequence	BD-rate (%)	TR (%)
A	Traffic	5.90	84.86
	PeopleOnStreet	8.56	81.98
B	Kimono	10.18	84.00
	ParkScene	5.55	83.92
	Cactus	9.84	83.52
	BasketballDrive	11.89	84.75
	BQTerrace	8.32	85.17
C	BasketballDrill	9.64	83.21
	BQMall	10.19	83.67
	PartyScene	10.33	81.94
	RaceHorses	10.71	82.53
D	BasketballPass	9.66	80.84
	BQSquare	13.28	80.85
	BlowingBubbles	9.07	80.19
	RaceHorses	9.94	80.05
E	FourPeople	6.22	86.08
	Johnny	6.78	86.73
	KristenAndSara	6.97	86.53
F	BasketballDrillText	8.98	82.88
	ChinaSpeed	15.41	83.51
	SlideEditing	4.13	85.96
	SlideShow	16.32	84.58
	Average (all sequences)	9.45	83.53

should also be noted that the fast inter prediction techniques proposed in previous sections were left enabled. The results are expressed in terms of BD-rate and TR.

Table 7.3 shows the results of the proposed model. As can be seen, the average BD-rate is 9.45%, which is significantly higher than the coding efficiency obtained by the first proposed integration. However, the TR obtained is 83.53% on average, which represents an enormous improvement compared with the previous splitting and mode decision algorithm. In fact, this represents a 43.17% reduction in the average encoding time of the first integration. With regard to the results obtained per class, it can be observed that lower resolutions achieve a lower TR. This can be explained by the proportion of 2N×2N PUs in these sequences compared with high-resolution sequences.

Although the resulting coding efficiency may seem a bit high in absolute terms, the TR achieved is very significant. In fact, if these results are compared with those of some of the best-known HEVC encoders on the market, it is possible to see that the proposed method outperforms them in terms of coding efficiency. Table 7.4 shows the results obtained by

Table 7.4: Results of the open-source HEVC encoders.

Encoder	BD-rate (%)	Speed-up
HomerHEVC [68]	123.7	56.44
DivX265 [69]	19.0	5.24
Kvazaar [70]	67.6	6.91
x265 [71]	37.5	1.93
Proposed method	9.4	6.07

four different open-source HEVC encoders using the highest-quality preset possible [64]. The speed-up of the proposed method was calculated using the following equation:

$$\text{Speed-up} = \frac{T_{\text{baseline}}}{T_{\text{test}}} = \frac{1}{1 - TR} \tag{7.1}$$

where T_{baseline} and T_{tested} represent the encoding time of the baseline encoder and the test encoder, respectively, and TR is the time reduction.

As can be seen in the table, the proposed method achieves a similar speed-up to DivX265 and Kvazaar, and a much superior acceleration compared with the well-known x265 encoder, even when these encoders implement additional fast encoding techniques and hardware optimizations such as intrinsics and vector instruction sets. It is also worth noting that HomerHEVC achieves the highest speed-up, but at the expense of excessive coding efficiency losses. With regard to the coding efficiency, the proposed method obtains the best BD-rate results, outperforming all the evaluated encoders. In particular, it achieves a four times better coding efficiency than x265 on average.

7.3 Conclusions

In this chapter, we have proposed the integration of the GPU-based ME algorithm and the look-ahead stage into a single GPU-based fast encoding algorithm. The motion information calculated by the former is used by the latter to accelerate the inter prediction, and the splitting and mode decision algorithm. Given that the GPU is able to provide more detailed information than the original ME performed in the look-ahead stage, the resulting integration is able to omit a greater number of CUs and prediction modes, which leads to higher speed-ups.

The integration was carried out in two steps. Firstly, we replaced the ME performed in the look-ahead stage with the GPU-based ME, and adapted the proposed fast inter prediction techniques to make use of the new information provided by the device. The splitting and mode decision algorithm originally proposed for the look-ahead stage was left unchanged, though. The results showed that the time reduction and coding efficiency achieved were similar to those obtained by the look-ahead stage before the integration, which was

the expected behavior, and proved that the GPU-based algorithm is suitable for the proposed integration.

