UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

PROYECTO FIN DE CARRERA

IMPROVED INTRA-PREDICTION FOR VIDEO CODING

Ingeniería de Telecomunicación

Marta Merlo Serrano

July 2014

IMPROVED INTRA-PREDICTION FOR VIDEO CODING

AUTOR: Marta Merlo Serrano.

TUTOR: Ebroul Izquierdo

PONENTE: José María Martínez Sánchez

Master Thesis

Multimedia and Vision Research Group

School of Electronic Engineering and Computer Science

Queen Mary University of London

July 2014

ABSTRACT

This thesis focuses on improving the HEVC (High Efficiency Video Coding) standard. HEVC is the newest video coding standard developed by the ITU-T Video Coding Experts Group (VCEG) and the ISO / IEC Moving Picture Experts Group (MPEG), as a successor to the popular state-of-the-art H.264/MPEG-4 AVC (Advanced Video Coding) standard. HEVC makes use of prediction to exploit redundancies in the signal and therefore achieve high compression efficiency. In particular, the Intra-Picture prediction block consists of predicting a block in the current frame using the reference information from neighbouring blocks in the same frame. It supports three different modes, the angular mode with 33 different directions, the planar mode and DC mode. HEVC is reportedly able to achieve in average more than 50% higher efficiency than H.264/MPEG-4 AVC, but this comes at the cost of very high computational complexity. The contributions of this thesis mainly consist in improvements to the Intra-Picture prediction block, with the goal of drastically reducing computational complexity and, at the same time achieving comparable compression efficiency as conventional HEVC.

In average, 16.5% encoding operations can be saved using the proposed approach at the cost of relatively small compression efficiency losses.

Keywords: High Efficiency Video Coding, Video Coding, Intra-Picture Prediction, DC prediction, Planar mode, Angular mode.

RESUMEN

Éste proyecto se va a centrar en mejorar el estándar HEVC (High Efficiency Video Coding). HEVC es el estándar de codificación de video más reciente desarrollado por el UIT-T Video Coding Experts Group (VCEG) e ISO/IEC Moving Picture Experts Group (MPEG), siendo sucesor del popular estado del arte H.264/MPEG-4 AVC (Advanced Video Coding) estándar . HEVC hace uso de la predicción para aprovechar las redundancias en las señales y por lo tanto conseguir una alta eficiencia de compresión. En particular, el bloque Intra-Picture prediction consiste en predecir un bloque en el cuadro actual, utilizando información de referencia de bloques vecinos en el mismo cuadro. Soporta tres modos distintos, el modo angular con 33 diferentes direcciones, el modo Planar y el modo DC. HEVC es suficientemente capaz de lograr de media una eficiencia mayor del 50% que H.264/MPEG-4 AVC, a costa de una alta complejidad computacional. Las aportaciones a esta tesis consisten principalmente en mejoras en el bloque Intra-Picture prediction, con el objetivo de reducir drásticamente la complejidad computacional y a la vez, lograr una eficiencia de compresión comparable al HEVC convencional.

En promedio, un 16.5% de las operaciones de codificación pueden evitarse usando el enfoque propuesto a costa de pérdidas relativamente pequeñas de la eficiencia de compresión.

Palabras Clave: High Efficiency Video Coding, Video Coding, Intra-Picture Prediction, DC prediction, Planar mode, Angular mode.

ACKNOWLEDGEMENTS

A period of my life, full of great times and some difficulties, is finishing. Without it I would not have gotten where I am today, the end of my degree. So first I want to thank Ebroul for giving me the possibility to make this project in London, which was an unforgettable and enriching experience. And the people form MMV and Antennas for the warm welcome. And above all, I would like to thank Chema for giving me this wonderful opportunity and supporting me during this long path.

I especially want to mention and thank Saverio for all the help he provided me, without whom this project would have never been possible.

Also thank my teachers which over the years of my degree have given me the needed knowledge for a future that is already immediate. I do not forget to stand out their kindness and dedication at all times and their help whenever I needed it.

Principally, I would like to thank my family for all their support from the very first moment I made my decision. My parents have taught me to be tenacious and to fight for what I want in life. My grandparents have been a role model to follow with their wise advices. And my sister has been there any time I needed her.

Finally, I would like to mention all my friends from my degree; I met amazing people who made my day a day all worth it. It is impossible to name them all, so I can only thank them for every single moment we lived together. It is now time to give a special mention to Eva and Maya for being always by my side, you mean a lot to me. To all my friends in London, especially Laura and Sunny, for all the fun we shared, without them it would have never been the same. To my friend Ali for her unconditional love since we were kids. But mostly I want to thank Pablo for cheering me up on the hard moments and make me even happier on the good ones; I would never change the time we have spent together.

All the effort has worth it. Thank you all.

Marta Merlo Serrano

July 2014

AGRADECIMIENTOS

Se cierra una etapa de mi vida, llena de momentos buenos y otros difíciles sin los cuales no habría llegado a donde estoy hoy, al final de mi carrera. Por ello primero quiero dar las gracias a Ebroul por darme la posibilidad de realizar este proyecto en Londres, una experiencia que ha sido inolvidable y enriquecedora. A los grupos del MMV y Antenas por aceptarme con cariño. Y sobre todo dar las gracias a Chema por otorgarme esta maravillosa oportunidad y el apoyo recibido durante este largo camino.

Quiero mencionar especialmente a Saverio y agradecerle toda la ayuda que me ha proporcionado, sin él este proyecto no habría sido posible.

Asimismo dar las gracias a mis profesores que durante los años de carrera me han aportado el conocimiento necesario para un futuro que ya, es inmediato. Sin olvidar destacar su amabilidad y dedicación en todo momento y su ayuda siempre que la he necesitado.

Me gustaría agradecer fundamentalmente, a mi familia por todo el apoyo brindado desde el minuto que decidí embarcarme en esta decisión. A mi madre y a mi padre por enseñarme a ser tenaz y a luchar por lo que uno quiere. A mis abuelos por ser siempre un ejemplo a seguir con sus sabios consejos. Y a mi hermana, por estar ahí siempre que la he necesitado.

Finalmente, mencionar a todos mis amigos de la carrera, personas extraordinarias que hacían que mi día a día en la universidad valiera la pena, es imposible nombrarlos a todos así que sólo me queda darles las gracias por cada uno de los buenos momentos que hemos vivido. Mención especial a Eva y Maya por estar siempre a mi lado, significáis mucho para mí. A todos mis amigos de Londres, en especial a Laura y Sunny, por todas las risas compartidas, sin vosotros aquello no habría sido lo mismo. A mi amiga incondicional Ali por tu cariño desde pequeñas. Pero sobre todo quiero dar las gracias a Pablo, por alegrarme en los momentos difíciles y hacerme aún más feliz en los buenos, esta etapa que hemos vivido juntos no la cambiaría por nada.

Todo el esfuerzo ha merecido la pena. ¡Gracias!

Marta Merlo Serrano

Julio 2014

CONTENTS

FIGURE INDEX

TABLE INDEX

1.

INTRODUCTION

1.1. Motivation

Digital Video Coding is used to compress the information so it can be stored or transmitted occupying minimal space. This process is done in a video encoder and the main goal for them is to optimize coding efficiency. Coding efficiency is the ability to minimize the bit rate necessary for representation of video content while maintaining a given level of video quality.

The two previous video coding standards, the H.262/MPEG-2 Video [27] and H.264/MPEG-4 AVC (Advanced Video Coding) [11], have had a significant influence in a variety of products that are frequently present in our daily lives. Nevertheless, the growing popularity of HD video, the emergence of beyond-HD formats (4k, 8k), the need of higher resolution 3D or multiview, and the increased desire for higher quality and resolutions in mobile applications, are generating emergency needs for coding efficiency superior to H.264/MPEG-4 AVC’s capabilities. In addition, more than 50% of the current network traffic is video targeted to mobile devices and tablet-PCs; this growth in traffic and the transmission needs for video-on-demand services, are imposing harsh challenges in today’s networks.

For those reasons and mainly due to the increased need for higher compression of video content, the H.265/HEVC (High Efficiency Video Coding) [15] standard has been developed. This Recommendation addresses all existing applications of H.264/MPEG-4 AVC and also many new applications such as those aforementioned. It has been designed to pay special attention to three key issues: increased video resolution, ease of integration of the transport system and increased use of parallel processing architectures.

Thanks to many new improvements and an optimised coding scheme, the HEVC standardization effort is capable to improve the compression efficiency by reportedly reducing by an average 50% the bit-rate for the same objective and perceptual video quality with respect to its predecessor H.264/MPEG-4 AVC [7]. This comes at the cost of a generally much higher computational complexity with respect to previous standards.

1.2. Objectives and approach

The main goal of this project is to reduce the computational complexity while at the same time achieving comparable compression efficiency as conventional HEVC. To acquire this, the Intra-Picture prediction block has been improved.

Therefore, first step is to analyse the conventional HEVC Intra-Picture prediction block, which consists of predicting a block in the current frame using the reference information from neighbouring blocks in the same frame. It supports three different modes, the angular mode with 33 different directions, the planar mode and DC mode.

Then, second step is to design of a framework to test and develop the HEVC intra methods replicating the procedure as found in the HEVC reference software.

And finally, third step is to use this framework to improve the HEVC intra methods using binary classification of some features extracted from the frames. A Comparison between both frameworks will be performed in terms of BD-rate and elapsed time. To test those designs, 24 images and several videos have been used.

1.3. Report structure

This report is structured as follows:

1. **Introduction:** In this first chapter is explained the motivation, specifically why this new standard has been developed, and also is explain the objectives and approach of this project.

2. **State of the art:** In this part is explained the fundamentals of video coding to understand the second part that explains the state of the art that now is the High Efficient Video Coding standard. And the third part of this chapter consists in explain in detail the Intra-prediction method.

3. **Design and development:** This part introduces the problem formulation and the HEVC framework design as well as the improvement proposed for the intra-prediction block.

4. **Experiments and Results:** This section shows the results for all the experiments carried out during the master thesis, that consists in evaluate the different combinations of the features for 24 images and evaluate the quality of the proposed approach framework also for 2 videos.

5. **Conclusions and future work:** Conclusions and future work are presented in detail.

2.

STATE OF THE ART

2.1. Fundamental of Video Compression

2.1.1. Video: Digital images and video signals

Digital video data consists of a time-ordered sequence of a natural or real-world visual scene in digital form, sampled spatially and temporally. A scene is a sampled at a point in time to produce a complete frame or an interlaced field (namely one field consists of half of the data in a frame, spatially sampled at odd- or even-numbered interval lines) [6].

Spatial sampling: it means to take samples of the signal at a concrete point in time using a rectangular grid in the video image plane producing a frame. At each intersection point on the grid is taken a sample on spatial domain and it represents a square picture element (pixel), as shown in Figure 1,

Temporal sampling: it means to capture a series of frames sampled at periodic intervals in time producing a moving video signal. Cameras typically generate approximately 24, 25 or 30 frames per second.

This results in a large amount of information that demands the use of **compression**.

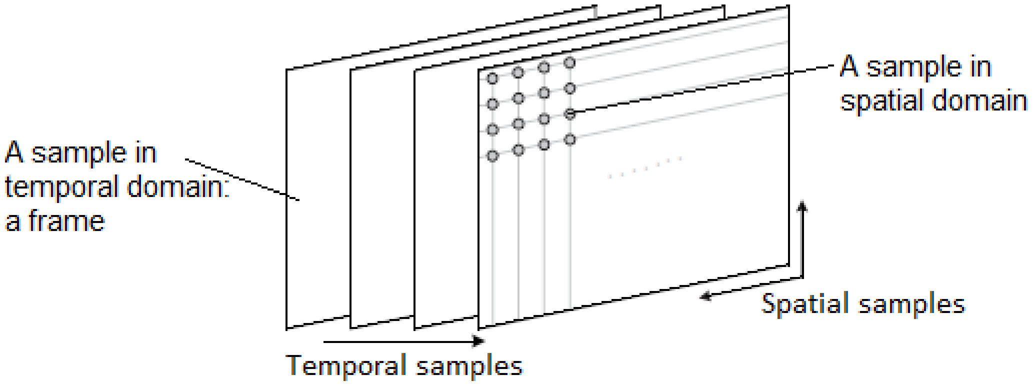

A sample in spatial domain
A sample in temporal domain: a frame
Spatial samples
Temporal samples

Figure 1: Spatial and temporal sampling of a video sequence.

2.1.2. Colour space

The visual information at each spatio-temporal sample (picture element or pixel) must be described in digital form by means of a number that depict the physical appearance of the sample. For a scene or a picture in colour, it is required at least three colour components samples (three different numbers) to represent a colour space accurately for each pixel position. This happens by means of colour spaces, namely models to map physical colours to measurable numeric expressions. Two well-known colour spaces are described here:

A. RGB

In the RGB colour space, a colour image sample is represented with three numbers that indicate the relative proportions of Red (R), Green (G) and Blue (B) (the three additive primary colours of light). Colours can be created by combining red, green and blue in varying proportions. These three colours are equally important and so are usually all stored at the same resolution.

B. YCbCr

In the YCbCr colour space (sometimes referred to as YUV), a colour image sample is represented with three numbers; one component indicate the brightness (luminance or luma) whereas the other two components indicate the colour. Due to the fact that human visual system is more sensitive to luminance than to colour, sometimes the resolution of the chroma components is down-sampled with respect to the resolution of the luma component to reduce the amount of information needed to describe each pixel. This colour space with chroma subsampling is an efficient way of representing colour images. The YCbCr colour space can be converted to RGB by means of simple expressions.

Y is the **luminance** (luma) component, and indicates the brightness in an image, and can be calculated as a weighted average of R, G and B:

$$\boxed{Y = k_r R + k_g G + k_b B} \tag{1}$$

where *k* are weighting factors. The colour information can be represented as **colour difference** (chrominance or chroma) components, where each chrominance component is the difference between R, G or B and the luminance Y:

$$\boxed{\begin{array}{c} C_b = B - Y \\ C_r = R - Y \\ C_g = G - Y \end{array}} \tag{2}$$

The complete description of a colour image is given by *Y* (the luminance component) and three colour differences *Cb, Cg* and *Cr* that represent the difference between the colour intensity and the mean luminance of each image sample. *Cg* is not transmitted because it is possible to extract it from the other components [6].

- <u>YCBCR SAMPLING FORMATS</u>

Chroma subsampling can be performed in different ways. The most common formats of sampling the images to obtain the three components are:

4:4:4 sampling: this means that each chroma component (Cb, Cr) have the same resolution as luma component (Y), preserving the full fidelity of the chrominance components.

4:2:2 sampling: this means that the chroma components have half the horizontal resolution of luma component. For every four luminance samples in the horizontal direction there are two Cb and two Cr samples. 4:2:2 video is used for high-quality colour reproduction.

4:2:0 sampling: this means that each chroma component has one fourth of the number of samples of the luma component (half the number of samples in both the horizontal and vertical dimensions). 4:2:0 sampling is widely used for consumer applications such as video conferencing, digital television and digital versatile disk (DVD) storage.

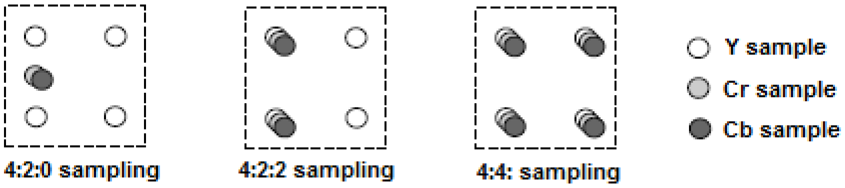


Figure 2: A sample of the YCbCr Sampling Formats.

Representing chroma with a lower resolution than luma in this way is a simple but effective form of image **compression**.

2.1.3. Video Coding

Video coding schemes are formed of two main entities, an encoder and a decoder. The video encoders encode the input video signals (variations of an electrical signal that represents a video image) to its compressed binary representation (0s or 1s signal), and the decoders decode such representation to an output video signal ready for consumption. This process is called **video coding**. [6]

The process of compressing a digital video signal into a smaller number of bits is called **video compression.** So, the typical video coding scenario involves a complementary pair of systems, where a compressor (encoder) compresses or encodes a source video signal into a coded bit-stream, and a decompressor (decoder), which decompresses or decodes the coded bit-stream back into an output video signal similar to the original video data. The encoder or decoder is often built in to a device such as a video camera or a DVD player. For example, the video cameras have limited space in their memory, so capturing a scene with it and using the built-in video encoder results in a compress information that allow to store as much video as possible in this limited storage.

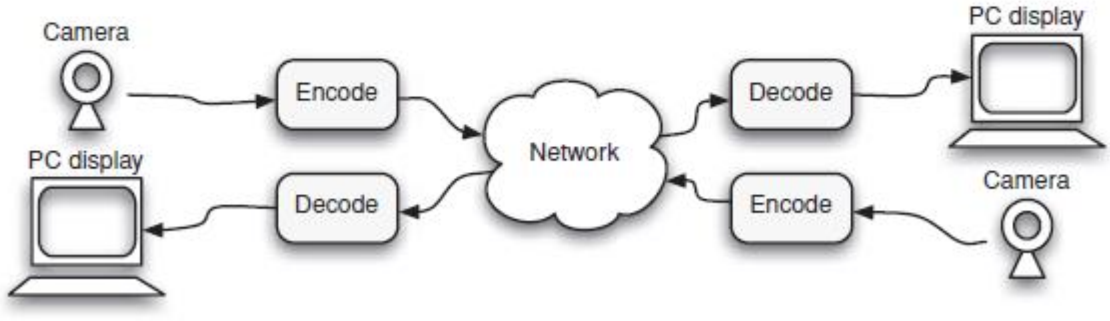


Figure 3: Video coding scenario, two-way.

To reach the data compression, video encoders takes advantage of the fact that video sequences have a large amount of **redundancy** in temporal domain (inter-frame correlation) and spatial domain (intra-frame correlation) [8]. Redundancy happens in a message in case there are predictable or redundant parts of the message; therefore some parts of the message do not provide new information but instead “repeat” some of the information already contained in previous parts. By removing the redundant information in the different domains it is possible to compress the data significantly [22] [24].

There are two different types of compression:

- **Lossless compression** that achieve moderate compression, but the decoded signal is an identical reconstruction of the original signal. This type of compression exploits the ***statistical redundancy***, which means for *still image* that neighbouring pixel values within an image are often highly correlated (***spatial redundancy***) and for *video data* it is also add the fact that the value of pixels at the same positions across neighbouring frames in time are also highly correlated (t***emporal redundancy***) [23].

- **Lossy compression** that achieves higher compression at the expense of a certain amount of irreversible information loss (distortion) of visual quality, so the decoded signal is similar but not equal to the original source data. This type of compression exploits the ***subjective redundancy*** due to the problem of perception, which refers to the fact that the human visual system is not sensitive to certain components (usually high frequency) of the visual information. For this reason these components can be removed without causing hard degradation in the subjective quality of the decoded video signal.

This last type of compression is often used for the compression of speech, audio, picture, and video signals, where huge compression ratios are usually needed and at the same time an exact reconstruction of the source data may not be required. Most video coding standards (including HEVC) make use of lossy compression schemes to achieve very high compression efficiency

2.1.4. Video CODEC

A CODEC is an encoder-decoder pair (enCOder/DECoder). Most video codecs are based on a very well-known scheme referred to as the hybrid model in which the signal is processed both in the spatial and frequency domain.

A video encoder following the hybrid model consists of taking a sequence of video frames (video source) and **partitioning** each picture into multiple blocks that are independently processed. Each block is first **predicted** in the spatial domain, using inter or intra prediction (as detailed in the rest of this chapter). The residual signal (the difference between the original picture and the prediction) is **transformed** into another domain in which the samples are represented by transform coefficients; these coefficients are **quantized** to remove insignificant values. And finally, the quantized transform coefficients, prediction information, mode information and headers (essential information placed at the beginning of a block of data) are compressed by the **entropy encoder**. These steps are performed sequentially in a video encoder to create a compressed video bit-stream; this bit-stream is stored or transmitted [5].