In the second step of the integration, the splitting and mode decision algorithm was modified to make full use of the motion information calculated by the GPU. In this way, the look-ahead stage was able to estimate the CTU/CU partitioning, and the optimal inter prediction mode for each CU. On the basis of this estimate, the proposed method was able to omit a greater number of possible combinations, which resulted in an average time saving of 83.53% at the expense of a 9.45% BD-rate.

Compared with some of the current HEVC encoders on the market, it was shown that the proposed GPU-based fast encoding algorithm was able to outperform all of them in terms of coding efficiency, while achieving equivalent time savings. These results highlighted the applicability of the proposed method in scenarios in which it is necessary to provide the best quality or compression rate, e.g. the encoding of multimedia content for video-on-demand services such as Netflix or HBO. In this regard, the proposed method is able to deliver significantly better coding efficiency at the cost of similar encoding time.

CHAPTER 8

Conclusions and Future Work

In this chapter, we conclude this Doctoral Thesis by summarizing the conclusions drawn from the work carried out in the development of our proposal. Then, we discuss the possible objectives that could be tackled in future work.

8.1 Conclusions

As mentioned in Chapter 1, the HEVC standard has attracted the attention of numerous companies in the multimedia sector. Not only does it achieve an enormous improvement in coding efficiency compared to its predecessor, H.264/MPEG-4 AVC, but it also supports a wide range of new formats and resolutions. It aims to respond to the needs imposed by the bandwidth and storage requirements of current consumer trends, and to the demands of the market for higher quality content. Nevertheless, these improvements have been achieved at the cost of an increased computational complexity. In fact, this complexity has become one of the main constraints of this standard in its implementation in current scenarios and devices, and in particular for real-time applications.

In view of this, one of the main conclusions of this Doctoral Thesis is that the development of fast encoding algorithms is essential in overcoming the complexity of HEVC. This conclusion led, in turn, to the definition of the main objective of this work, which corresponds to the reduction of the total time devoted to the encoding, while maintaining the coding efficiency achieved by the standard.

To accomplish this objective, the first goal defined for this Doctoral Thesis was to carry out a study of the standard, and to analyze the state of the art. From this goal, it was possible to conclude that the most time-consuming operation of the encoder was the inter prediction, accounting for approximately 60% of the total encoding time. Moreover, another conclusion was that the process of determining the best CTU/CU splitting and the optimal mode can be very complex if performed for the entire search space.

In view of the conclusions of the first goal, we defined the method aimed at achieving the main objective. In particular, we proposed a fast encoding algorithm using GPU-based heterogeneous platforms targeted at the inter prediction, and the splitting and mode decision algorithms. This approach consisted in the acceleration of these two operations on the basis of the preliminary motion information calculated by a look-ahead stage. In this way, once the main proposal had been established, four additional goals were defined, each of them composed of multiple tasks. These goals included developing a GPU-based ME algorithm, implementing the look-ahead stage to accelerate the inter prediction, then extending it to support the splitting and mode decisions, and finally integrating all these approaches into one.

We developed a GPU-based ME algorithm that performed a full search on the reference area. This exhaustiveness enabled higher coding efficiency in certain cases in which the normal ME was not able to find the optimal candidate. However, the major shortcoming of this approach is the unavailability of neighboring information, and thus the inability to calculate the rate component in the R-D model used to select the optimal MV. In spite of this, the proposed algorithm was able to save approximately 10.46% of the encoding time at the expense of only a 0.2% BD-rate. Apart from showing that this approach is a good method for offloading the ME, these results also indicated that it is a suitable option for calculating the motion information needed by the main method proposed.

The second step in the development of the proposed fast encoding scheme was to design the look-ahead stage. The basic operation performed by this stage is the ME that estimates the motion information used later in the inter prediction. With the aim of performing a fast preprocessing, the ME is performed on a block basis. We concluded, however, that although using one single block size leads to a faster estimation, it also has a negative impact on the coding efficiency. For this reason, we decided to use multiple block sizes. The resulting estimated information was utilized to accelerate the inter prediction in two ways. Firstly, to reduce the search area, and secondly, to omit the evaluation of some reference frames. The two approaches together achieved average time savings of 16.10%, at the expense of only a 0.3% BD-rate.