A video stream will contain some intra frames and between this frames the information needed to transform the previous frame into the next frame

On the other hand, the bit-stream is decompressed in the video decoder to create a copy or approximation of the source sequence (decoded frames). The coefficients and prediction parameters are decoded by an entropy decoder after which the transform output is rescaled and inverted to obtain the residual frame. Finally, the decoder reconstructs the frame by adding the residual frame to a prediction; this prediction usually is created in the decoder using motion vectors, together with previously decoded image samples [11].

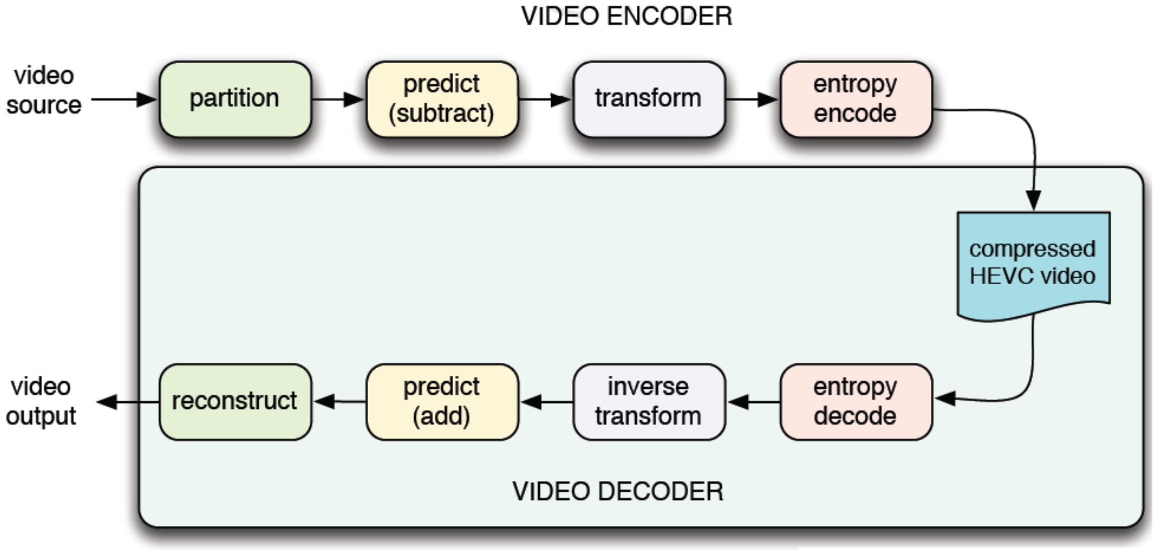


Figure 4: General structure of an encoder and decoder.

A. Intra-frame prediction

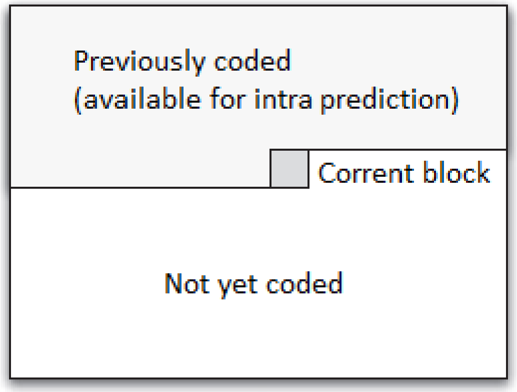
In the spatial domain, the redundancy means that pixels (samples) that are close to each other in the same frame or field are usually highly correlated. This means that the appearance of samples in an image is often similar to their adjacent neighbour samples; this is called the **spatial redundancy** or **intra-frame correlation**.

This redundant information in the spatial domain can be exploited to compress the image. Note that when using this kind of compression, each picture is compressed without referring to other pictures in the video sequence. This technique is called **Intra-frame prediction** and it is designed to minimize the duplication of data in each picture (spatial-domain redundancy) [1]. It consists in forming a prediction frame and subtracting this prediction from the current frame.

Figure 5: Spatial (intra-frame) correlation in a video sequence.

Several methods can be used to remove this redundant information in the spatial domain. Typically the values of the prediction samples are constructed by combining their adjacent neighbour samples (reference samples) by means of several techniques (as detailed in the rest of this chapter). In some cases, considerably prediction accuracy can be obtained by means of efficient intra-prediction techniques.

The image is partitioned in blocks and for each block the prediction is created by extrapolating samples from previously-coded samples in the same frame, as shown in Figure 6. In order for this process to be exactly replicable also at the decoder side, only the pixels along the upper and/or left edges can be used to create the prediction block, because these were previously encoded in the bit-stream and therefore they can be used to re-create the prediction, as shown in Figure 7 [11].



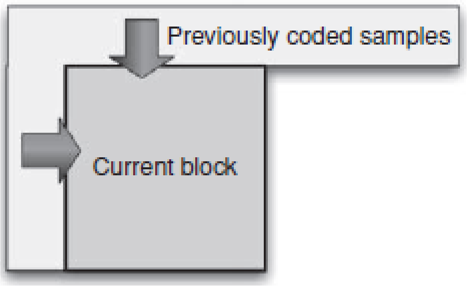


Figure 6: Intra prediction: available samples

Figure 7: Intra prediction: spatial extrapolation.

Once the prediction has been generated, it is subtracted from the current block to form a residual signal. The residual signal is transformed and encoded, together with an indication of how the prediction was generated.

B. Inter frame prediction

In the temporal domain, redundancy means that successive frames in time order are usually high correlated; therefore parts of the scene are repeated in time with little or no changes. This type of redundancy is called **temporal redundancy** or **inter-frame correlation** [21].

It is clear then that the video can be represented more efficiently by coding only the changes in the video content, rather than coding each entire picture repeatedly. This technique is called **Inter-frame prediction**; it is designed to minimize the temporal-domain redundancy and at the same time improve coding efficiency to achieve video compression [1].

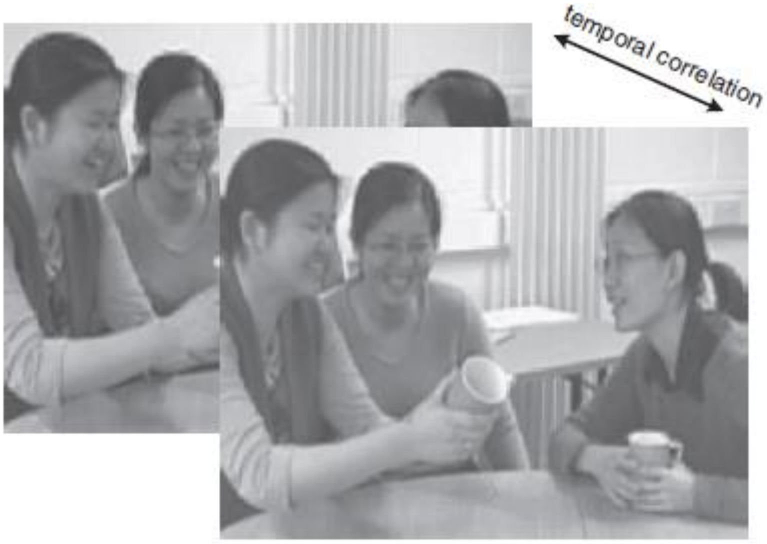
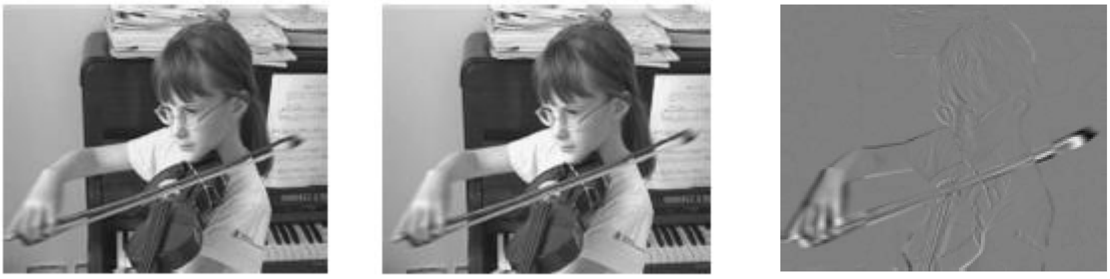


Figure 8: Temporal (inter-frame) correlation in a video sequence.

To remove the redundant information in the temporal domain typically **motion compensated prediction** or **inter prediction** methods are used. Motion compensation (MC) consists of constructing a prediction of the current video frame from one or more previous or future encoded frames (reference frames) by compensating differences between the current frame and the reference frame. To achieve this, the motion or trajectory between successive blocks of the image is estimated. The information regarding motion vectors (describes how the motion was compensated) and residuals from the previous frames are coded and sent to the decoder.

Figure 9 shows two successive frames from a video sequence and the difference between them that is the residual. The light and dark area in the residual frame indicate the energy that remains in the residual frame, this means that there is still a significant amount of information to compress, due to the movements of the objects between the two frames. More efficient compression can be achieved by compensating the movements between the two frames.



Frame 1 Frame 2 Residual: no MC

Figure 9: Residual: differences between frame 1 and 2.

Less information is sent when coding the changes in the video content, so the information will be more compressed. It is possible to estimate the trajectory of each sample in the block between successive video frames producing an optical flow, as shown in Figure 10. An optical flow consists of all the motion vectors which indicate the direction of the trajectory of the movement between each block in a frame and their best match in a previously encoded frame. The motion compensation method is applied for each block of the current frame to compensate for this movement. So, it should be possible to form an accurate prediction of most samples in the block of the current frame by translating each sample from the reference along its motion vector.

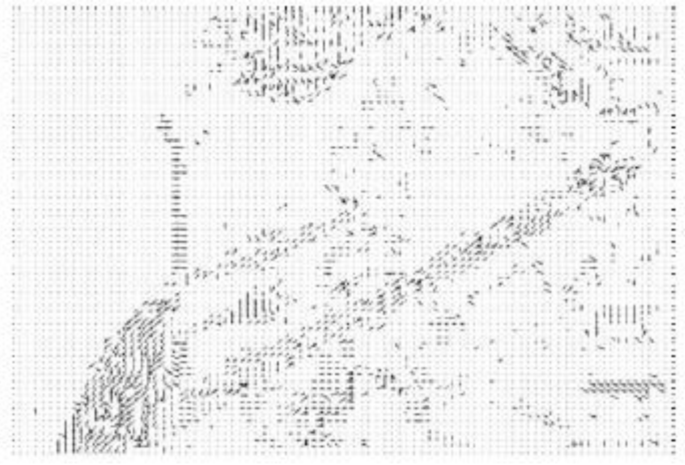


Figure 10: Optical flow: motion vectors.

To obtain the motion vector and the motion compensation, the following procedure is carried out for each MxN block in the current frame, where M and N are the block height and width respectively:

1. The first step is called **Motion Estimation (ME)** and it consists in finding the best spatial displacement approximation in a previously encoded reference frame between an MxN block extracted from the reference, and the current block. A region centred on the current block position of the reference frame is localized (referred to asthe **search area).** Then each possible MxN block in the search area is compared with the MxN current block in terms of a certain matching criterion. The block at a given displacement that minimizes the matching criterion is chosen as the best match, as shown in Figure 11. This spatial displacement offset between the position of the candidate block and the current block is the **motion vector (MV)**.

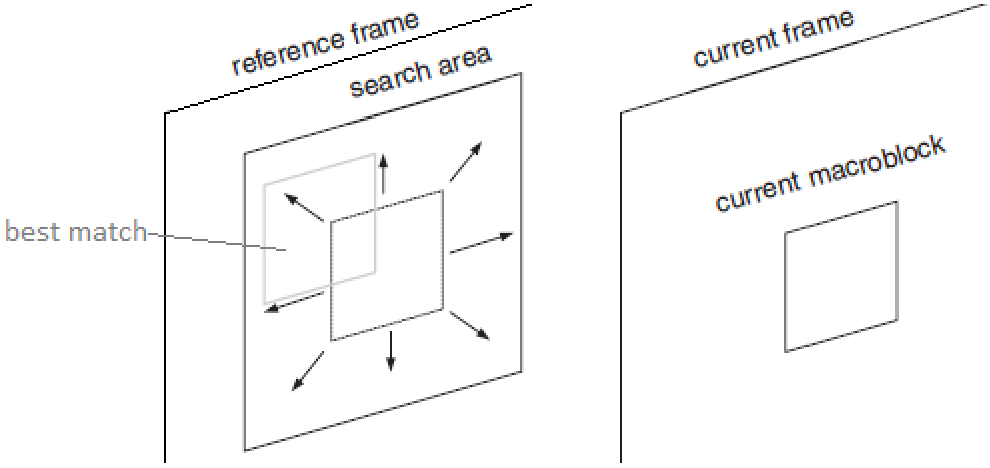


Figure 11: Motion Estimation.

A popular matching criterion is the energy in the residual formed by subtracting the candidate region from the current MxN block, so that the candidate region that minimises the residual energy is chosen as the best match. The energy of the residual block may be defined as Sum of Absolute Differences (SAD) or Mean Square Error (MSE), which are the most popular energy definitions:

Sum of Absolute Difference
$$\boxed{SAD = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left|C_{i,j} - R_{i,j}\right|} \quad (3)$$

Mean Square Error:
$$\boxed{MSE = \frac{1}{M \cdot N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(C_{i,j} - R_{i,j}\right)^2} \quad (4)$$

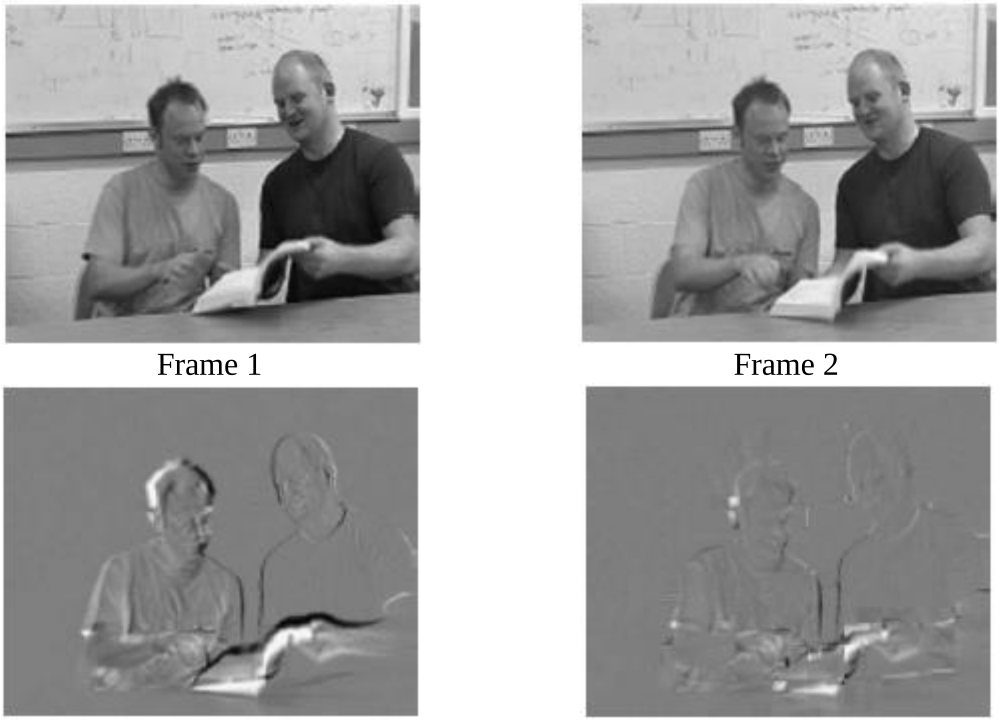
Where $C_{i,j}$ is the current area samples and $R_{i,j}$ is the reference area samples.

2. The second step is known as **Motion compensation (MC)**, which consists in taking the optimal motion vector found in the previous step, and applying it to the reference frame to obtain the motion compensation prediction for the current block.

3. The third step is to encode and transmit the residual block and the motion vectors.

On the other side, the decoder uses the received motion vector to re-create the candidate region. This is added tothe decoded residual block, to reconstruct a version of the original block.

The next example in Figure 12 shows two frames (referred to as frame 1 and frame 2), the residual signal obtained subtracting frame 1 and frame 2 without motion compensation, and the energy in the residual signal obtained after motion compensating each 16x16 block in the frame. It is clear that the use of motion compensation can greatly reduce the amount of information to be transmitted [11].



Frame 1

Frame 2

Residual: no MC

Residual with MC (16x16 blocks)

Figure 12: Comparison between the use of MC or not.

It should be noted that it is more efficient to use Intra prediction than Inter prediction in case a significant change between the current frame and the reference frame occurs. It should be noted that close objects in a video scene may move by a different number of pixel positions between frames, so it may be more efficient to use a variable block size for motion estimation and compensation to capture these different behaviours.

A better prediction may also be formed using sub-pixel motion estimation and compensation. It involves interpolating the reference frame to sub-pixel positions as well as integer-pixel positions before choosing the position that gives the best match and minimizes the residual energy.

The Figure 13 shows the concept of quarter-pixel motion estimation. The motion estimation starts at the integer pixel grid. After the best integer-precision match is found, a new search starts at half-pixel positions; finally the search is refined at quarter-pixel position. The final match, at integer, half-pixel or quarter-pixel position is used for motion compensation.

Interpolation at sub-pixel precision produces a smaller residual (fewer bits to encode it) at the expense of higher computational complexity. The use of sophisticated interpolation filters improves the efficiency of sub-pixel interpolation.

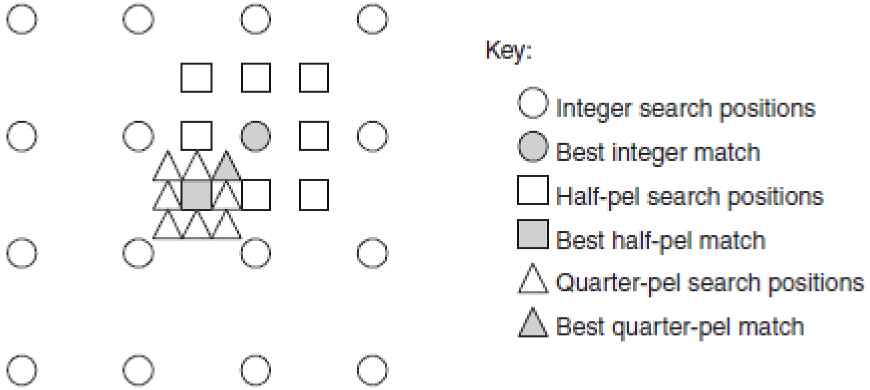


Figure 13: Integer, half-pixel and quarter-pixel motion estimation.

C. Transform and quantization

The residual signal output from the intra or inter prediction mode is transformed into frequency domain in which the samples are represented by transform coefficients in order to reduce spatial redundancy. The transform domain provides a more compact way of representing the visual information. The transform method being used should be reversible, and should decorrelate the signal as much as possible (i.e. separated into components within minimal inter-dependence) and it should be easily computed. One of the most popular transform methods used at this purpose is the Discrete Cosine Transform (DCT). DCT is a block-based transform. [20].

The transform coefficients are quantized (to compress a range of values to a single quantum value) to remove less significant values (usually high frequency content) without seriously affecting the visual quality of the image. In this way the quantized signal can be represented using fewer bits than the original. The output is a set of quantized transform coefficients.

D. Entropy encoder

The motion vector with integer or sub-pixel resolution, quantized transform coefficients, headers information and supplementary information can be efficiently encoded/compressed using entropy encoding. This removes statistical redundancy in the data, taking advantage of the relative probabilities of the various possible values for each sample, and produces a compressed bit stream suitable for transmission or storage.

2.2. High Efficiency Video Coding (HEVC)

The **H.265/High Efficiency Video Coding (HEVC) standard** is the most recent joint effort of the ITU -T Video Coding Experts Group (VCEG) and the ISO / IEC Moving Picture Experts Group (MPEG) standardization organizations. HEVC is a new international standard for video compression; a successor to the H.264/MPEG-4 AVC (Advanced Video Coding) standard. The first edition of the HEVC standard was completed and published in early 2013, though several extensions to the technology remain under active development [4].

HEVC is reportedly particularly effective for high resolution video content, low bit rates and low-delay communications applications [2]. It is able to reduce on average 50% the bit-rate at about the same perceptual and objective video quality with respect to its predecessor [18].

2.2.1. Diagram blocks

The general structure of HEVC is very similar to previous standards, and it is based on the hybrid video-coding model. Such model encodes the video data using a hybrid combination of motion compensated prediction (in the spatial domain) and transform coding (in the frequency domain).The next figure represents the block diagram of a hybrid video encoder to generate a bit-stream conforming to HEVC.

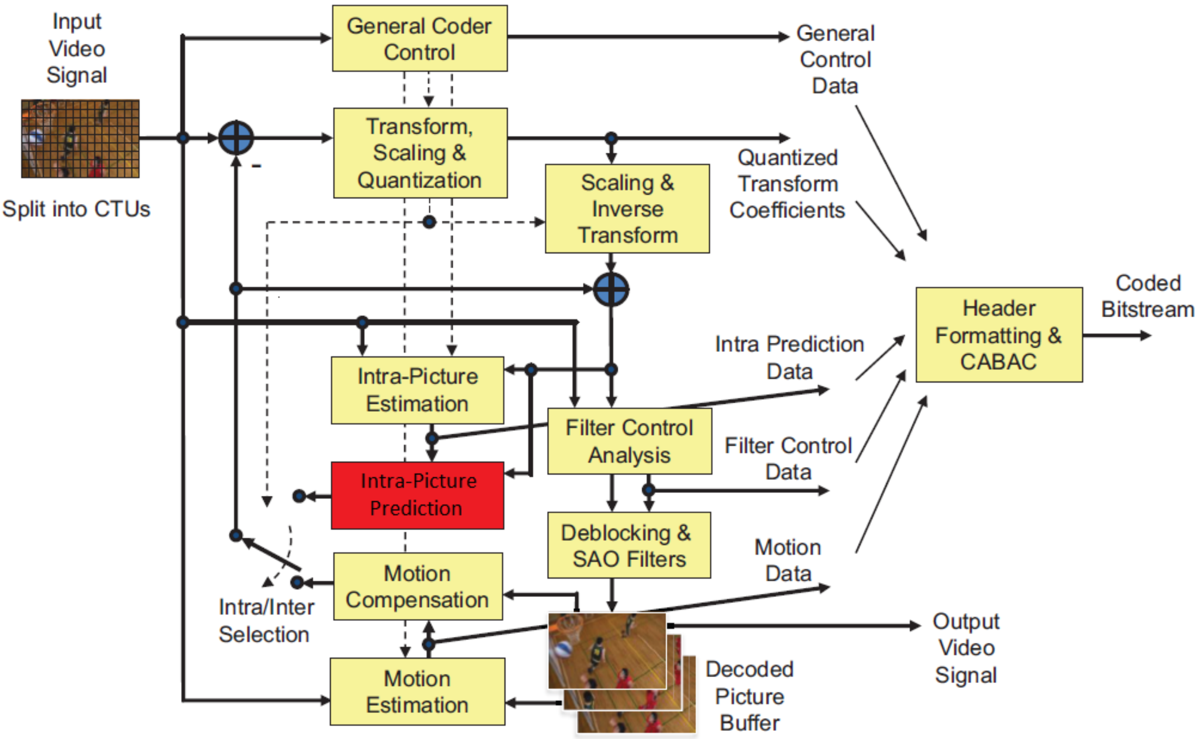


Figure 14: Typical HEVC video encoder

In this report only the encoder scheme is detailed due to the fact that during the encoding scheme, an identical process to the decoding is performed. Encoder and decoder will create identical predictions for subsequent data, which means that the encoder replicates the decoder processing.

2.2.2. Sampled Representation of Pictures

HEVC uses the colour space called **YCbCr** for representing colour video signals. This colour space is composed of three components called luminance for **Y**, blue chrominance for **Cb** and red chrominance for **Cr**. The component Y indicates the brightness in an image, Cb indicates the difference between blue and luma (B-Y), and the other component Cr indicates the difference between red and luma (R-Y). HEVC uses 8 bits precision to represent input and output data, although extensions to the standard are currently under development to support higher bit depths.

Human visual system is more sensitive to luma than chroma components, so for this reason the first version of HEVC only supports a 4:2:0 chroma subsampling. It means that each chroma component has one fourth of the number of samples of the luma component. Support for other chroma subsampling schemes such as 4:2:2 and 4:4:4 will be defined in subsequent versions of HEVC [6].

2.2.3. Slices and Tiles

A picture is split in one or several **slices**, which are sequences of an integer number of CTUs (Coding Tree Units, explained in the next section), as shown in Figure 15. CTUs in a slice are processed in raster scan order. A slice may contain a variable number of CTUs. The main purpose of slices is resynchronization after data losses [3]. Slices are used for controlling packet sizes.

Each slice can be coded using different coding types as follows:

- **I slice:** All CUs (Coding Units, explained later in this chapter) of the slice are encoded using intra-picture prediction. They can be reconstructed without any reference to other frames.
- **P slice:** Some CUs are encoded using inter-picture prediction with one motion-compensated prediction signal per block. P slices only use reference picture list 0.
- **B slice:** Some CUs are encoded using inter-picture prediction with two motion-compensated prediction signals per block. B slices may use reference picture list 0 or list 1 or both.

In addition, a new feature was introduced in HEVC that splits a picture into rectangular arranged group of CTUs, as shown in Figure 16, defined as **tiles**. A single tile may contain multiple slices. Note that tiles are always aligned with CTB boundaries.

Tiles are used for parallel processing architectures for encoding and decoding [14]; different tiles can be encoded in parallel.

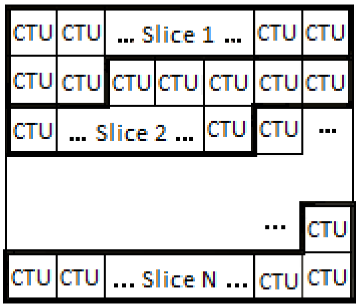


Figure 15: Subdivision of a picture into slice.

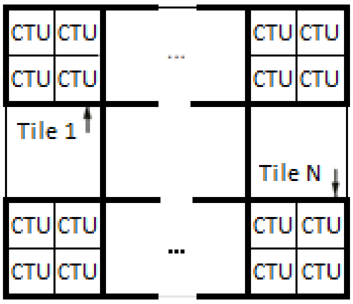


Figure 16: Subdivision of a picture into tiles.

2.2.4. Picture partitioning

The previous standards split the pictures in block-shaped regions called Macroblocks and Blocks. Nowadays we have high-resolution video content, so the use of larger blocks is advantageous for encoding. To support this wide variety of blocks size in efficient manner HEVC pictures are divided into so-called **coding tree Units (CTUs).**

Depending on the stream parameters, the CTUs in a video sequence can have the size: 64×64, 32×32, or 16×16 [3].

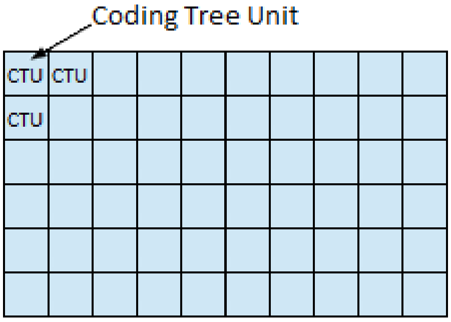


Figure 17: Frame split in CTUs

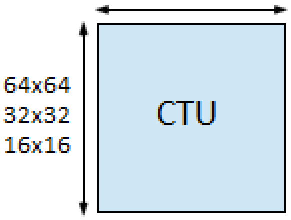


Figure 18: Possible sizes of the CTUs

Coding Tree Unit (CTU) is therefore a coding logical unit, which is in turn encoded into an HEVC bit-stream. It consists of three blocks, namely *luma (Y)*, that covers a square picture area of LxL samples of the luma component, and two *chroma components (Cb and Cr)*, that cover L/2xL/2 samples of each of the two chroma components, and *associated syntax elements*. Each block is called **Coding Tree Block (CTB)** [2].

Syntax elements describe properties of different types of units of a coded block of pixels and how the video sequence can be reconstructed at the decoder. This includes the method of prediction (e.g. inter or intra prediction, intra prediction mode, and motion vectors) and other parameters [9].

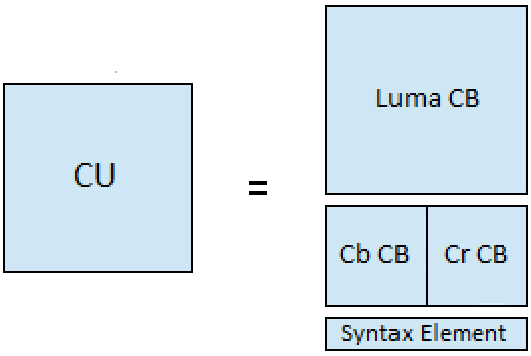


Figure 19: CTU consist of three CTBs and Syntax Elements

Each CTB has the same size (LxL) as the CTU (64×64, 32×32, or 16×16). However, CTB could be too big to decide whether we should perform inter-picture prediction or intra-picture prediction. So each CTB can be split recursively in a quad-tree structure, from the same size as CTB to as small as 8×8. Each block resulting from this partitioning is called **Coding Blocks (CBs)** and becomes the decision making point of prediction type (inter or intra prediction) [16].

The following picture illustrates an example of how 64×64 CTB can be split into CBs.

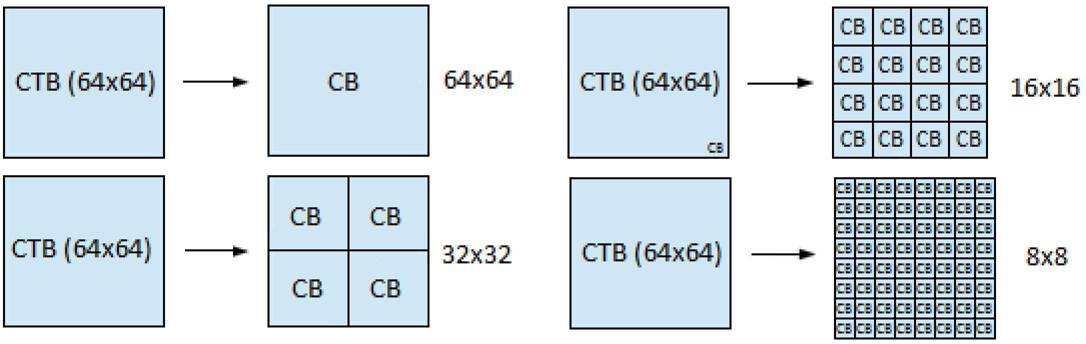


Figure 20: How 64×64 CTB can be split into CBs

Or another example could be:

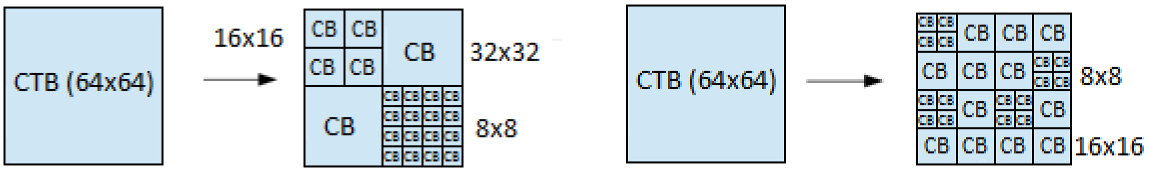


Figure 21: Another example of splitting the CTB into CBs

The prediction type along with other parameters is coded in **Coding Unit (CU)**. So CU is the basic unit of prediction in HEVC, each of which is predicted from previously coded data. And the CU consists of three CBs (*Y*, *Cb* and *Cr*), and *associated syntax elements*.

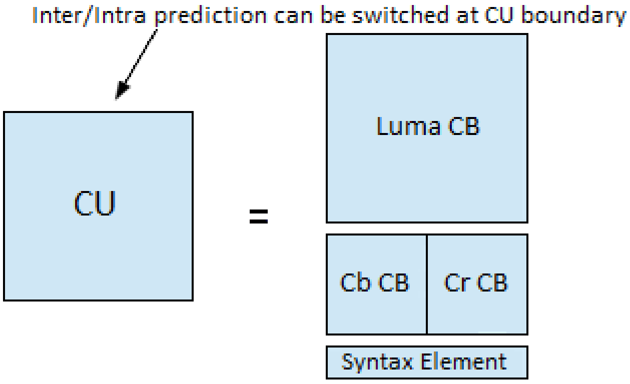


Figure 22: CU consist in three CBs and associated syntax elements

CBs could still be too large to store motion vectors (inter-picture (temporal) prediction) or intra-picture (spatial) prediction mode. Therefore, **Prediction Block (PB)** was introduced. Each CB can be split into PBs differently depending on the temporal and/or spatial predictability.

2.2.5. Intra-prediction

To predict a new prediction block (PB), ***intra-picture prediction*** uses the previously decoded boundary samples from spatially neighbouring image data (in the same picture). So the first picture of a video sequence and the first picture at each clean random access point[2] into a video sequence are coded using only intra-picture prediction. This section will be explained in detail in the rest of this chapter.

An intra-predicted CU can be split into PBs only in two modes: either the PB is the same as the CB size, or the CB is split onto four smaller PBs. This latter case is only allowed for the smallest 8x8 CUs; in this case a flag specifies if the CB is split into four PBs (4x4) and each PB has their own intra prediction mode [3].

HEVC employs 35 different intra modes to predict a PB, compared to eight in H.264/MPEG-4 AVC. The modes are:

- **DC prediction:** the value of each sample of the PB is an average of the boundary samples of the neighbouring blocks.
- **Planar prediction:** the value of each sample of the PB is calculated assuming an amplitude surface with a horizontal and vertical slope derived from the boundaries samples of the neighbouring blocks.
- **Directional prediction with 33 different directional orientations:** the value of each sample of the PB is calculated extrapolating the value from the boundaries samples of the neighbouring blocks.

[2] Random access point (RAP), a point in an encoded media stream, that can be accessed directly, i.e., without the need to decode any previous portions of the bit-stream.

2.2.6. Inter-prediction

For all remaining pictures of a sequence or between random access points ***inter-picture prediction*** may be used, to make use of temporally-predictive coding modes to predict a new prediction block (PB), employing image data from one or two reference pictures before or after the current picture, and using motion compensated prediction (as has been explained in the previous section).

The encoding process for inter-picture prediction consists of choosing motion data comprising, the selected reference picture and motion vector (MV) to be applied for predicting the samples of each block. Motion vectors have up to quarter-sample resolution (luma component). The encoder and decoder generate identical inter prediction signals by applying motion compensation (MC) using the MV and mode decision data, which are transmitted as side information.

A. PB partitioning

HEVC supports more PB partition shapes for interpicture-prediction than for intrapicture-prediction. When the prediction mode is indicated as inter-prediction, the luma and chroma CBs are split into **one**, **two**, or **four** prediction blocks (PBs). When the CB is split in **one (MxM)**, the resulting PB is the same size as the corresponding CB. When a CB is split into ***two PBs***, various types of this splitting are possible (Figure 23). The cases are, $M{\times}M/2$ (CB is split into two equal-size PBs vertically), $M/2{\times}M$ (CB is split into two equal-size PBs horizontally), M/4(L)xM, M/4(R)xM, MxM/4(U), MxM/4(D). These last four modes are known as asymmetric motion partitions. The splitting into **four equally-sized PBs (M/2xM/2)** is only supported when the CB size is equal to the smallest allowed CB size (8x8 samples); in this case each PB covers a quadrant of the CB. Each inter-coded PB is assigned one or two motion vectors and reference picture indices [3]; these reference indices pointing into a reference picture list. Similar to H.264/ MPEG-4 AVC, HEVC has two reference picture, list 0 and list 1.

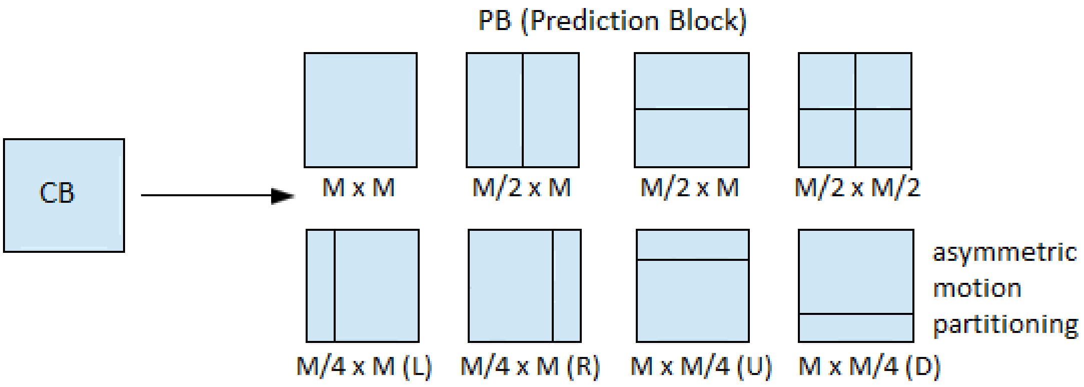


Figure 23: Prediction Block for Inter Prediction.

B. Fractional Sample Interpolation

The horizontal and vertical components of a motion vector indicate the location of the prediction in the reference picture. These components identify a block region in the reference picture, needed to obtain the prediction samples of the PB for an inter-picture predicted CB [3] [19].

In the case of **luma samples** HEVC supports motion vectors with units of one quarter of the distance between luma samples. Samples at fractional locations need to be interpolated using the content available at integer prediction locations. In order to obtain these samples, HEVC makes use of an eight-tap filter for the half-sample positions and two possible seven-tap filters for the quarter sample positions.

In Figure 24 the position labelled with capital letters, $A_{i,j}$, represent the available luma samples at integer sample locations, and the other positions labelled with lower-case letters represent samples at non-integer sample locations, which need to be generated by interpolation.

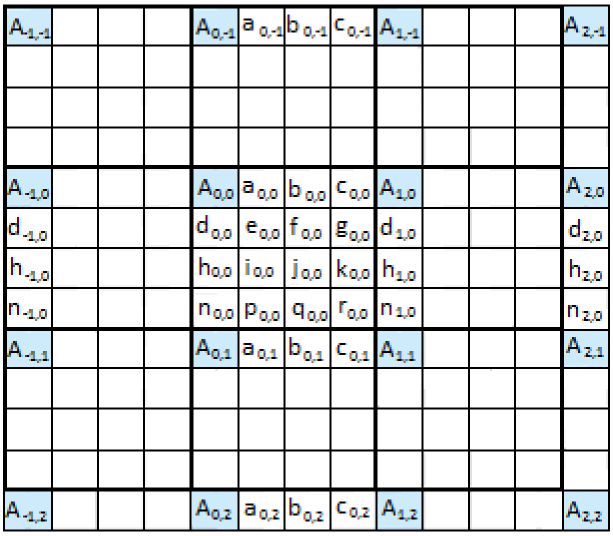


Figure 24: Integer and fractional sample positions for luma interpolation.

The next luma samples are derived from the samples Ai,j, by applying the eight-tap filter for half-sample positions and the seven-tap filter for the quarter-sample position as follows [3]:

$$\boxed{\begin{aligned}
a_{0,j} &= \left(\textstyle\sum_{i=-3..3} A_{i,j}\; qfilter[i]\right) \gg (B-8)\\
b_{0,j} &= \left(\textstyle\sum_{i=-3..4} A_{i,j}\; hfilter[i]\right) \gg (B-8)\\
c_{0,j} &= \left(\textstyle\sum_{i=-2..4} A_{i,j}\; qfilter[1-i]\right) \gg (B-8)\\
d_{0,0} &= \left(\textstyle\sum_{i=-3..3} A_{0,j}\; qfilter[j]\right) \gg (B-8)\\
h_{0,0} &= \left(\textstyle\sum_{i=-3..4} A_{0,j}\; hfilter[j]\right) \gg (B-8)\\
n_{0,0} &= \left(\textstyle\sum_{i=-2..4} A_{0,j}\; qfilter[1-j]\right) \gg (B-8)
\end{aligned}} \qquad (5)$$

B is the bit depth (number of bits used to indicate the colour of a single pixel) of the reference sample, >> denotes an arithmetic right shift operation.

The next table shows the filter coefficients for luma fractional sample interpolation:

Index i	-3	-2	-1	0	1	2	3	4
hfilter[i]	-1	4	-11	40	40	-11	4	1
qfilter[i]	-1	4	-10	58	17	-5	1	

Table 1 : Filter Coefficients for Luma Fractional Sample Interpolation.