Once the look-ahead stage had been defined, the next goal was to extend it to support the splitting and mode decisions. In this way, the motion information estimated by this stage was used to calculate an estimate of the CTU/CU partitioning. We analyzed the performance achieved by static models, which evaluated a fixed set of modes on the basis of the estimation performed by the look-ahead stage. While the time savings obtained were significantly high, the lack of adaptivity of these models lead to some coding inefficiencies. Consequently, we designed a dynamic model that adaptively evaluated additional CUs and PUs on the basis of not only the estimate partitioning, but also prior encoding decisions. The results showed a significant improvement in terms of speed-up and coding efficiency. When combined with the aforementioned inter prediction approaches, this model was able to obtain 70.55% time savings at the expense of a negligible 1.8% BD-rate. These values represent an excellent trade-off between the two output variables, outperforming the vast

majority of related fast encoding methods in the literature. Therefore, it can be concluded that the use of preliminary motion information can help predict a significant number of encoding decisions with a negligible impact in terms of coding efficiency.

Finally, we combined the look-ahead stage with the proposed GPU-based ME so that the former could benefit from the motion information calculated by the latter. After pertinent adaptations, it was shown that this combined solution could achieve similar results to the standalone look-ahead stage, proving the suitability of the GPU-based algorithm. Using the information calculated by the device in the estimation of the partitioning and the inter prediction modes, the time reduction rose to 83.53%, at the cost of a 9.45% BD-rate. While the increase in bit rate may seem significant, the achieved time savings are notably high. In fact, this method outperforms some of the best-known open-source HEVC encoders in terms of coding efficiency when using high-quality presets.

In conclusion, the proposed algorithm successfully achieves the main objective defined in this Doctoral Thesis, that is, to reduce the encoding time of HEVC while maintaining the coding efficiency of the standard.

8.2 Future Work

The work presented in this Doctoral Thesis can be extended in several ways. In particular, the integration of the look-ahead stage with the GPU-based algorithm still leaves room for improvement. As mentioned above, although this scheme offers excellent time savings, it also involves significant coding efficiency penalties. Given that this parameter depends directly on the accuracy of the splitting and mode decision algorithm, it would be of interest to analyze this aspect in further detail. In particular, the introduction of all of the partition sizes in the look-ahead stage may make it necessary to define a more complete R-D model to calculate the full costs used to estimate the partitioning and the inter prediction modes. Additionally, an alternative way to obtain the predictors from the neighboring blocks should be explored.

With regard to the GPU-based algorithm, it is clear that the full search carried out as part of the ME is able to provide the best results in terms of SAD. Nevertheless, it also involves larger computation times than other search patterns. While this is not currently an issue in the proposed design, it may need to be taken into consideration if, for example, the GPU is assigned additional tasks, thus increasing its occupancy. In this regard, some instances of search patterns include diamond-shaped patterns, which achieve a good hit rate while evaluating a significantly lower number of candidates. Additionally, the implementation of an R-D model in the GPU-based algorithm could also help improve the resulting coding efficiency. However, this would be a challenge given the number of dependencies introduced in the algorithm.

Alternatively, to further reduce the coding efficiency losses of the proposed method, it would be of interest to consider the use of data mining techniques to provide the splitting and mode decisions with more intelligence. In particular, instead of using the CBF flag and the heuristics of previous encoding decisions to determine the CTU/CU splitting, it would be possible to use classification models such as Bayesian rules. In this way, the proposed splitting and mode decision models may have greater accuracy in terms of hit rate, and thus greater coding efficiency.

Another improvement that could be studied is the use of intra prediction information, along with the motion information currently calculated in the look-ahead stage, to estimate the CTU/CU partitioning. As described in Chapters 1 and 4, we provided the look-ahead stage with intra prediction capabilities with the aim of enabling the applicability of the proposed methods in AI scenarios. However, we did not assess the use of this information to predict the splitting decision, which may be of interest in terms of time savings using this configuration. Regarding RA scenarios, however, the infrequent use of intra prediction in P and B frames, and the difficulty of comparing inter and intra costs without performing full rate-distortion optimization (RDO), still make the use of this information unsuitable with the exception of I frames.

A different future direction of work to be considered is the use of the look-ahead stage for other purposes. Examples of these additional uses are rate control, bit allocation, scene change detection, or the determination of the encoding pattern. All these aspects, which are beyond the scope of this Doctoral Thesis, are alternative applications of the look-ahead stage.