The other samples can be derived by applying the corresponding filters to samples located at vertically adjacent $a_{0,j}$, $b_{0,j}$ and $c_{0,j}$ positions as follows [3]:

$$\boxed{\begin{aligned}
e_{0,0} &= \left(\textstyle\sum_{v=-3..3} a_{0,v} qfilter[v]\right) \gg 6\\
f_{0,0} &= \left(\textstyle\sum_{v=-3..3} b_{0,v} qfilter[v]\right) \gg 6\\
g_{0,0} &= \left(\textstyle\sum_{v=-3..3} c_{0,v} qfilter[v]\right) \gg 6\\
i_{0,0} &= \left(\textstyle\sum_{v=-3..4} a_{0,v} qfilter[v]\right) \gg 6\\
j_{0,0} &= \left(\textstyle\sum_{v=-3..4} b_{0,v} qfilter[v]\right) \gg 6\\
k_{0,0} &= \left(\textstyle\sum_{v=-3..4} c_{0,v} qfilter[v]\right) \gg 6\\
p_{0,0} &= \left(\textstyle\sum_{v=-2..4} a_{0,v} qfilter[1-v]\right) \gg 6\\
q_{0,0} &= \left(\textstyle\sum_{v=-2..4} b_{0,v} qfilter[1-v]\right) \gg 6\\
r_{0,0} &= \left(\textstyle\sum_{v=-2..4} c_{0,v}\, qfilter[1-v]\right) \gg 6
\end{aligned}} \tag{6}$$

In HEVC only explicit weighted prediction is applied, by scaling and offsetting the prediction with values explicitly transmitted in the slice header by the encoder. The bit depth of the prediction is then adjusted to the original bit depth of the reference samples.

For the **chroma samples,** the fractional sample interpolation process is similar to the one for luma component in the case of 4:2:0 sampling, except that the number of filter coefficients is 4 and the fractional accuracy is one eighth units of the distance between chroma samples. The next table shows the filter coefficients for chroma fractional sample interpolation:

Index i	-1	0	1	2
filter1[i]	-2	58	10	-2
filter2[i]	-4	54	16	-2
filter3[i]	-6	46	28	-4
filter4[i]	-4	36	36	-4

Table 2: Filter Coefficients for Chroma Fractional Sample Interpolation.

C. Merge Mode

The HEVC encoder builds a candidate list containing the motion information of spatially or temporally neighbouring PBs. The motion parameters consist of:

- The horizontal and vertical motion vector (MV) displacement values,
- One or two reference picture indices,
- In the case of prediction a region in B slices, an identification of which reference picture list is associated with each index.

Then, one of these candidates can be selected to be used in place of the motion information found following motion estimation, similar to the SKIP mode used in the H.264/AVC standard.

There are three important differences between HEVC merge mode and H.264/ MPEG-4 AVC direct and skip modes (A prediction mode in which the picture content of a block is a copy of the block in the same location in a previously decoded picture):

1. It transmits an index to select one out of several available candidates.
2. It explicitly identifies the reference picture list and reference picture index.
3. Residual samples may still be encoded and transmitted (as opposite to SKIP mode in which residuals are always skipped)

This mode builds a list of candidate MVs, and then selects one of them using an index coded in the bit-stream. The set of possible candidates list of up to 5 entries consists of spatial neighbour candidates, a temporal candidate and other artificially generated candidates. Each entry might end up being list 0, list 1 or bidirectional. The next figure shows the positions of five spatial candidates:

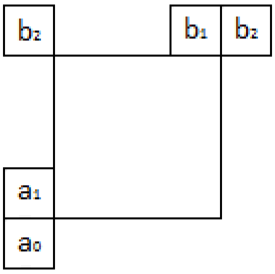


Figure 25: Positions of spatial candidates of motion parameters.

The availability is checked for each candidate, and then the candidate is added to the candidate list according to the order {a_1, b_1, b_0, a_0, b_2}. The candidate is considered as unavailable if the block located at the position is outside of the current slice or tile, or the position is intra-picture predicted.

Once the ***spatial candidates*** are validated, the position is excluded if the candidate position for the current PU would refer to the first PU within the same CU. Moreover, when the candidates have exactly the same motion parameters any redundant entries are also excluded.

If there is still space in the list, the ***temporal candidates*** can also be added. If the right bottom position (just outside of the co-located PU of the reference picture) is available, it will be added to the list. If not, the centre position will be added instead. How HEVC choose the collocated reference picture and the collocated PU allows more flexibility by transmitting an index to specify which reference picture list is used for the collocated reference picture.

For ***B slices***, if the list still has free space additional bidirectional candidates are generated formed by two existing candidates from list 0 and list 1. When the slice is a ***P slice***, zero motion vectors associated with reference indices from zero to the number of reference pictures minus one are used to fill any remaining entries in the merge candidate list. In HEVC, the skip mode, which means no transmission of residual data, is treated as a particular case of the merge mode when all coded block flags are equal to zero.

D. Advanced Motion Vector Prediction (AMVP)

This mode is used when the inter-coded CB is not coded using the merge mode or the skip mode. In order to reduce the bits needed to transmit the motion vector components, the motion vector is differentially coded, namely it is coded using a motion vector predictor. In HEVC, the encoder chooses the vector predictor among multiple candidates. The difference between the chosen predictor and the actual motion vector is transmitted along with the index of the chosen candidate.

The bit-stream specifies the reference picture to use for each MV. According to the availability of up to five candidates, HEVC only allows to choose two spatial motion candidates in Figure 25. The first spatial motion candidate is chosen from the set of left positions {a_0, a_1} and the second one from the set of above positions {b_0, b_1, b_2}.

If two spatial candidates are available, the temporal candidate is not used at all. If not the temporal candidate is included If it still having space. A zero motion vector is included until the number of motion vector prediction candidates is equal to two.

2.2.7. Transform, Scaling and Quantization

The residual signal of the intra or inter prediction, which is the difference between the original block and its prediction, is **transformed** using a block transform based on the Discrete Cosine Transform (DCT) or Discrete Sine Transform (DST). The latter is only used for intra-predicted 4x4 CUs. By means of transform, the residual signal is converted to the frequency domain in order to decorrelate and compact the information. HEVC supports four transform sizes: 4x4, 8x8, 16x16 and 32x32.

At this purpose, each CB can be differently split into **Transform Block (TBs)** using the same quad-tree method, as the CTB splitting, now called residual quadtree. The largest possible TB size is equal to the CB size. In the case of luma CB (MxM size), a flag indicate if it is split into four blocks of size $M/2{\times}M/2$, and in the case of chroma CB, size is half the luma TB size. So the smallest allowable block (TU) is 4x4 size. For example a 16x16 CU could contain three 8x8 TUs and four 4x4 TUs. For each luma TU there is a corresponding chroma TU of one quarter the size, so a 16x16 luma TU comes with two 8x8 chroma TUs [3].

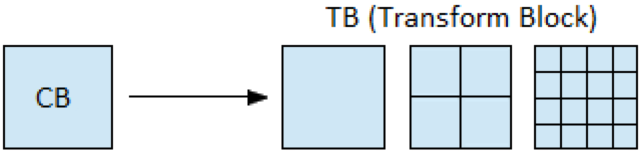


Figure 26: Transform Block

If the prediction mode was an **intra-prediction mode**, the decoded samples of the nearest neighboring TBs are used as reference data. The HEVC design allows a TB to span across multiple PBs for **inter-predicted CUs** to maximize the potential coding efficiency benefits of the quadtree structured TB partitioning.

The values of the block transform based matrix have symmetry properties to enable fast implementations with less mathematical operations than an ordinary matrix multiplication. For example, the transform matrix for the length-16 block transform is as shown:

$$H = \begin{vmatrix}
64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 \\
90 & 87 & 80 & 70 & 57 & 43 & 25 & 9 & -9 & -25 & -43 & -57 & -70 & -80 & -87 & 90 \\
89 & 75 & 50 & 18 & -18 & -50 & -75 & -89 & -89 & -75 & -50 & -18 & 18 & 50 & 75 & 89 \\
87 & 57 & -9 & -43 & -80 & -90 & -70 & -25 & 25 & 70 & 90 & 80 & 43 & -9 & -57 & -87 \\
83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 & 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 \\
80 & 9 & -70 & -87 & -25 & 57 & 90 & 43 & -43 & -90 & -57 & 25 & 87 & 70 & -9 & -80 \\
75 & -18 & -89 & -50 & 50 & 89 & 18 & -75 & -75 & 18 & 89 & 50 & -50 & -89 & -18 & 75 \\
70 & -43 & -87 & 9 & 90 & 25 & -80 & -57 & 57 & 80 & -25 & -90 & -9 & 87 & 43 & -70 \\
64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 \\
57 & -80 & -25 & 90 & -9 & -87 & 43 & 70 & -70 & -43 & 87 & 9 & -90 & 25 & 80 & -57 \\
50 & -89 & 18 & 75 & -75 & -18 & 89 & -50 & -50 & 89 & -18 & -75 & 75 & 18 & -83 & 50 \\
43 & -90 & 57 & 25 & -87 & 70 & 9 & -80 & 80 & -9 & -70 & 87 & -25 & -57 & 90 & -43 \\
36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 & 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 \\
25 & -70 & 90 & -80 & 43 & 9 & -57 & 87 & -87 & 57 & -9 & -43 & 80 & -90 & 70 & -25 \\
18 & -50 & 75 & -89 & 89 & -75 & 50 & -18 & -18 & 50 & -75 & 89 & -89 & 75 & -50 & 18 \\
9 & -25 & 43 & -57 & 70 & -80 & 87 & -90 & 90 & -87 & 80 & -70 & 57 & -43 & 25 & -9
\end{vmatrix}$$

Figure 27: Block transform based size 16x16.

After the first inverse transform stage of the transform HEVC explicitly inserts a 7-bit right shift and 16-bit clipping operation, to ensure that all intermediate values can be stored in 16-bit memory.

As already mentioned only in case a Transform Block (TB) in an intra-predicted CU has size 4x4, it is transformed by means of Discrete Sine Transform (DST). The corresponding transform matrix is:

$$H = \begin{vmatrix} 29 & 55 & 74 & 84 \\ 74 & 74 & 0 & -74 \\ 84 & -29 & -74 & 55 \\ 55 & -84 & 74 & -29 \end{vmatrix}$$

Figure 28: Block transform based size 4x4.

This transform is not more computationally challenging than the DCT and it provides 1% bit-rate reduction in intra prediction [3].

After obtaining the transform coefficients, they are then **scaled** and **quantized**. There is a pre-scaling operation in the dequantization block in H.264/MPEG-4 AVC, but in HEVC this is not needed, because the rows of the transform matrix are close approximations of values of **uniformly-scaled** basis functions of the orthonormal DCT (i.e. the scaling is incorporated in the transform operations).

For **quantization**, HEVC uses the same uniform-reconstruction quantization (URQ) scheme as in H.264/MPEG-4 AVC. URQ is controlled by a quantization parameter (QP) that is defined from 0 to 51 and an increase by 6 doubles the quantization step size [3]. This parameter regulates how much spatial detail is saved. When QP is very small, almost all the details are retained. AS QP is increased, the bit rate is lower at the price of some distortion and some loss of quality.

2.2.8. Entropy Coding

A. Transform Coefficient Coding

Once the quantized transform coefficients are obtained, they are combined with prediction information such as prediction modes, motion vectors, partitioning information and other header data, and then coded in order to obtain an HEVC bit-stream. All of these elements are coded using ***Context Adaptive Binary Arithmetic Coding (CABAC)***.

The method to encode, the quantised residual coefficients is performed in five steps: scanning, last significant coefficient coding, significance map coding, coefficient level coding, and sign data coding.

- Scanning, Last significant coefficient coding and Significance map coding:

There are three ***coefficient scanning methods***, diagonal, horizontal, and vertical scans. Those are selected for coding the transform coefficients of 4x4 and 8x8 TB sizes in intra-picture predicted regions. The selection of the scanning order depends on the directionality of the intra-picture prediction (i.e. the intra-prediction mode).

Depending on the scanning method, the transform coefficients are scanned and the position of the last coefficient different than zero is entropy coded (more details in section B). Then, starting at this last position, the coefficients are scanned backwards until coefficient in the top-left corner, known as the DC coefficient. If the size of a TB is 8x8 or larger, the TB is divided into 4x4 subblocks, called a ***coefficient group*** (GC). Each subblock is scanned depending on the scanning method, and if it contains non-zero coefficients it is entropy coded; a bit is transmitted for each of the coefficients in the group to indicate which are non-zero [10].

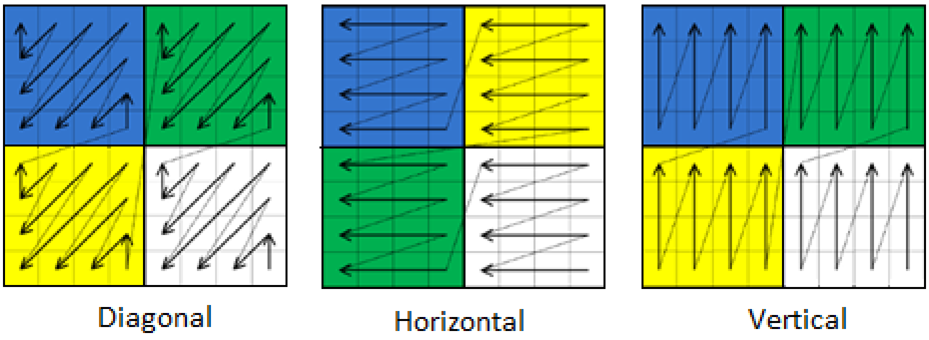


Figure 29: Three coefficient scanning methods.

In the previous image each colour represents a coefficient group. The vertical scan is used when the prediction direction is close to horizontal and the horizontal scan is used when the prediction direction is close to vertical. For other prediction directions, the diagonal scan is used. For the transform coefficients of 16×16 or 32×32 intra prediction and for the transform coefficients in inter prediction modes of all block sizes, the 4×4 diagonal scan is exclusively applied to sub-blocks of transform coefficients.

- Coefficient level coding and Sign data hiding

After the previous steps, for each of the non-zero coefficients in a group, the remaining level value (namely the absolute value of the actual coefficient value) is coded depending on two flags, whose specifies whether the level value is greater than 1 or 2.

Finally the signs of all the non-zero coefficients in the group are coded for further compression improvement. The sign bits are coded conditionally based on the number and positions of coded coefficients. HEVC has an optional tool called ***sign data hiding***. If enabled and there are at least two nonzero coefficients in a group and the difference between the scan positions of the first and the last nonzero coefficients is greater than 3, the sign bit of the first non-zero coefficient is inferred from the parity of the sum of all the coefficient's absolute values. This means that when the encoder is coding the coefficient group in question and the inferred sign is not the correct one, it has to adjust one of the coefficients up or down to compensate. The reason this tool works is that sign bits are coded in bypass mode (not compressed) and consequently are expensive to code. By not coding some of the sign bits, the savings more than compensate for any distortion caused by adjusting one of the coefficients [10].

B. Entropy coding CABAC

Entropy coding is a form of lossless compression used at the last stage of video encoding (and the first stage of video decoding), after the video has been reduced to a series of syntax elements. HEVC specifies only one entropy coding method called ***Context Adaptive Binary Arithmetic Coding (CABAC)*** [17].

CABAC involves three main functions: binarization, context modeling, and arithmetic coding. ***Binarization*** maps the syntax elements to binary symbols (bins), creating a binary string (if it is needed). Several different binarization processes are used in HEVC, including Unary (U), Truncated Unary (TU), Truncated Rice code (TR), kth-order Exp-Golomb (EGk), and Fixed-length (FL). These forms were also used in H.264/MPEG-4 AVC, so this will not be explained in detail. ***Context modelling*** estimates the probability of the bins, in order to achieve high coding efficiency. The number of context state variables used in HEVC is substantially less than in H.264/MPEG-4 AVC. Moreover, more extensive use is made in HEVC of the bypass-mode of CABAC operation (bins are coded with equi-probability, not compressed), to increase throughput by reducing the amount of data that needs to be coded using CABAC contexts. Finally, ***arithmetic coding*** compresses the bins to bits based on the estimated probability. HEVC uses the same arithmetic coding as H.264/MPEG-4 AVC [9].

2.2.9. In-Loop filters

The quantized transform coefficients are dequantised by inverse scaling and are then inverse-transformed to obtain a reconstructed approximation of the residual signal. The residual samples are then added to the prediction samples, and the result of that addition is the reconstructed samples. These samples may then be fed into two loop filters to smooth out artefacts induced by the block-wise processing and quantization. The final picture representation (which is a duplicate of the output of the decoder) is stored in a *decoded picture buffer* to be used for the prediction of subsequent pictures.

In HEVC, the two loop filters are ***deblocking filter (DBF)*** followed by a ***sample adaptive offset (SAO).*** The DBF is intended to reduce the blocking artefacts around the block boundaries that may be introduced by the lossy encoding process. The SAO operation is applied adaptively to all samples satisfying certain conditions, e.g. based on gradient. The DBF is similar to the DBF of the H.264/MPEG-4 AVC standard, while SAO is newly introduced in HEVC [3].

During the development of HEVC, it had also been considered to operate a third processing step called the adaptive loop filter (ALF) after the SAO filter, however, the ALF feature was not included in the final design [3].

A. Deblocking Filter

Deblocking in HEVC is performed to the edges that are aligned on an 8x8 sample grid only, unlike H.264/MPEG-4 AVC in which the deblocking filter is applied to every 4x4 grid. The filter is applied to the luma and chroma samples adjacent to a TU or PU boundaries. The smoothing strength depends on the QP value and on the reconstructed sample values difference at the CU boundaries. The strength of this filter is controlled by syntax elements signalled in the HEVC bit-strem.

For the deblocking filter (DBF) process, HEVC first applies horizontal filtering for vertical edges to the picture, and only after that does it apply vertical filtering for horizontal edges to the picture [3]. This process order allows for multiple parallel threads to be used for the DBF. The actual filter is very similar to H.264/MPEG-4 AVC, but only three boundary strengths 2, 1 and 0 are supported. Denote for instance as P and Q two adjacent blocks with a common 8x8 grid boundary; then a filter strength of:

- 2 means that one of the blocks is intra-picture predicted.
- 1 could mean:
  - P or Q has at least one nonzero transform coefficient.
  - The reference indices of P and Q are not equal
  - The motion vector of P and Q are not equal
  - The difference between a motion vector component of P and Q is greater than or equal to one integer sample.
- 0 means the deblocking process is not applied.

Because of the 8-pixel separation between edges, edges do not depend on each other, enabling a highly parallelized implementation. In theory the vertical edge filtering could be performed with one thread per 8-pixel column in the picture. Chroma is only deblocked when one of the PUs on either side of a particular edge is intra-coded [3].

B. SAO

After deblocking is performed, a second filter optionally processes the picture. The SAO classifies reconstructed pixels into categories and reduces the distortion, improving the appearance of smooth regions and edges of objects, by adding an offset to pixels of each category in the current region. The SAO filter is a non-linear filter that makes use of look-up tables transmitted by the encoder.

This relatively simple process is done on a per-CTB basis, and operates once on each pixel. There are two types of filters: Band and Edge.

Band Offset: In this case, SAO classifies all pixels of a region into multiple segments; each segment contains pixels in the same sample amplitude interval. The full sample amplitude range is uniformly divided into 32 intervals, called bands, from zero to the maximum sample amplitude value, and the samples values, belonging to four of these bands, are modified by adding band offsets, which can be positive or negative. This offset value directly depends on the sample amplitude. Next, the 32 bands are divided into two groups. One group consists of the 16 central bands, while the other group consists of the remaining 16 bands. Only offsets in one group are transmitted.

Edge Offset: In this case, Edge Offset uses the edge directional information (horizontal, vertical or one of two diagonal gradient directions) for the edge offset classification in the CTB. There are four gradient patterns used in SAO, as shown in Figure 30; "n_0" and "n_1" indicate two neighbouring samples along the gradient pattern and "p" specifies a centre sample to be considered, so the directionalities are (a) horizontal (0-degrees), (b) vertical (90-degree), (c) diagonal (135-degrees) and (d) 45-degree.