Bibliography

[1] "Cisco Visual Networking Index: Forecast and Methodology, 2016–2021," Cisco, May 2015.

[2] ISO/IEC and ITU-T, "Advanced Video Coding for Generic Audiovisual Services. ITU-T Recommendation H.264 and ISO/IEC 14496-10 (version 11)," Feb. 2016.

[3] ISO/IEC and ITU-T, "High Efficiency Video Coding (HEVC). ITU-T Recommendation H.265 and ISO/IEC 23008-2 (version 4)," Dec. 2016.

[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] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC Complexity and Implementation Analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.

[6] A. Fuldseth, M. Horowitz, S. Xu, and M. Zhou, "Tiles," Tech. Rep. JCTVC-E408, Mar. 2011.

[7] F. Henry and S. Pateux, "Wavefront Parallel Processing," Tech. Rep. JCTVC-E196, Mar. 2011.

[8] I.-K. Kim, J. Min, T. Lee, W.-J. Han, and J. Park, "Block Partitioning Structure in the HEVC Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1697–1706, Dec. 2012.

[9] Z. Pan, S. Kwong, Y. Zhang, J. Lei, and H. Yuan, "Fast Coding Tree Unit Depth Decision for High Efficiency Video Coding," in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct. 2014, pp. 3214–3218.

[10] Q. Zhang, J. Zhao, X. Huang, and Y. Gan, "A Fast and Efficient Coding Unit Size Decision Algorithm Based on Temporal and Spatial Correlation," *Optik - Int. J. Light Electronic Optics*, vol. 126, no. 21, pp. 2793–2798, Jul. 2015.

[11] J. Liu, H. Jia, G. Xiang, X. Huang, B. Cai, C. Zhu, and D. Xie, "An Adaptive Inter CU Depth Decision Algorithm for HEVC," in *2015 Visual Communications and Image Processing (VCIP)*, Dec. 2015.

[12] H. L. Tan, C. C. Ko, and S. Rahardja, "Fast Coding Quad-Tree Decisions Using Prediction Residuals Statistics for High Efficiency Video Coding (HEVC)," *IEEE Trans. Broadcast.*, vol. 62, no. 1, pp. 128–133, Mar. 2016.

[13] T.-L. Lin, C.-C. Chou, Z. Liu, and K.-H. Tung, "HEVC Early Termination Methods for Optimal CU Decision Utilizing Encoding Residual Information," *J. Real-Time Image Process.*, Jun. 2016, [in press].

[14] Y. Li, G. Yang, Y. Zhu, X. Ding, and X. Sun, "Adaptive Inter CU Depth Decision for HEVC Using Optimal Selection Model and Encoding Parameters," *IEEE Trans. Broadcast.*, vol. 63, no. 3, pp. 535–546, Sep. 2017.

[15] Y. Zhang, S. Kwong, X. Wang, H. Yuan, Z. Pan, and L. Xu, "Machine Learning-Based Coding Unit Depth Decisions for Flexible Complexity Allocation in High Efficiency Video Coding," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2225–2238, Jul. 2015.

[16] S. Momcilovic, N. Roma, L. Sousa, and I. Milentijevic, "Run-Time Machine Learning for HEVC/H.265 Fast Partitioning Decision," in *2015 IEEE International Symposium on Multimedia (ISM)*, Dec. 2015, pp. 347–350.

[17] H.-S. Kim and R.-H. Park, "Fast CU Partitioning Algorithm for HEVC Using an Online-Learning-Based Bayesian Decision Rule," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 130–138, Jan. 2016.

[18] G. Corrêa, P. Assunção, L. Agostini, and L. A. da Silva Cruz, "Fast Coding Tree Structure Decision for HEVC Based on Classification Trees," *Analog Integr. Circ. Sig. Process.*, vol. 87, no. 2, pp. 129–139, May 2016.

[19] S. Ahn, B. Lee, and M. Kim, "A Novel Fast CU Encoding Scheme Based on Spatiotemporal Encoding Parameters for HEVC Inter Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 422–435, Mar. 2015.