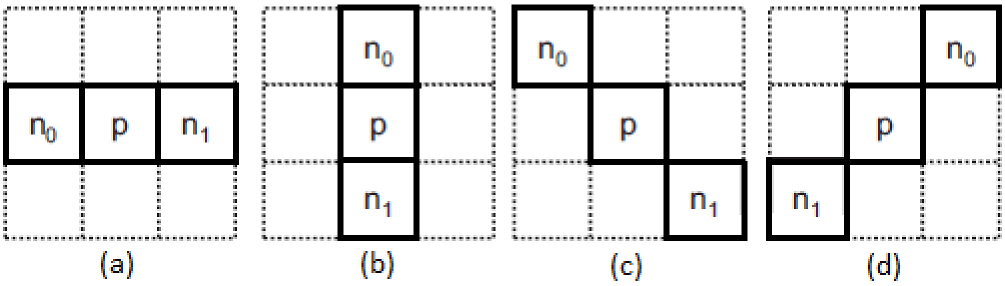


Figure 30: Four gradient patterns.

Each region of a picture can select one pattern to classify sample into five EdgeIdx categories by comparing each sample ("p") with its two neighbouring samples ("n_0" and "n_1"). Each of these two neighbours can be less than, greater than or equal to the current sample, as shown in table 3. Depending on the outcome of these two comparisons, the sample is either unchanged or one of the four offsets is added to it. The offsets and filter modes are picked by the encoder in an attempt to make the CTB more closely match the source image.

EdgeIdx	Condition	Meaning
0	$p=n_0$ and $p=n_1$	Flat area
1	$p < n_0$ and $p < n_1$	Local area
2	$p < n_0$ and $p=n_1$ or $p < n_1$ and $p=n_0$	Edge
3	$p > n_0$ and $p=n_1$ or $p > n_1$ and $p=n_0$	Edge
4	$p > n_0$ and $p > n_1$	Local mas

Table 3 : Sample EdgeIdx Categories in SAO Edge Classes.

2.3. Intra Prediction

The Intra-picture prediction uses the previously decoded boundary samples from spatially neighbouring TBs in order to predict a new prediction block PB. So the first picture of a video sequence and the first picture at each clean random access point into a video sequence are coded using only intra-picture prediction [3].

Several improvements have been introduced in HEVC in the intra prediction module:

- Due to the larger size of the pictures, the range of supported coding block sizes has been increased.
- A plane mode that guarantees continuity at block boundaries is desired.
- The number of directional orientations has been increased.
- For intra mode coding, efficient coding techniques to transmit the mode for each block are needed due to the increased number of intra modes.
- HEVC supports a large variety of block size, so it needs consistency across all block size.

HEVC employs 35 different intra modes to predict a PB: 33 Angular modes, Planar mode, and DC mode. The next table shows the mode name with their corresponding intra prediction mode index as by the convention used throughout the standard [12].

Mode Name	Intra Prediction Mode
Planar	0
DC	1
Angular	2…34

Table 4: Specification of Intra Prediction modes and associated index.

At the end the video encoder will choose the intra prediction mode that provides de best Rate-Distortion performance.

2.3.1. PB partitioning

The CB can be split into size of MxM or M/2xM/2. The first one means that the CB is not split, so the PB has the same size as the CB. It is possible to use it in all CUs. The second partitioning means that the CB is split into four equally-sized PBs. This can only be used in the smallest 8x8 CUs. In this case, a flag is used to select which partitioning is used in the CU. Each resulting PB has its own intra prediction mode [3]. The prediction blocks size range from 4x4 to 64x64.

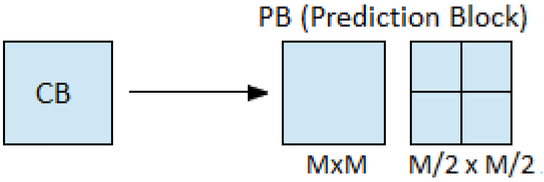


Figure 31: Prediction Block for Intra Prediction.

2.3.2. Reference Samples

All the intra prediction modes use the same set of **reference samples,** which are extracted at the boundary from the upper (red samples) and left (blue samples) blocks adjacent to the current PU. For the diagonal directions the top-left corner (pink) sample may also be used. Also, it is possible to use the lower left (orange samples) and above right (green samples), if they are available from preceding decoding operations, as shown in Figure 32. Therefore, to predict the current PU of size MxM, a total of 4M+1 spatially neighbouring reference samples may be used [3].

In the next sections, the reference samples are named as $R_{x,y}$ with the origins in the top-left corner. Also, the predicted sample value in the current PU in the position (x,y) is denoted as $P_{x,y}$. This notation is shown as follows in a block of size MxM samples [12].

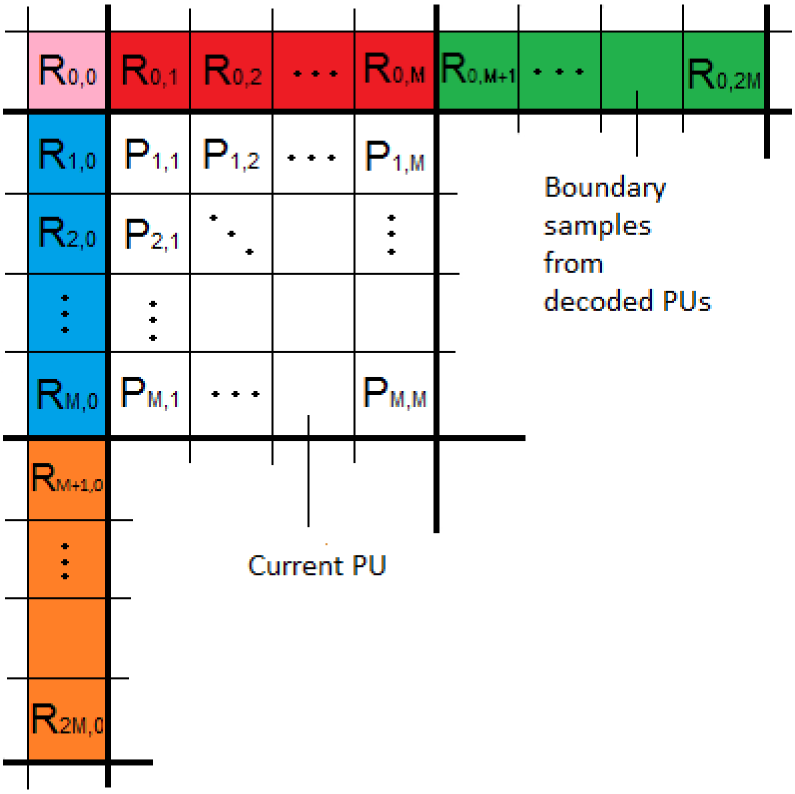


Figure 32: Reference samples $R_{x,y}$.

There are three cases in which reference samples may not be available. The **first case** is when the reference samples at picture or slice boundaries are unavailable; in this case a nominal average sample value is used as a replacement for the reference samples depending on the sample bit-depth; for an 8-bit data, this is 128. The **second case** is when just the left boundary references are missing; these reference samples are generated copying the samples from the closest available reference samples above. And the **third case** is when just the top boundary references are missing; these reference samples are generated copying the samples from the closest available reference samples on the left [12].

2.3.3. Angular Modes

The angular prediction process is similar to H.264/MPEG-4 AVC, operating in the spatial domain. This method has though improved significantly, thanks to the increased number of directional orientations and the bigger block sizes, providing a good compromise between encoding complexity and coding efficiency for typical video material [3].

The angular mode in HEVC supports 33 different directional orientations, compared to eight in H.264/MPEG-4 AVC. This number of directional orientations has been selected to provide a good compromise between coding efficiency and encoding complexity for typical video [12].

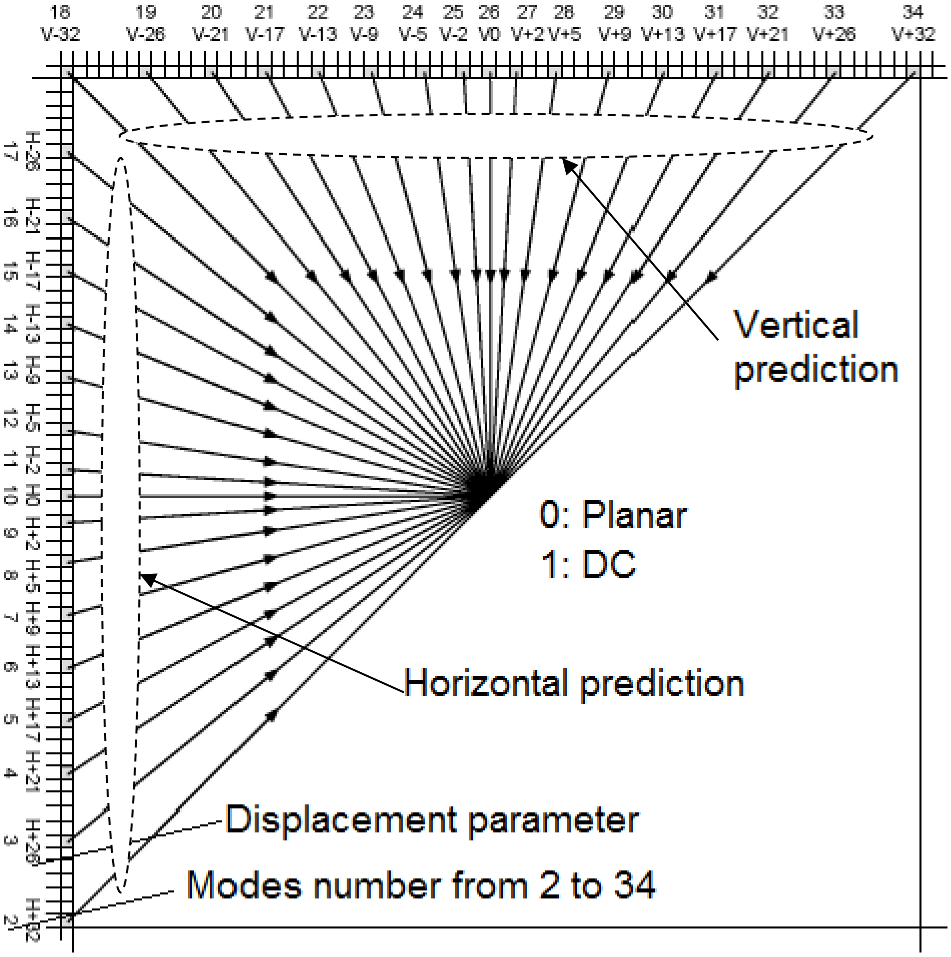


Figure 33: Modes and directional orientations for intra-picture prediction.

Angular modes are indexed from 2 to 34, as shown in Figure 33. Directions may cover angles from near-horizontal through near-diagonal to near-vertical. Each mode has associated a displacement parameter *d*, where the value of *d* indicates the numeric part which is the pixel's displacement expressed in 1/32 pixel fractions of accuracy, and *H* and *V* indicate the horizontal and vertical directionalities, respectively [12]. The modes 10 are 26 are known as pure horizontal prediction and pure vertical prediction, respectively.

It must be taken into account that the horizontal and vertical patterns normally occur more frequently in the natural imagery than patterns with other directionalities. The number of directional orientations is chosen in HEVC because it optimizes the prediction accuracy based on these patterns [12]. So for example, in an image with vertical stripes, the prediction mode 26 would give better compression capability than the others.

For each octant eight angles are defined with associated displacement parameters, as shown in Figure 33. When getting closer to diagonal directions, the displacement parameter become larger in order to reduce the density of prediction modes for less occurring directions. For modes close to horizontal and vertical directions, the displacement becomes smaller in order to provide more accurate prediction for nearly horizontal and vertical patterns.

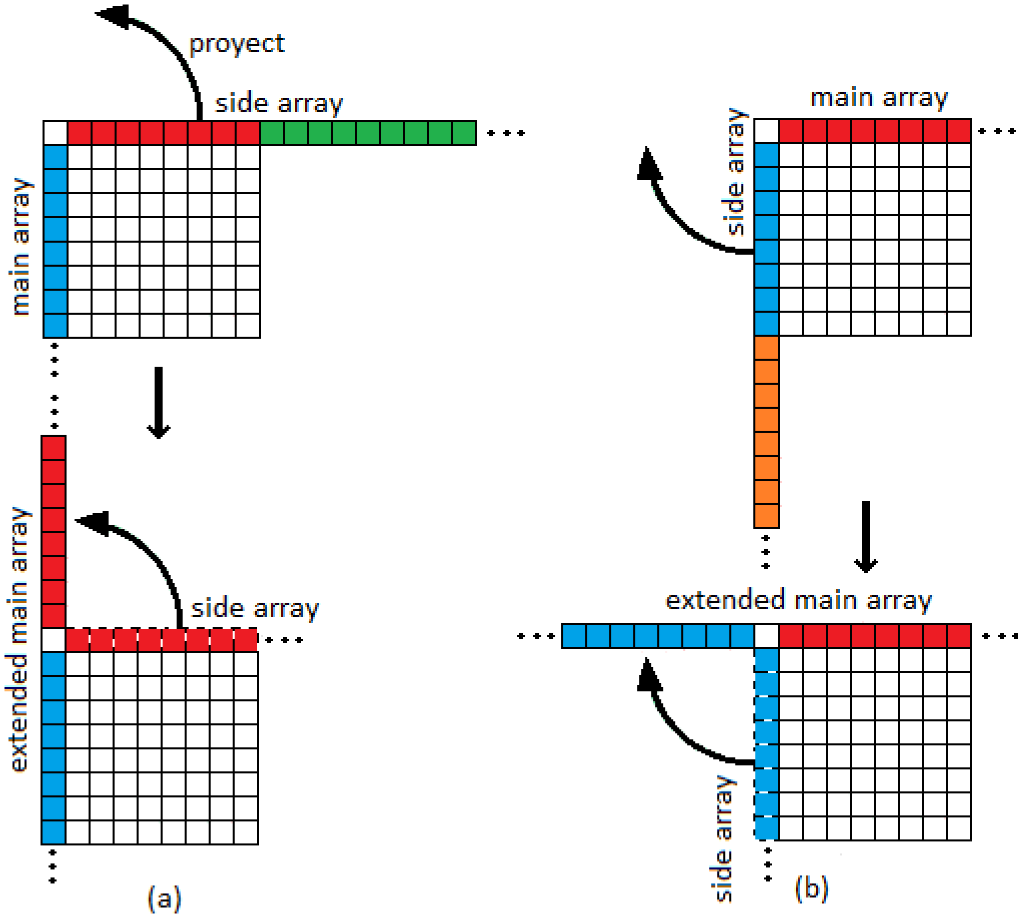


Figure 34: Example diagonal orientation: mode 18.

In order to calculate the value of each sample of the PB, the angular mode extrapolates the samples from the reference samples, depending on the directional orientation in order to achieve lower complexity. When the direction selected is between 2 and 17, the samples located in the above row (red samples and may be green samples, as shown in Figure 34 (a)) are projected as additional samples located in the left column, extending the left reference column. When the direction selected is between 18 and 34, the samples located at the left column (blue samples and may be orange samples, as shown in figure 34 (b)) are projected as samples located in the above row, extending the top reference row. In both cases the samples projected would have negative indexes [3].

Once the reference samples are known, a bi-linear interpolation is used to obtain the value of each predicted sample $P_{x,y}$ by projecting its location into the reference samples and applying the selected prediction direction, utilizing the two closest reference samples [13].

For the vertical prediction modes 18 to 34, the reference row above the block is used as follows:

$$\boxed{P_{x,y} = \left((32 - \omega_y) \cdot R_{0,i} + \omega_y \cdot R_{0,i+1} + 16\right) \gg 5} \tag{7}$$

$$\boxed{\begin{gathered}\omega_y = (y \cdot d) \circ 31 \\ i = x + c_y \\ c_y = (y \cdot d) \gg 5\end{gathered}} \tag{8}$$

For the horizontal prediction modes 2 to 17, the left reference column from the current block is used as in:

$$\boxed{P_{x,y} = \left((32 - \omega_x) \cdot R_{i,0} + \omega_x \cdot R_{i+1,0} + 16\right) \gg 5} \tag{9}$$

$$\boxed{\begin{gathered}w_y = (y \cdot d) \circ 31 \\ i = y + c_x \\ c_x = (x \cdot d) \gg 5\end{gathered}} \tag{10}$$

Where,
ω_y or $\omega_x$is the weighting parameter, namely the weighting factor between the two reference samples $R_{0,i}$ and $R_{0,i+1}$ or $R_{i,0}$ and $R_{i+1,0}$, respectively.

- $\gg$ is the bit shift operation to the right.
- $\circ$ is the logical operation AND.
- i is the reference sample index.
- d is the projection displacement associated with the selected prediction direction, having a value from -32 to 32.

The parameter c_y and ω_y or c_x and ω_x remain constant when the prediction samples are being calculated for one line of samples, although (1) or (3) change at each sample, respectively.

2.3.4. Planar Mode

This mode in HEVC is similar to the planar mode in H.264/MPEG-4 AVC, and is known as mode 0. In H.264/MPEG-4 AVC this method is a plane prediction mode for textured images, and may introduce discontinuities along the block boundaries. Conversely, in HEVC this mode was improved in order to preserve continuities along the block edges.

The value of each sample of the PB is calculated assuming an amplitude surface with a horizontal and vertical smooth gradient derived from the boundaries samples of the neighbouring blocks [3]. HEVC supports all block sizes for this mode.

$$\boxed{\begin{aligned} P^V_{x,y} &= (N-y)\cdot R_{0,x} + i\cdot R_{N,0} \\ P^H_{x,y} &= (N-x)\cdot R_{y,0} + j\cdot R_{0,N} \end{aligned}} \tag{11}$$

$$\boxed{P_{x,y} = \left(P^V_{x,y} + P^H_{x,y} + N\right) \gg (\log_2(N)+1)} \tag{12}$$

Planar mode is essentially defined as an average value of two linear predictions using four corner reference samples. This mode is implemented as follows, with reference to Figure 35; the sample X is the first sample predicted as an average of the samples D and E, then the right column samples (blue samples) are predicted using bilinear interpolation between samples in D and X, and the bottom row samples (orange samples) are predicted using bilinear interpolation between samples in E and X. The remaining samples are predicted as the averages of bilinear interpolations between boundaries samples and previously coded samples [25].

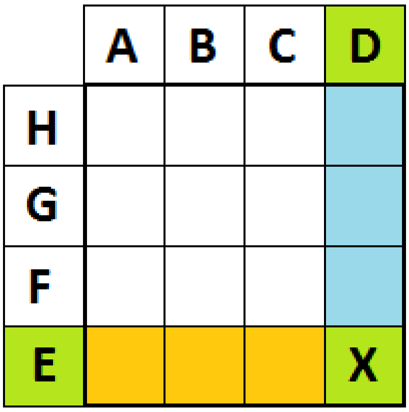


Figure 35: Planar intra prediction mode.

2.3.5. DC Mode

This mode is also similar to the DC mode in H.264/MPEG-4 AVC and is known as mode 1. It is efficient to predict plane areas of smoothly-varying content in the image, but gives a coarse prediction on the content of higher frequency components and as such it is not efficient for finely textured areas.

The value of each sample of the PB is an average of the reference samples. As explained before, for this case the reference samples will be the boundary samples of the top and left neighbouring TBs. So, all the samples of the PB are predicted with the same value [26] as in:

$$\boxed{P_{x,y} = \frac{1}{2M}\sum_{M}\left(R_{0,\mathrm{x}} + R_{\mathrm{y},0}\right)} \tag{13}$$

The previous equation is shown in the next picture in a block of MxM samples:

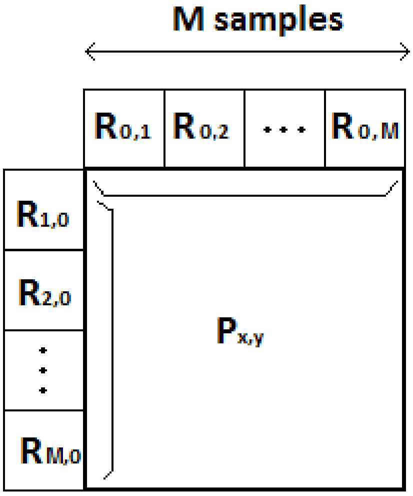


Figure 36: DC intra prediction mode.