[20] Q. Hu, X. Zhang, Z. Shi, and Z. Gao, "Neyman-Pearson-Based Early Mode Decision for HEVC Encoding," *IEEE Trans. Multimedia*, vol. 18, no. 3, pp. 379–391, Mar. 2016.

[21] M. Tang, X. Chen, J. Gu, Y. Han, J. Wen, and S. Yang, "Accelerating HEVC Encoding Using Early-Split," *IEEE Signal Process. Lett.*, vol. 25, no. 2, pp. 209–213, Feb. 2018.

[22] J. Lee, S. Kim, K. Lim, and S. Lee, "A Fast CU Size Decision Algorithm for HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 411–421, Mar. 2015.

[23] J. Zhang, S. Kwong, and X. Wang, "Two-Stage Fast Inter CU Decision for HEVC Based on Bayesian Method and Conditional Random Fields," *IEEE Trans. Circuits Syst. Video Technol.*, 2017, [in press].

[24] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An Effective CU Size Decision Method for HEVC Encoders," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 465–470, Feb. 2013.

[25] W. Zhao, T. Onoye, and T. Song, "Hierarchical Structure-Based Fast Mode Decision for H.265/HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 10, pp. 1651–1664, Oct. 2015.

[26] Z. Liu, T.-L. Lin, and C.-C. Chou, "Efficient prediction of CU depth and PU mode for fast HEVC encoding using statistical analysis," *J. Visual Commun. Image Representation*, vol. 38, no. Supplement C, pp. 474–486, Jul. 2016.

[27] J.-H. Lee, K. Goswami, B.-G. Kim, S. Jeong, and J. S. Choi, "Fast Encoding Algorithm for High-Efficiency Video Coding (HEVC) System Based on Spatio-Temporal Correlation," *J. Real-Time Image Process.*, vol. 12, no. 2, pp. 407–418, Aug. 2016.

[28] Z. Liu, T.-L. Lin, and C.-C. Chou, "HEVC Coding-Unit Decision Algorithm Using Tree-Block Classification and Statistical Data Analysis," *Multimed. Tools Appl.*, vol. 76, no. 6, pp. 9051–9072, Mar. 2017.

[29] G. Corrêa, P. Assunção, L. V. Agostini, and L. A. da Silva Cruz, "Fast HEVC Encoding Decisions Using Data Mining," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 4, pp. 660–673, Apr. 2015.

[30] I. Zupancic, S. G. Blasi, E. Peixoto, and E. Izquierdo, "Inter-Prediction Optimizations for Video Coding Using Adaptive Coding Unit Visiting Order," *IEEE Trans. Multimedia*, vol. 18, no. 9, pp. 1677–1690, Sep. 2016.

[31] R. Rodríguez, J. L. Martínez, G. Fernández-Escribano, J. M. Claver, and J. L. Sánchez, "Accelerating H.264 Inter Prediction in a GPU by Using CUDA," in *2010 Digest of Technical Papers International Conference on Consumer Electronics (ICCE)*, Jan. 2010, pp. 463–464.

[32] R. Rodríguez, J. Martínez, J. De Cock, J. L. Sánchez, J. M. Claver, and R. Van de Walle, "Low Delay H.264/AVC Bidirectional Inter Prediction on a GPU," in *2013 IEEE International Conference on Image Processing*, Sep. 2013, pp. 2111–2115.

[33] J. Ma, F. Luo, S. Wang, N. Zhang, and S. Ma, "Parallel Intra Coding for HEVC on CPU Plus GPU Platform," in *2015 Visual Communications and Image Processing (VCIP)*, Dec. 2015.

[34] S. Radicke, J.-U. Hahn, Q. Wang, and C. Grecos, "A parallel hevc intra prediction algorithm for heterogeneous cpu+gpu platforms," *IEEE Trans. Broadcast.*, vol. 62, no. 1, pp. 103–119, Mar. 2016.

[35] W. Jiang, Y. Chi, H. Jin, X. Liao, Y. Zhang, and G. Ye, "A Fine-Grained Parallel Intra Prediction for HEVC Based on GPU," in *2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)*, Dec. 2016, pp. 778–784.

[36] S. Radicke, J.-U. Hahn, Q. Wang, and C. Grecos, "Bi-predictive Motion Estimation for HEVC on a Graphics Processing Unit (GPU)," *IEEE Trans. Consum. Electron.*, vol. 60, no. 4, pp. 728–736, Nov. 2014.

[37] F. Luo, S. Ma, J. Ma, H. Qi, L. Su, and W. Gao, "Multiple Layer Parallel Motion Estimation on GPU for High Efficiency Video Coding (HEVC)," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 1122–1125.

[38] W. Xiao, B. Li, J. Xu, G. Shi, and F. Wu, "HEVC Encoding Optimization Using Multicore CPUs and GPUs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 11, pp. 1830–1843, Nov. 2015.

[39] H. Igarashi, F. Takano, and T. Moriyoshi, "Highly Parallel Transformation and Quantization for HEVC Encoder on GPUs," in *2016 Visual Communications and Image Processing (VCIP)*, Nov. 2016.

[40] Y. Wang, X. Guo, Y. Lu, X. Fan, and D. Zhao, "GPU-Based Optimization for Sample Adaptive Offset in HEVC," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 829–833.

[41] D. F. de Souza, N. Roma, and L. Sousa, "Cooperative CPU+GPU deblocking filter parallelization for high performance HEVC video codecs," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 4993–4997.

[42] J. Garrett-Glaser, "A Novel Macroblock-Tree Algorithm for High-Performance Optimization of Dependent Video Coding in H.264/AVC," Department of Computer Science. Harvey Mudd College, Tech. Rep., 2009.

[43] "HEVC Test Model (HM) Reference Software," https://hevc.hhi.fraunhofer.de/, accessed on Jan. 28, 2018.

[44] F. Bossen, "Common Test Conditions and Software Reference Configurations," Tech. Rep. JCTVC-L1100, Jan. 2013.

[45] G. Cebrián-Márquez, C. C. Chi, J. L. Martínez, P. Cuenca, M. Álvarez Mesa, S. Sanz-Rodríguez, and B. Juurlink, "Reducing HEVC Encoding Complexity Using Two-Stage Motion Estimation," in *2015 Visual Communications and Image Processing (VCIP)*, Dec. 2015.

[46] G. J. Sullivan, J.-R. 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.

[47] J. Sole, R. Joshi, N. Nguyen, T. Ji, M. Karczewicz, G. Clare, F. Henry, and A. Dueñas, "Transform Coefficient Coding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1765–1777, Dec. 2012.

[48] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," ITU-T Video Coding Experts Group (VCEG), Tech. Rep. VCEG-M33, Mar. 2001.

[49] G. Cebrián-Márquez, J. L. Martínez, P. Cuenca, M. Tang, and J. Wen, "Accelerating HEVC Using GPU-based Heterogeneous Platforms," in *Proceedings of the 14th International Conference on Computational and Mathematical Methods in Science and Engineering*, vol. 1, Jul. 2014, pp. 288–299.

[50] G. Cebrián-Márquez, J. L. Hernández-Losada, J. L. Martínez, P. Cuenca, M. Tang, and J. Wen, "Accelerating HEVC Using Heterogeneous Platforms," *J. Supercomput.*, vol. 71, no. 2, pp. 613–628, Feb. 2015.

[51] G. Cebrián-Márquez, H. Migallón, J. L. Martínez, O. López-Granado, P. Piñol, and P. Cuenca, "GPU-Based Heterogeneous Coding Architecture for HEVC," in *Algorithms and Architectures for Parallel Processing. Lecture Notes in Computer Science*, vol. 10048. Springer International Publishing, Nov. 2016, pp. 529–536.

[52] G. Cebrián-Márquez, V. Galiano, H. Migallón, J. L. Martínez, P. Cuenca, and O. López-Granado, "Heterogeneous CPU Plus GPU Tile-Based Approach for HEVC," in *Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering*, vol. 2, Jul. 2017, pp. 534–545.

[53] G. Cebrián-Márquez, V. Galiano, H. Migallón, J. L. Martínez, P. Cuenca, and O. López-Granado, "Heterogeneous CPU Plus GPU Approaches for HEVC," *J. Supercomput.*, [Under review].