2.3.6. Smoothing filter

HEVC uses a smoothing filter in order to reduce the discontinuities introduced by the intra-prediction modes. This is applied to the boundary samples, namely the first prediction row and column for DC mode, or the first prediction row for pure horizontal prediction, or the first prediction column for pure vertical prediction. The smoothing filter consists of a two-tap finite impulse response filter for DC prediction or a gradient-based smoothing filter for horizontal (mode 10) and vertical (mode 26) prediction. Due to the fact that chroma components tend to be already smooth, this filter is not used in this case. Prediction boundary smoothing is only applied to luma component.

The smoothing filter is applied to the reference samples depending on the size of the blocks and the directionalities of the prediction. Thanks to using this filter, contouring artefacts caused by boundaries in the reference samples may be drastically reduced.

3. DESIGN AND DEVELOPMENT: IMPROVED INTRA-PREDICTION FOR FAST HEVC

3.1. Problem Formulation

The HEVC standard is significantly more computationally complex than its predecessor H.264/AVC. In the case of HEVC intra-prediction, the first reason why it is more complex is because larger block sizes have been introduced (4x4, 8x8, 16x16, 32x32 and 64x64) compared to H.264/AVC. And the second reason is because there are a larger number of possible prediction modes (35 in HEVC specification versus 9 in H.264/AVC). This means the process of deciding the optimal intra prediction mode in HEVC consists in testing all the possible intra prediction modes and choosing the best one in terms of a cost metric (for instance the SAD, Sum of Absolute Differences) optimization criterion. As a result, HEVC intra-prediction is extremely complex.

This thesis presents a fast intra coding mode decision algorithm whose objective is that of reducing the computational complexity of HEVC while at the same time trying to maintain its coding efficiency. To avoid an exhaustive search among all possible intra modes, an optimal intra prediction selection procedure is proposed which consists of extracting some features from the currently coded blocks and then obtaining a classification of the current block to determine which modes need to be used or tested.

As it will be shown the proposed method significantly reduces the computational complexity of the intra prediction block, causing very small losses in terms of coding efficiency.

In the following sections, the HEVC intra prediction framework developed in MATLAB for implementing and testing the approach will be presented. Particular emphasis will be given to the differences between the implemented framework and conventional HEVC specifications. Finally, the proposed approach for a fast intra coding mode decision algorithm will be detailed in the second part of this chapter.

3.2. HEVC framework

In this section the HEVC intra prediction framework developed in MATLAB is described. The intra prediction algorithm designed is similar to the HEVC specification but with a few differences, as described below. The first difference is that the **maximum block size is set to 32x32,** instead of 64x64 as in conventional HEVC. The **bit depth used is 8 bits**, as opposite to conventional HEVC in which this can be set to as high as 16 bits. Both these limitations have been considered to limit the complexity of the implementation: in particular due to the way that transform is computed in HEVC, limiting the blocks to 32x32 makes it much easier to obtain the transformed and quantised coefficients.

To test the algorithm 24 images in yuv format [29] were imported with a dimension of 768x512 (width and height respectively) sampled in 4:2:0 format. Four different quantization parameters (QP) are tested (22, 27, 32, and 37). For each QP, each image is partitioned into blocks of a specified fixed size (4x4, 8x8, 16x16 or 32x32). This size is controlled by the parameter SizeBlocks, named in this thesis as N. This step consists in taking the samples of a frame in sizes of NxN, as shown in Figure 37. Then for each block the following is calculated:

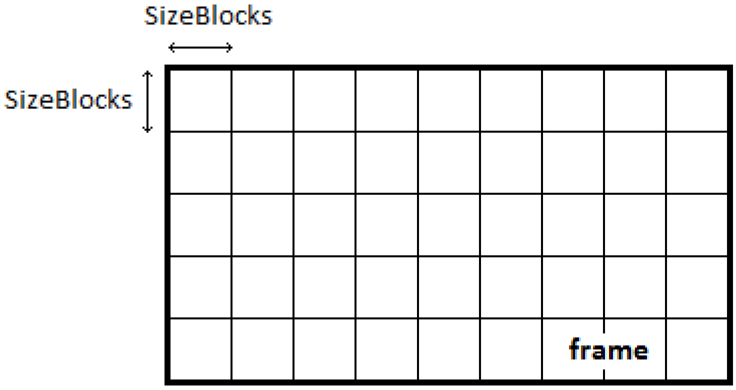


Figure 37 : Representation of the partition blocks.

A. Reference samples

The samples needed as reference are extracted from the top row and left column adjacent to the current block. The **lower left and above right reference samples are not used** because they are not encoded. For this reason some of the angular modes available in HEVC are not defined in the proposed framework: in particular modes 2 to 9 and 27 to 34. This means that in the proposed framework only **angular modes from 10 to 26 are used.**

When there are some reference samples unavailable (for instance in blocks extracted at the boundaries) instead of copying the samples from the closest available reference samples (as in conventional HEVC), **the value 128** (2^{8-1}, because the bit depth is fixed to 8) is used as reference sample for all missing samples.

The samples in the reference are written first in the top row and then in the left column as shown in Table 5.

Top row (1)	Left column (2)

Table 5: Representation of the reference samples.

There are four different cases to extract the reference samples depending on the current block location:

- **First case:** when the current block is in the top-left corner (first block), all the reference samples are filled with the value 128. For instance in case of a frame partitioned in 4x4 blocks, the reference samples in this case will be:

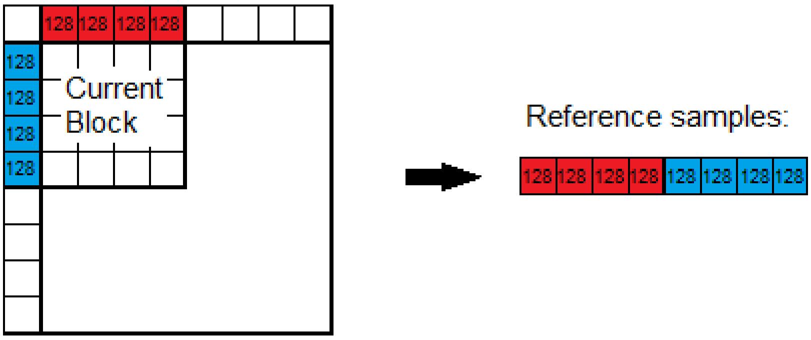


Figure 38: Representation first case.

- **Second case:** when the current block is in the first row, the top row reference samples have the value 128, and the left column reference samples are the samples in the boundary from the block on the left of the current block. For instance in case of a frame partitioned in 4x4 blocks, the reference samples in this case will be:

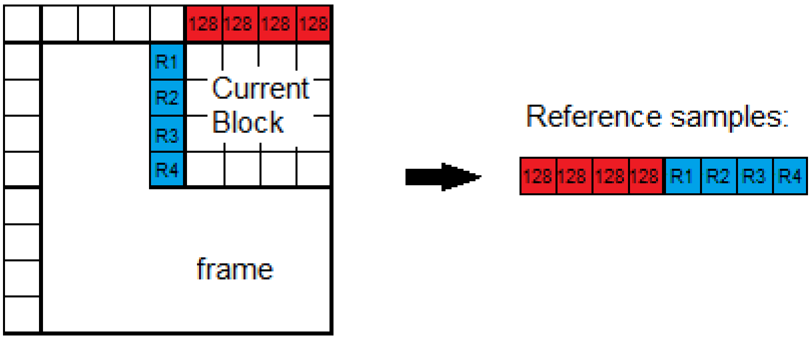


Figure 39: Representation second case.

- **Third case:** when the current block is in the first column, the top row reference samples are the samples in the boundary from the block on top of the current block, and the left column reference samples have the value 128. For instance in case of a frame partitioned in 4x4 blocks, the reference samples in this case will be:

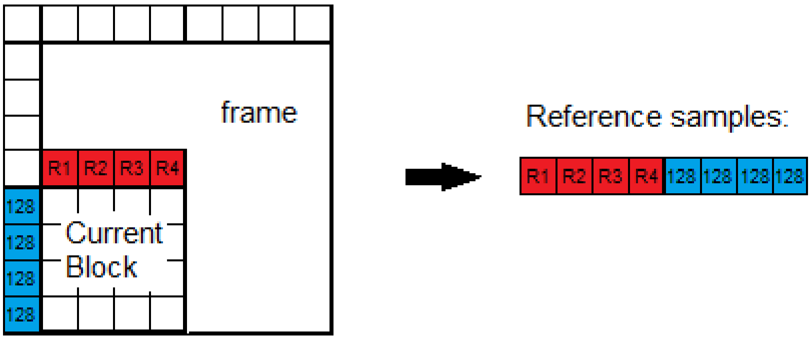


Figure 40: Representation third case.

- **Fourth case:** when the current block is located at one of the remaining positions, the reference samples will be the samples in the boundary from the upper and left blocks adjacent to the current PU. For instance in case of a frame partitioned in 4x4 blocks, the reference samples in this case will be:

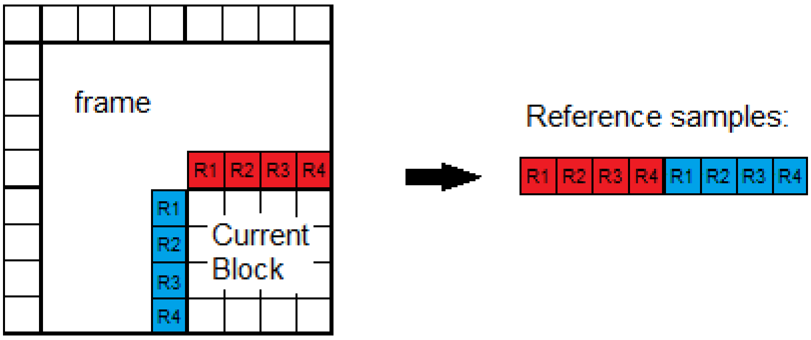


Figure 41: Representation fourth case.

B. Angular Reference Samples

Only when the angular mode is used or tested to predict the block, the reference samples are extracted depending on the directionality of the mode being tested as illustrated here:

- **If the mode is a horizontal direction:** The top row is first saved and then the left column. There are also four cases:
  - **First case:** when the block is in the top left corner of the frame, the reference samples are filled with the number 128.

1	2… …N+1	N+2	N+3… …(2*N)+2	(2*N)+3
128	128… …128	128	128… …128	128

  - **Second case:** when the current block is in the first row of the frame, the top row reference samples (1) are filled in with the number 128 and the left column reference samples (2) are filled in with the samples in the boundary from the left TB.

1	2… … N+1	N+2	N+3… …(2*N)+2	(2*N)+3
128	128… …128	128	Left column (2)	128

- **Third case:** when the current block is in the first column of the frame, the top row reference samples (1) are filled in with the samples in the boundary from the top TB and the left column reference samples (2) are filled in with the number 128.

1	2… … N+1	N+2	N+3……(2*N)+2	(2*N)+3
128	Top row (1)	128	128… …128	128

- **Fourth case:** for the remaining cases, the corner is filled in first to perform the diagonal directionality with the down-right corner sample of the up and left TB, then the top row is first filled in and then the left column.

1	2… … N+1	N+2	N+3……(2*N)+2	(2*N)+3
corner	Top row (1)	128	Left column (2)	128

Table 6: Angular reference samples (horizontal directionality).

- **If the mode is a vertical direction:** The left column is first saved and then the top row. There are also four cases:

  - **First case:** when the block is in the top left corner of the frame, the reference samples are filled in with the number 128.

1	2… … N+1	N+2	N+3… …(2*N)+2	(2*N)+3
128	128… …128	128	128… …128	128

  - **Second case:** when the current block is in the first row, the left column reference samples (2) are filled in with the samples in the boundary from the left TB and the top row reference samples (1) are filled in with the number 128.

1	2… … N+1	N+2	N+3… …(2*N)+2	(2*N)+3
128	Left column (2)	128	128… …128	128

  - **Third case:** when the current block is in the first column, the left column reference samples (2) are filled in with the number 128 and the top row reference samples (1) are filled in with the samples in the boundary from the top TB.

1	2… … N+1	N+2	N+3… …(2*N)+2	(2*N)+3
128	128… …128	128	Top row (1)	128

  - **Fourth case:** for the remaining cases, the corner is filled in first to perform the diagonal directionality with the down-right corner sample of the up and left TB, then the left column is first filled in and then the top row.

1	2… … N+1	N+2	N+3… …(2*N)+2	(2*N)+3
corner	Left column (2)	128	Top row (1)	128

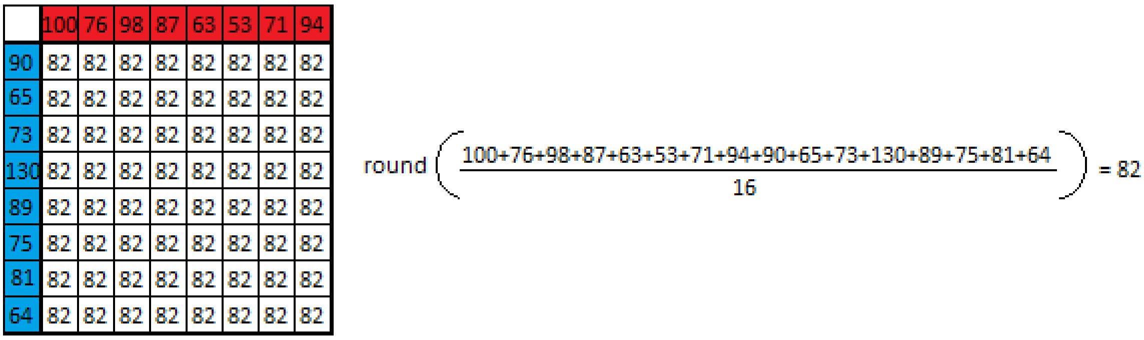
Table 7: Angular reference samples (vertical directionality).

C. Mode DC prediction

The mode DC uses all the reference samples to calculate each sample in the prediction block. In this case an average is performed on this reference samples obtaining the prediction block. In particular:

```
PredictionDCBlock = round(double(sum(References)) / (2*N));
```

For example in an 8x8 block, the prediction block will be:



	100	76	98	87	63	53	71	94
90	82	82	82	82	82	82	82	82
65	82	82	82	82	82	82	82	82
73	82	82	82	82	82	82	82	82
130	82	82	82	82	82	82	82	82
89	82	82	82	82	82	82	82	82
75	82	82	82	82	82	82	82	82
81	82	82	82	82	82	82	82	82
64	82	82	82	82	82	82	82	82

$$\text{round}\left(\frac{100+76+98+87+63+53+71+94+90+65+73+130+89+75+81+64}{16}\right) = 82$$

Figure 42: An example of mode DC prediction.

The residual block is then obtained as the difference between the original block and the prediction block, or:

```
residualDCBlock = double(OriginalBlock) - double(PredictionDCBlock);
```

Finally the SAD (Sum of Absolute Difference) of the residual is calculated as:

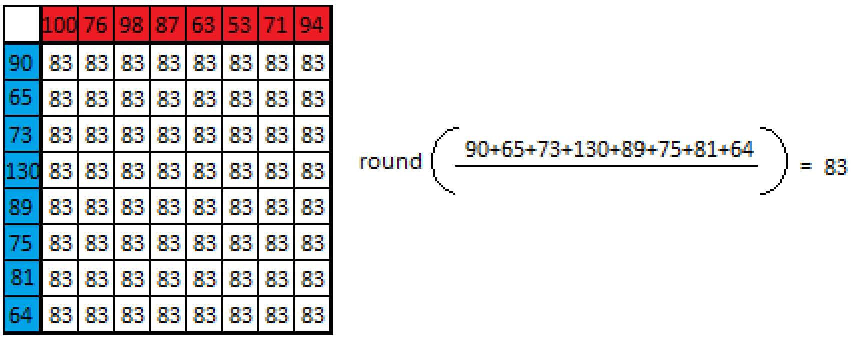
```
SADmodeDC = sum(sum(abs(residualDCBlock)));
```

D. Mode LeftDC prediction

The mode Left DC function uses just the left column samples of the reference (2) in order to calculate each sample for each prediction block. This mode does not exist in HEVC specifications. In this case an average is performed on this specified reference samples obtaining the prediction block, in general:

```
PredictionLeftDCBlock = round(double(sum(References))  / N);
```

For example in an 8x8 block, the prediction block will be:



	100	76	98	87	63	53	71	94
90	83	83	83	83	83	83	83	83
65	83	83	83	83	83	83	83	83
73	83	83	83	83	83	83	83	83
130	83	83	83	83	83	83	83	83
89	83	83	83	83	83	83	83	83
75	83	83	83	83	83	83	83	83
81	83	83	83	83	83	83	83	83
64	83	83	83	83	83	83	83	83

$$\text{round}\left(\frac{90+65+73+130+89+75+81+64}{8}\right) = 83$$

Figure 43: An example of mode Left DC prediction.

The residual block is then obtained as the difference between the original block and the prediction block, or:

residualLeftDCBlock = double(OriginalBlock) - double(PredictionLeftDCBlock);

Finally the SAD (Sum of Absolute Difference) of the residual is calculated as:

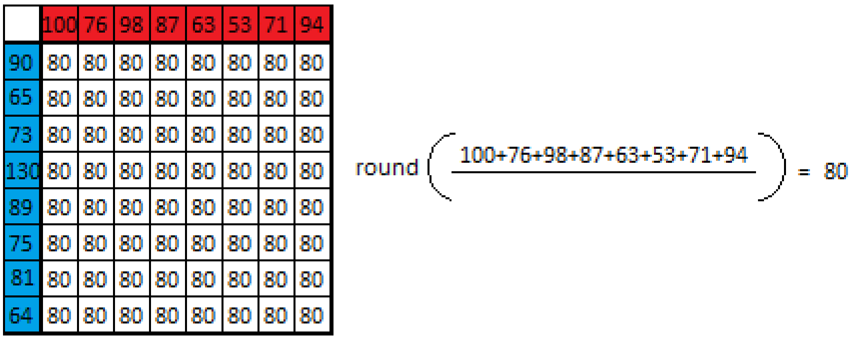
SADmodeLeftDC = sum(sum(abs(residualLeftDCBlock)));

E. Mode TopDC prediction

The mode Top DC function uses just the top row samples of the reference array (1) in order to calculate each sample for each prediction block. This mode does not exist in the HEVC specifications. In this case an average is performed on these reference samples to obtain the prediction block, or:

PredictionTopDCBlock = round(double(sum(References)) / N);

For example in an 8x8 block, the prediction block will be:



	100	76	98	87	63	53	71	94
90	80	80	80	80	80	80	80	80
65	80	80	80	80	80	80	80	80
73	80	80	80	80	80	80	80	80
130	80	80	80	80	80	80	80	80
89	80	80	80	80	80	80	80	80
75	80	80	80	80	80	80	80	80
81	80	80	80	80	80	80	80	80
64	80	80	80	80	80	80	80	80

$$\text{round}\left(\frac{100+76+98+87+63+53+71+94}{}\right) = 80$$

Figure 44: An example of mode Top DC prediction.

The residual block is then obtained as the difference between the original block and the prediction block, or:

residualTopDCBlock = double(OriginalBlock) - double(PredictionTopDCBlock);

Finally the SAD (Sum of Absolute Difference) of the residual is calculated as:

SADmodeTopDC = sum(sum(abs(residualTopDCBlock)));

F. Mode Planar prediction

The Planar mode uses the extracted reference samples to calculate each sample for each prediction block. In this case as mentioned in the state of the art the planar mode is performed using the following formulas to obtain the prediction block ($P_{i,j}$) for each i,j sample (i indicates the row and j indicates the column). Assuming a block of size NxN:

$$\boxed{\begin{aligned} P^{V}_{i,j} &= (N-i)\cdot R_{0,j} + i\cdot R_{N,0} \\ P^{H}_{i,j} &= (N-j)\cdot R_{i,0} + j\cdot R_{0,N} \end{aligned}} \qquad (14)$$

$$\boxed{P_{i,j} = floor\left(\left(P^{V}_{i,j} + P^{H}_{i,j} + N\right)/2^{(\log_2(N)+1)}\right)} \qquad (15)$$

Where $R_{0,j}$ indicates the top row reference samples (1) and $R_{i,0}$ indicates the left column reference samples (2), as shown in table 8:

Top row (1)					**Left column (2)**				
$\boldsymbol{R_{0,1}}$	**…**	$\boldsymbol{R_{0,j}}$	**…**	$\boldsymbol{R_{0,N}}$	$\boldsymbol{R_{1,0}}$	**…**	$\boldsymbol{R_{i,0}}$	**…**	$\boldsymbol{R_{N,0}}$

Table 8: Reference samples mode planar.