[54] A. J. Díaz-Honrubia, G. Cebrián-Márquez, J. L. Martínez, P. Cuenca, J. M. Puerta, and J. A. Gámez, "Low-Complexity Heterogeneous Architecture for H.264/HEVC Video Transcoding," *J. Real-Time Image Process.*, vol. 12, no. 2, pp. 311–327, Aug. 2016.

[55] G. Cebrián-Márquez, J. L. Martínez, and P. Cuenca, "A Pre-Analysis Algorithm for Fast Motion Estimation in HEVC," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 2013–2017.

[56] G. Cebrián-Márquez, J. L. Martínez, and P. Cuenca, "Two-Stage Intra Prediction Algorithm for HEVC," in *Proceedings of the 16th International Conference on Computational and Mathematical Methods in Science and Engineering*, vol. 2, Jul. 2016, pp. 334–345.

[57] G. Cebrián-Márquez, J. L. Martínez, and P. Cuenca, "Inter and Intra Pre-Analysis Algorithm for HEVC," *J. Supercomput.*, vol. 73, no. 1, pp. 414–432, Jan. 2017.

[58] G. Cebrián-Márquez, J. L. Martínez, and P. Cuenca, "Look-Ahead Partitioning Based on Motion Information for HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, [Under review].

[59] G. Cebrián-Márquez, J. L. Martínez, and P. Cuenca, "Fast Inter CU Partitioning Based on a Look-Ahead Stage for HEVC," *IEEE Trans. Multimedia*, [Under review].

[60] H. Migallón, J. L. Hernández-Losada, G. Cebrián-Márquez, P. Piñol, J. L. Martínez, O. López-Granado, and M. Malumbres, "OpenMP HEVC Parallel Version Based on a

GOP Approach," in *Proceedings of the Ninth International Conference on Engineering Computational Technology*, Sep. 2014, paper 24.

[61] H. Migallón, J. L. Hernández-Losada, G. Cebrián-Márquez, P. Piñol, J. L. Martínez, O. López-Granado, and M. Malumbres, "Synchronous and Asynchronous HEVC Parallel Encoder Versions Based on a GOP Approach," *Adv. Eng. Soft.*, vol. 101, pp. 37–49, 2016, Civil-Comp (part 2).

[62] G. Cebrián-Márquez, A. J. Díaz-Honrubia, J. De Praeter, G. Van Wallendael, J. L. Martínez, 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*, 2015, pp. 241–245.

[63] D. García-Lucas, G. Cebrián-Márquez, and P. Cuenca, "On the Capabilities of the Open-Source HEVC Codecs," in *Proceedings of the 16th International Conference on Computational and Mathematical Methods in Science and Engineering*, vol. 5, Jul. 2016, pp. 1375–1386.

[64] D. García-Lucas, G. Cebrián-Márquez, and P. Cuenca, "Parallelization and Performance Evaluation of Open-Source HEVC Codecs," *J. Supercomput.*, vol. 73, no. 1, pp. 495–513, Jan. 2017.

[65] D. García-Lucas, G. Cebrián-Márquez, A. J. Díaz-Honrubia, and P. Cuenca, "Rate-Distortion/Complexity Analysis of Video Compression with Capability beyond HEVC," in *Proceedings of the 17th International Conference on Computational and Mathematical Methods in Science and Engineering*, vol. 6, Jul. 2017, pp. 2135–2146.

[66] D. García-Lucas, G. Cebrián-Márquez, A. J. Díaz-Honrubia, and P. Cuenca, "Acceleration of the Integer Motion Estimation in JEM through Pre-Analysis Techniques," *J. Supercomput.*, [Under review].

[67] C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. J. Sullivan, "High Efficiency Video Coding (HEVC) Test Model 16. Improved Encoder Description Update 6," Tech. Rep. JCTVC-X1002, Jun. 2016.

[68] "HomerHEVC — The Open-Source H.265-HEVC Encoder under LGPL License," http://homerhevc.com/, accessed on Jan. 28, 2018.

[69] "DivX HEVC Community Encoder," http://labs.divx.com/divx265, accessed on Jan. 28, 2018.

[70] "Kvazaar HEVC Encoder," http://ultravideo.cs.tut.fi/, accessed on Jan. 28, 2018.

[71] "x265 HEVC Encoder/H.265 Video Codec," http://x265.org/, accessed on Jan. 28, 2018.