For example in an 8x8 block (N = 8), with the reference samples as shown in table 9:

$R_{0,1}$	$R_{0,2}$	$R_{0,3}$	$R_{0,4}$	$R_{0,5}$	$R_{0,6}$	$R_{0,7}$	$R_{0,8}$	$R_{1,0}$	$R_{2,0}$	$R_{3,0}$	$R_{4,0}$	$R_{5,0}$	$R_{6,0}$	$R_{7,0}$	$R_{8,0}$
100	76	98	87	63	53	71	98	90	65	73	130	89	75	81	64

Table 9: An example of reference samples in mode planar.

The process of calculating some prediction samples is shown in table 10:

Sample position	$P^{V}_{i,j}$	$P^{H}_{i,j}$	$floor(P_{i,j})$
i = 1 and j = 1	(N-i)* $R_{0,1}$ + i* $R_{8,0}$ = (8-1)*100 + 1*64 = 764	(N-j)* $R_{1,0}$ + j* $R_{0,8}$ = (8-1)*90 + 1*98 = 728	($P_{1,1}^{V}$ + $P_{1,1}^{H}$ +N) /16 = 764+728+8/16 = **93**
i = 1 and j = 5	(N-i)* $R_{0,5}$ + i* $R_{8,0}$ = (8-1)*63 + 1*64 = 505	(N-j)* $R_{1,0}$ + j* $R_{0,8}$ = (8-5)*90+5*98 = 760	($P_{1,5}^{V}$ + $P_{1,5}^{H}$ +N) /16 = 505+760+8 /16 = 79
i = 3 and j = 6	(N-i)* $R_{0,6}$ + i* $R_{8,0}$ = (8-3)*53+3*64 = 457	(N-j)* $R_{3,0}$ + j* $R_{0,8}$ = (8-6)*73 + 6*98 = 734	($P_{3,6}^{V}$ + $P_{3,6}^{H}$ +N) /16 = 457+734+8 /16 = **74**
i = 8 and j= 8	(N-i)* $R_{0,8}$ + i* $R_{8,0}$ = (8-8)*98 + 8*64 = 512	(N-j)* $R_{8,0}$ + j* $R_{0,8}$ = (8-8)*64 + 8*98 = 784	($P_{8,8}^{V}$ + $P_{8,8}^{H}$ +N) /16= 512+784+8 /16 = **81**

Table 10: An example of prediction samples in planar mode.

The residual block is then obtained as the difference between the original block and the prediction block, or:

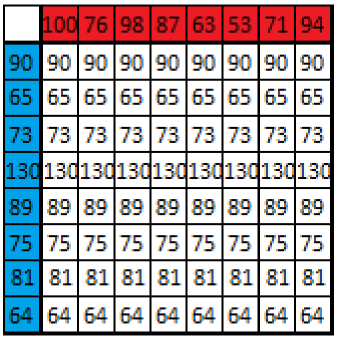
```
residualPlanarBlock = double(OriginalBlocks) - double(PredictionPlanarBlocks);
```

Finally the SAD (Sum of Absolute Difference) of the residual is computed as:

```
SADplanarMode = sum(sum(abs(residualPlanarBlock)));
```

G. Mode Pure Horizontal prediction

The Pure Horizontal mode uses the left column reference samples (2) in order to calculate each sample for each prediction block. This mode produces exactly the same output as the angular mode number 10 in the HEVC specifications. In this case each sample of the left reference column is replicated horizontally in the prediction block, for example an 8x8 prediction block will be:



	100	76	98	87	63	53	71	94
90	90	90	90	90	90	90	90	90
65	65	65	65	65	65	65	65	65
73	73	73	73	73	73	73	73	73
130	130	130	130	130	130	130	130	130
89	89	89	89	89	89	89	89	89
75	75	75	75	75	75	75	75	75
81	81	81	81	81	81	81	81	81
64	64	64	64	64	64	64	64	64

Figure 45: An example of mode pure horizontal prediction.

The residual block is then obtained as the difference between the original block and the prediction block, or:

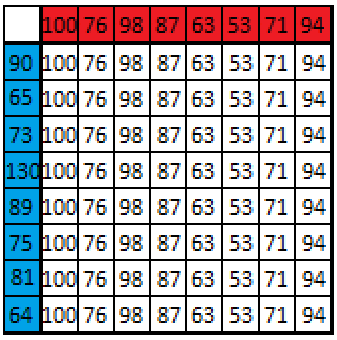
residualPHBlock = double(OriginalBlocks) - double(PredictionPHBlocks);

Finally the SAD (Sum of Absolute Difference) of the residual is calculated as:

SADPHorizontal = sum(sum(abs(residualPHBlock)));

H. Mode Pure Vertical prediction

The Pure Vertical mode function uses the top row reference samples (1) in order to calculate each sample for each prediction block. This mode produces exactly the same output as the angular mode number 26 in the HEVC specifications. In this case each sample of the top reference row is replicated vertically in the prediction block. For instance an 8x8 prediction block will be:



	100	76	98	87	63	53	71	94
90	100	76	98	87	63	53	71	94
65	100	76	98	87	63	53	71	94
73	100	76	98	87	63	53	71	94
130	100	76	98	87	63	53	71	94
89	100	76	98	87	63	53	71	94
75	100	76	98	87	63	53	71	94
81	100	76	98	87	63	53	71	94
64	100	76	98	87	63	53	71	94

Figure 46: An example of mode pure vertical prediction.

The residual block is then obtained as the difference between the original block and the prediction block, or:

residualPVBlock = double(OriginalBlocks) - double(PredictionPVBlocks);

Finally the SAD (Sum of Absolute Difference) of the residual is computed as:

SADPVertical = sum(sum(abs(residualPVBlock)));

I. Mode Angular prediction

The Angular modes use the reference samples generated for this particular case by the function Angular Reference Samples (as previously illustrated) in order to calculate each sample for each prediction block. In this case each sample is predicted according to a certain directional orientation. The angular modes used in this framework are **17 modes** instead of the 33 modes in the HEVC specification, from mode 10 to 26. The others modes are not used here because in the proposed framework, the reference samples necessary to produce a prediction in such modes are not encoded yet. A table with the index of the angular mode and its associated displacement parameter *d* (describing the tangent of the prediction direction in units of 1/32 sample and in this case having a value from -32 to 0) was produced and is shown in table 11:

Number mode	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Displacement (*d*)	0	-2	-5	-9	-13	-17	-21	-26	-32	-26	-21	-17	-13	-9	-5	-2	0

Table 11: Number of angular mode and associated displacement.

The displacement parameters for the angular modes used in this case are:

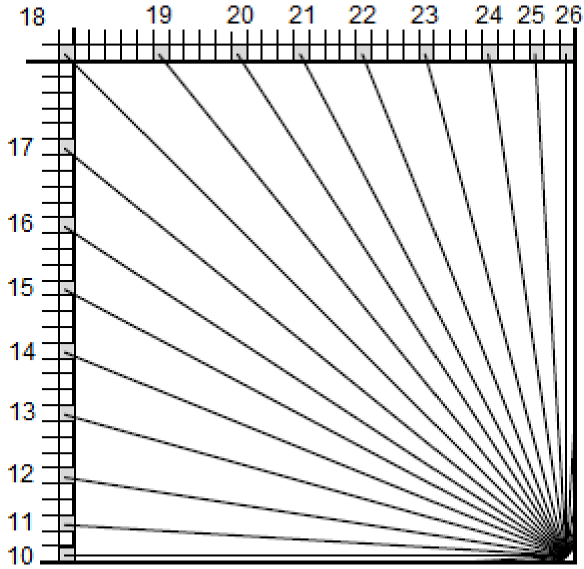


Figure 47: Available angular modes.

The angular modes between 10 and 18 have a horizontal directionality and the angular modes between 18 have a vertical directionality.

The prediction blocks ($P_{i,j}$) are calculated for all the angular modes (from 10 to 26), using the following formulas for each i,j sample (i indicates the row and j indicates de column) for a maximum size of N x N; finally *R* indicates the angular references samples.

First, the reference sample index (c_i) parameter and the weighting parameter (w_i) are calculated based on the projection displacement (d) and then the coordinate i; the index parameter is calculated based on the coordinate j and the c_i parameter.

$$\boxed{\begin{gathered} c_i = \text{floor}(i \cdot d/2^5) \\ w_i = bitand(abs(i \cdot d), 31) \\ index = j + c_i \end{gathered}} \tag{16}$$

These three expressions are always the same independently on the angular mode, because the angular reference samples change depending on the number of the angular mode.

It should be noted that the c_i and w_i parameters remain constant when calculating predictions for one line of samples within the prediction block.

For instance for a position of the sample in i = 2 and j = 3, the previous expressions are:

Number mode	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Displacement (d)	0	-2	-5	-9	-13	-17	-21	-26	-32	-26	-21	-17	-13	-9	-5	-2	0
c_i	0	0	0	0	0	-1	-1	-1	-2	-1	-1	-1	0	0	0	0	0
w_i	0	4	10	18	26	2	10	20	0	20	10	2	26	18	10	4	0
index	3	3	3	3	3	2	2	2	1	2	2	2	3	3	3	3	3

Table 12: An example of calculating parameters.

The prediction for each sample is obtained by the following expression depending on three different cases. Note that when the mode is less or equal to 17 the block is filled by j,i order ($P_{j,i}$), and when the mode is more or equal to 19 the block is filled by i,j order ($P_{i,j}$). Then:

$$if\ index > 0\ \rightarrow\ P = \Big((32 - w_y) \cdot R_{N+2+index} + w_y \cdot R_{N+1+index} + 16\Big)/2^5$$

$$elseif\ index = 0\ \rightarrow\ P = \Big((32 - w_y) \cdot R(1) + w_y \cdot R(N+3) + 16\Big)/2^5$$

$$if\ index < 0\ \rightarrow\ P = \Big((32 - w_y) \cdot R_{abs(index)+1} + w_y \cdot R_{abs(index)} + 16\Big)/2^5$$

For example, to predict the sample located in Z as shown in Figure 47, in a 4x4 block (N = 4), with the directional orientation depicted by the arrow, the mode 14 will use the reference samples in A and B. And the displacement parameter will be -13:

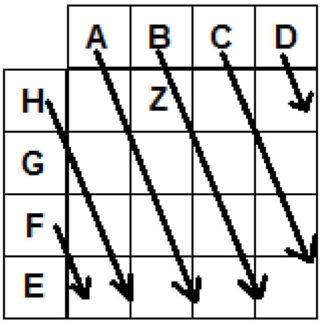


Figure 48: An example of angular orientation.

If for instance the reference samples are:

1	A	B	C	D	6	H	G	F	E	11
128	100	76	98	87	128	90	65	73	130	128

Table 13: An example of reference samples in angular mode.

Then the prediction block for this case, following the expression above is:

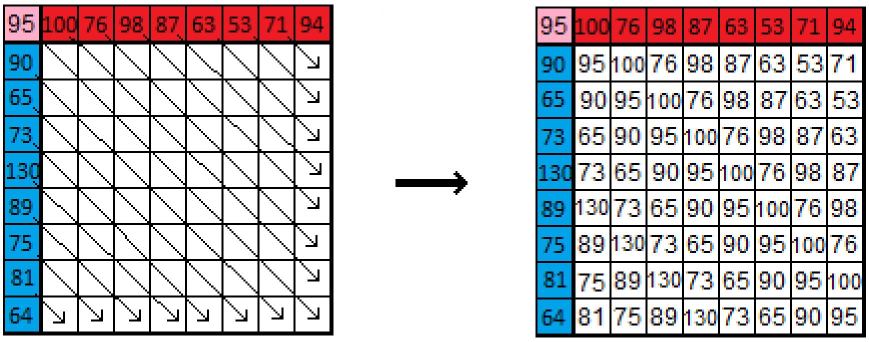
i,j	floor(c_i)	w_i	index	$P_{j,i}$
1,1	1*(-13)/32 = 0	(1*13) & 31 = 13	1 + 0 = 1	((32-13)*90+13*128+16)/32 = **105**
1,2	1*(-13)/32 = 0	(1*13) & 31 = 13	2 + 0 = 2	((32-13)*65+13*90+16)/32 = **75**
1,3	1*(-13)/32 = 0	(1*13) & 31 = 13	3 + 0 = 3	((32-13)*73+13*65+16)/32 = **70**
1,4	1*(-13)/32 = 0	(1*13) & 31 = 13	4 + 0 = 4	((32-13)*130+13*73+16)/32 = **107**
2,1	2*(-13)/32 = 0	(2*13) & 31 = 26	1 + 0 = 1	((32-26)*90+26*128+16)/32 = **121**
2,2	2*(-13)/32 = 0	(2*13) & 31 = 26	2 + 0 = 2	((32-26)*65+26*90+16)/32 = **85**
2,3	2*(-13)/32 = 0	(2*13) & 31 = 26	3 + 0 = 3	((32-26)*73+26*65+16)/32 = **67**
2,4	2*(-13)/32 = 0	(2*13) & 31 = 26	4 + 0 = 4	((32-26)*130+26*73+16)/32 = **84**
3,1	3*(-13)/32 = -1	(3*13) & 31 = 7	1 – 1 = 0	((32-7)*128+7*90+16)/32 = **120**
3,2	3*(-13)/32 = -1	(3*13) & 31 = 7	2 – 1 = 1	((32-7)*90+7*128+16)/32 = **98**
3,3	3*(-13)/32 = -1	(3*13) & 31 = 7	3 – 1 = 2	((32-7)*65+7*90+16)/32 = **70**
3,4	3*(-13)/32 = -1	(3*13) & 31 = 7	4 – 1 = 3	((32-7)*73+7*65+16)/32 = **71**
4,1	4*(-13)/32 = -1	(4*13) & 31 = 20	1 – 1 = 0	((32-20)*128+20*90+16)/32 = **104**
4,2	4*(-13)/32 = -1	(4*13) & 31 = 20	2 – 1 = 1	((32-20)*90+20*128+16)/32 = **114**
4,3	4*(-13)/32 = -1	(4*13) & 31 = 20	3 – 1 = 2	((32-20)*65+20*90+16)/32 = **81**
4,4	4*(-13)/32 = -1	(4*13) & 31 = 20	4 – 1 = 3	((32-20)*73+20*65+16)/32 = **68**

Table 14: An example of prediction samples for angular mode.

When the mode is equal to 18 (corresponding to pure diagonal direction), the prediction for each sample of the block is obtained as:

$$if\ index > 0 \rightarrow P_{j,i} = R_{index+1}$$
$$elseif\ index = 0 \rightarrow P_{j,i} = R_1$$
$$if\ index < 0 \rightarrow P_{j,i} = R_{N+index+2}$$

This mode is performed by direct extrapolation of the samples located in the upper-left corner, or w_i=0 on the above expression. For instance in the case of an 8x8 block:



95	100	76	98	87	63	53	71	94
90	95	100	76	98	87	63	53	71
65	90	95	100	76	98	87	63	53
73	65	90	95	100	76	98	87	63
130	73	65	90	95	100	76	98	87
89	130	73	65	90	95	100	76	98
75	89	130	73	65	90	95	100	76
81	75	89	130	73	65	90	95	100
64	81	75	89	130	73	65	90	95

Figure 49: An example of diagonal orientation.

Therefore, when the directionality points to integer samples (ie when $w_i = 32$ pure horizontal), the process is even simpler and consists of only copying integer reference samples from the reference row.

It should be noted that, in the case of samples at the boundaries, the reference row or column is extended by one sample using the value 128.

The residual block is then obtained as the difference between the original block and the prediction block, or:

```
residualAngularBlock = double(OriginalBlocks) - double(PredictionAngBlocks);
```

Finally the SAD (Sum of Absolute Difference) of the residual is calculated as:

```
SADmodeAngular = sum(sum(abs(residualAngularBlock)));
```

J. Comparison of SADs

After obtaining all the prediction blocks for each mode, a comparison between the SADs of each mode for each block is performed to obtain the best SAD for each block. The mode to predict a block is then selected as the mode corresponding to lowest SAD. In case the SAD of two or more modes is equal, the last mode is chosen.

K. Obtain best prediction and best residual

When the best SAD for a block is calculated, it is associated to it the best prediction block, so this step is performed by choosing the best prediction for the best SAD selected.

After obtaining the best prediction block, the best residual block is calculated by subtracting the original block to the best prediction block.

L. Transform, Quantise, Dequantise, Invtranform

The best residual block is transformed, obtaining the transformed coefficients for each block depending on the bit-depth that in this thesis is always set to 8 bits. In order to obtain the transform coefficients, a MATLAB implementation of HEVC DCT was used (transform.m) [28], which makes use of four different matrix transforms depending on the parameter SizeBlock as in conventional HEVC specifications. After this, the transform coefficients are quantised depending on the QP and bit-depth parameters [28].

Finally the quantised coefficients are dequantised and then are inverse transformed obtaining the reconstructed residuals for each block [28].

M. Reconstruct the block

The inverse transform block is added to the best prediction block in order to obtain the reconstruction block. This reconstructed block replaces the original block, so the next block that is going to be predicted uses this reconstruction as reference. This is done to avoid any propagation of errors at the decoder side. Note that in the proposed framework, the **TU and PU are always set to the same size**, namely the quadtree structure is not used.

In summary the following figure shows a diagram of the HEVC framework implemented:

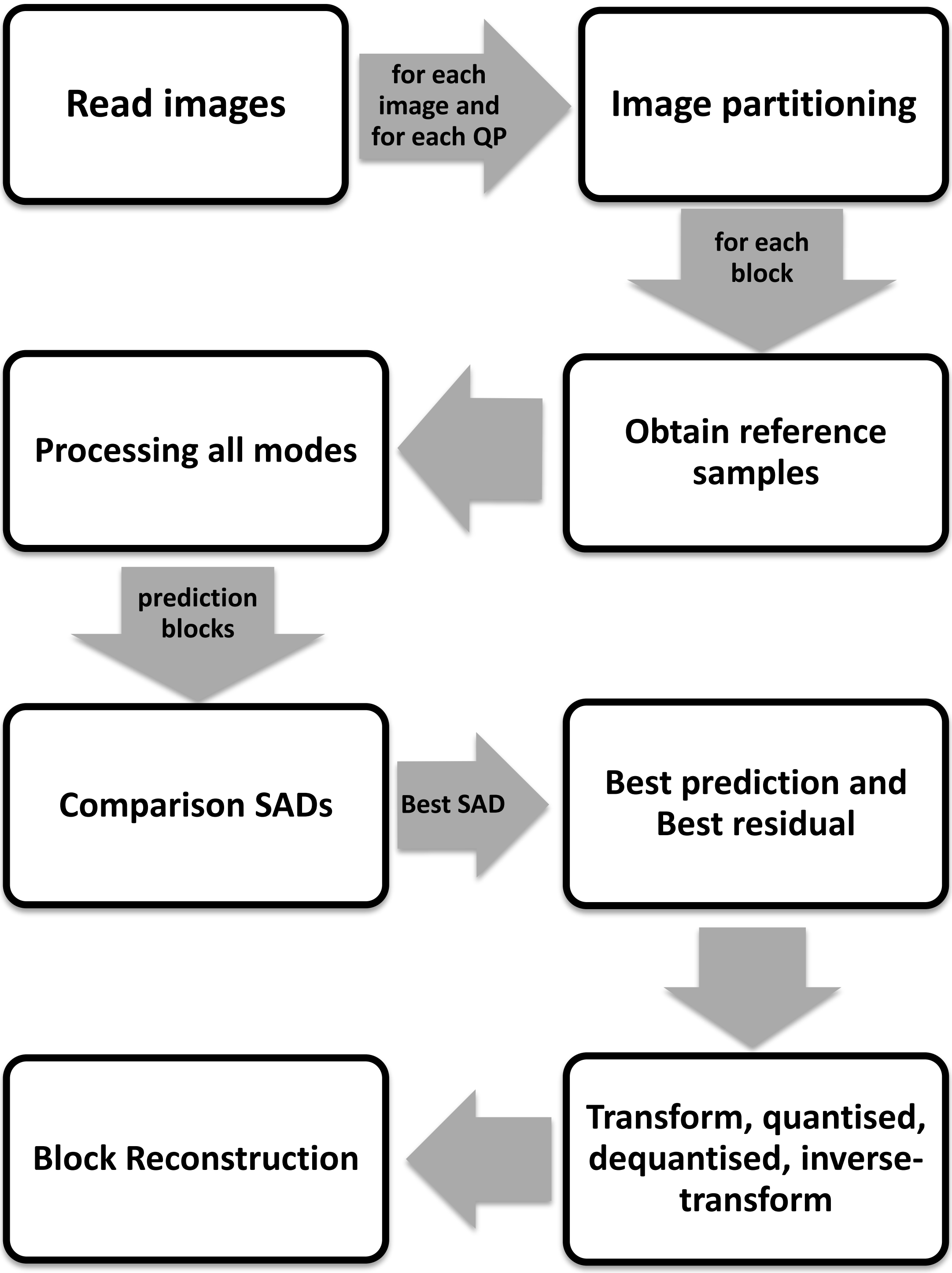


Figure 50: Diagram of the HEVC intra-prediction algorithm.

3.3. Proposed approach

In this section the proposed approach framework developed in MATLAB is described. It consists in a fast intra coding mode decision algorithm to avoid an exhaustive search among all possible intra modes that in this part are Mode DC, Planar Mode, Pure Horizontal Mode and Pure Vertical Mode. This is performed using binary classification.

3.3.1. Binary classification problems

Binary classification problems emerge when trying to separate two sets of data points, each corresponding to a given class, using a simple boundary based on discriminant functions. After finding the decision boundary, this is used to predict the class of a new point by checking on which side of the boundary it falls. The following figure shows a linear binary classifier which separates completely two sets of points in $\mathbb{R}^2$ [31] [32] [33].

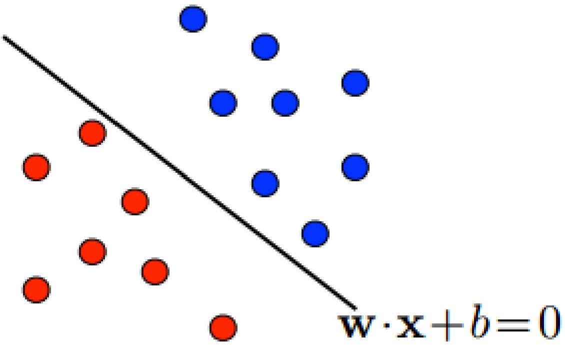


Figure 51: Linear binary Classification for separable data.

Each data point $\boldsymbol{x_i} \in \mathbb{R}^N$ in the classification problem is a vector that contains a numerical representation. It comes with a label $y_i \in \{-1, +1\}$ that is used to determine which class it belongs to. The linear binary classification problems involve a simple linear decision boundary, which can be described as the following expression:

$$\boxed{w \cdot x - b = 0} \text{ for some } w \in \mathbb{R}^N \text{ and } b \in \mathbb{R} \quad (17)$$

The decision boundary guarantees correct classification if all data points with $y_i = +1$ fall on one side and all the others on the other side or:

$$\boxed{y_i(w \cdot x - b) \geq 0, \quad i = 1, \dots, M} \quad (18)$$

- If the data is linearly separable, the following inequality is obtained:

$$\boxed{y_i(w \cdot x - b) \geq 1, \quad i = 1, \dots, M} \quad (19)$$

- If the data is not linearly separable for any line, some error (v_i) is added to the inequality.

$$\boxed{y_i(w \cdot x - b) \geq 1 - v_i, \quad i = 1, \dots, M} \quad for\ v_i \geq 0 \tag{20}$$

The following figure shows a linear binary classifier which does not separate completely two sets of points in $\mathbb{R}^2$.

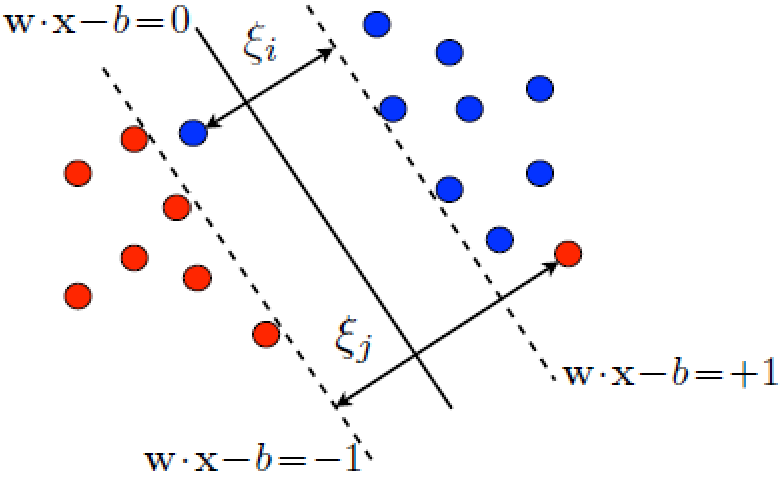

$\mathbf{w} \cdot \mathbf{x} - b = 0$
ξ_i
ξ_j
$\mathbf{w} \cdot \mathbf{x} - b = +1$
$\mathbf{w} \cdot \mathbf{x} - b = -1$

Figure 52: Linear binary Classification for non-separable data.

The goal now is to try to minimize the sum of the distances from the wrongly classified points (errors):

$$\boxed{\min_{w,b,v} \sum_{i=1}^{M} v_i} \quad for\ v_i \geq 0 \tag{21}$$

Once a classifier (a, b) is found, we can classify a new point $x \in \mathbb{R}^N$ by assigning to it the label:

$$\boxed{\hat{y} := \boldsymbol{sign}(w \cdot x - b)} \tag{22}$$

This procedure of training part and classification part is used in machine learning. Machine learning focuses on prediction, based on known properties learned from the training data. In this case, it is used the **Support Vector Machine (SMV)** is a discriminative classifier formally defined by a separating hyperplane.

As explained before, given a set of training data labelled as one of the two categories (supervised learning). The SVM training algorithm outputs an optimal hyperplane which categorizes new examples. The hyperplane divided the training data in both categories, so new examples are then mapped into that same space and predicted to belong to a category based on which side of the hyperplane they fall on.

The SVM algorithm is based on finding the hyperplane that gives the largest distance to the nearest training data of any class, so-called functional margin. Therefore, the optimal separating hyperplane maximizes the margin of the training data.

3.3.2. Summary of the proposed approach

First it should be mentioned that to evaluate the proposed approach, an HEVC framework was developed specifically for comparison, which contains the same modules as a conventional HEVC encoder, but only uses the Mode DC, Planar Mode, Pure Horizontal Mode and Pure Vertical Mode, obtaining the best SAD of this four modes for each block.

In this proposed approach the data points are some features extracted for each block of the original frame. The features extracted are the variance, the horizontal average difference and the vertical average difference of the samples of each block; these will be detailed in the rest of this thesis.

The features of the first three rows of blocks in a frame are used as training part of the **binary classification discriminant**. So in this case there are two different classes, one when the mode selected with the lowest SAD (obtained by HEVC framework for comparison) of the current block is the same as one of the classification modes, as shown in table 15 (label as 1, 3, or 4); the other (label as 0) when the mode selected with the lowest SAD is one of the other three modes.

CLASSIFICATION MODE	MODE
1	Mode DC
3	Pure Horizontal Mode
4	Pure Vertical Mode

Table 15: Classification modes.

All the remaining blocks in a frame are classified based on their features and according to the classification discriminant obtained during the training phase. This predicted classification is used to discriminate between encoding using just one mode and using the four modes in order to obtain the best prediction mode.

Different combinations of the features were tested in order to obtain the best results. An example of linear binary classifier of the horizontal and vertical average difference features for the first three rows of blocks in case of classification mode Pure Horizontal Mode (label as 3), QP=27, is shown in Figure 53.

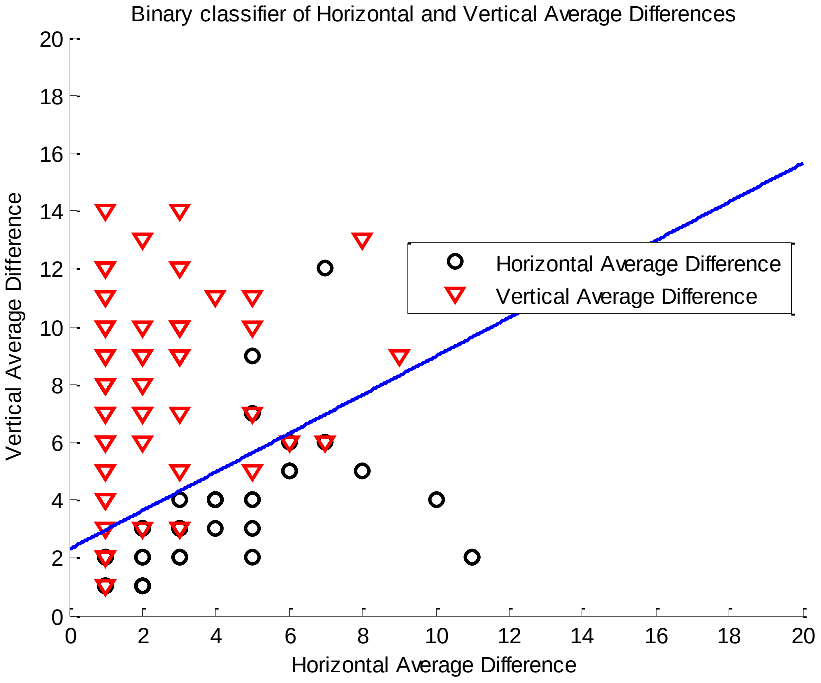


Figure 53: Binary classifier of the HAD and VAD features.

In summary the following figure shows a diagram of the proposed framework implemented:

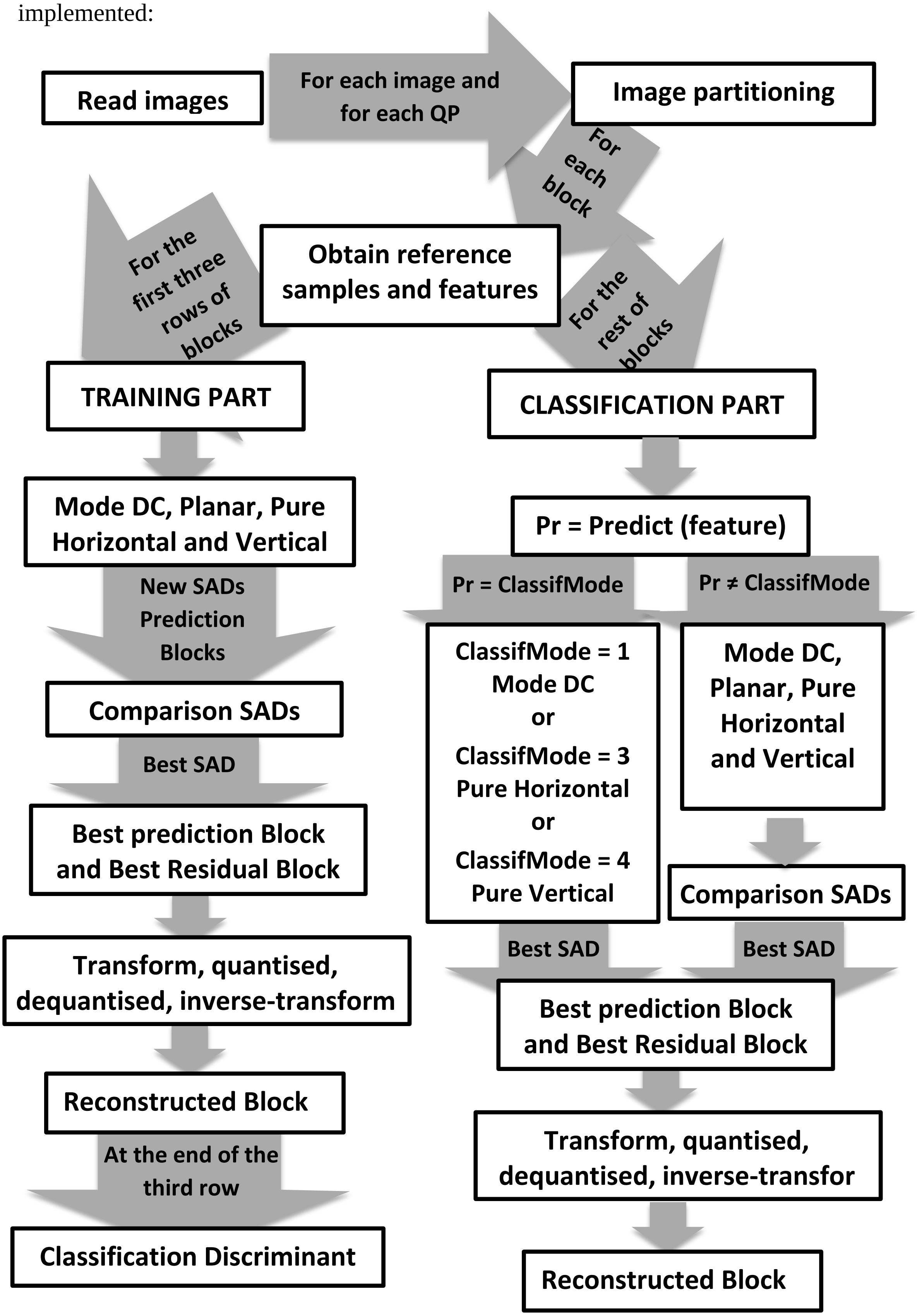

Read images
For each image and for each QP
Image partitioning
For each block
Obtain reference samples and features
For the first three rows of blocks
For the rest of blocks
TRAINING PART
CLASSIFICATION PART
Mode DC, Planar, Pure Horizontal and Vertical
New SADs Prediction Blocks
Comparison SADs
Best SAD
Best prediction Block and Best Residual Block
Transform, quantised, dequantised, inverse-transform
Reconstructed Block
At the end of the third row
Classification Discriminant
Pr = Predict (feature)
Pr = ClassifMode
Pr ≠ ClassifMode
ClassifMode = 1 Mode DC or ClassifMode = 3 Pure Horizontal or ClassifMode = 4 Pure Vertical
Mode DC, Planar, Pure Horizontal and Vertical
Comparison SADs
Best SAD
Best SAD
Best prediction Block and Best Residual Block
Transform, quantised, dequantised, inverse-transfor
Reconstructed Block

Figure 54: Diagram of the proposed algorithm.

3.3.3. Proposed approach

The algorithm was tested on the same 24 images in yuv format with a dimension of 768x512 (width and height respectively) sampled in 4:2:0 format as the HEVC framework for comparison. For each quantization parameters (QP = 22, 27, 32 and 37), each image is partitioned into blocks of a specified fixed size NxN (4x4, 8x8, 16x16 and 32x32). For instance if N = 16, the luminance of the two first images of the 24 Kodak test set images is:

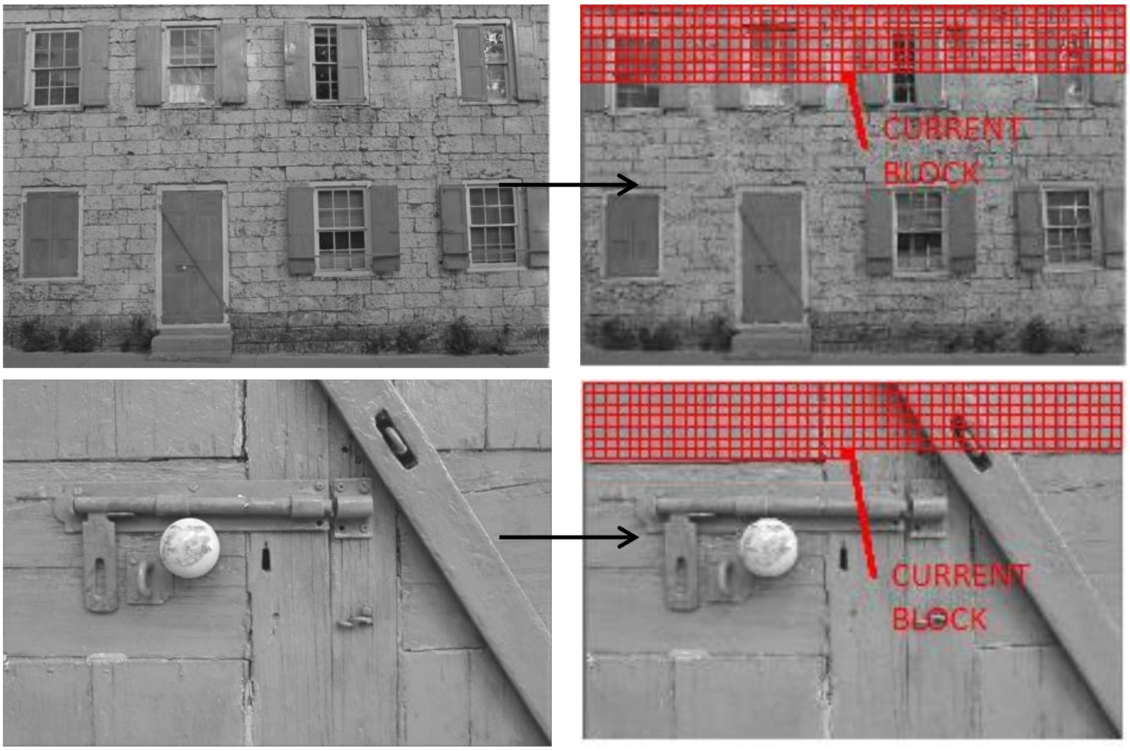


Figure 55: Partition of first and second Kodak test set images.

After obtaining the current block the reference samples and the features are extracted:

A. Reference Samples

This function is the same as the HEVC framework. The samples needed as reference are extracted from the top row and left column adjacent to the current block. There are four different cases to extract the reference samples depending on the current block location:

- **First case:** when the current block is in the top-left corner, the reference samples are:

Top row (1)	Left column (2)
1… …N	N+1… …(2*N)
128… …128	128… …128

- **Second case:** when the current block is in the first row, the reference samples are:

Top row (1)	Left column (2)
1… …N	N+1… …(2*N)
128… …128	$R_{n,0}$… …$R_{N,0}$

- **Third case:** when the current block is in the first column, the reference samples are:

Top row (1)	Left column (2)
1… …N	N+1… …(2*N)
$R_{0,n}$… …$R_{0,N}$	128… …128

- **Fourth case:** when the current block is located at one of the remaining positions, the reference samples are:

Top row (1)	Left column (2)
1… …N	N+1… …(2*N)
$R_{0,n}$… …$R_{0,N}$	$R_{n,0}$… …$R_{N,0}$

Table 16: Reference samples for proposed approach.

B. Features

For each block some features are extracted. These features are the variance, the horizontal average difference and the vertical average difference from the current block.

Features		
Variance	HorAveDif	VerAveDif

Table 17: Features for the current block.

➢ Variance (Var)

The variance is a statistic measure that measures the average difference between each sample and the mean of the original frame in an NxN block. This average is calculated by adding all the squared difference of each sample from the mean (in order to eliminate the negative signs), and dividing the result by the number of total samples in a block (NxN), as shown in equation (23).

$$\boxed{Var = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(OriginalBlock(i,j) - Mean)^2} \qquad (23)$$

➢ Horizontal average difference (HAD)

The Horizontal Average Difference measures the average difference between each sample (X) and the next horizontal sample (Y) of the original frame in an NxN block, as shown in Figure 56. This average is calculated by adding all the difference of each sample from the next horizontal sample, and dividing the result by the number of total samples in a block (NxN), as shown in equation (24).

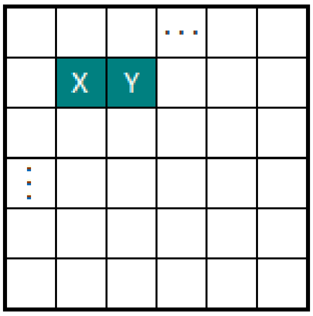


Figure 56: Horizontal Average Difference.

$$\boxed{HAD = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(OriginalBlock(i,j) - OriginalBlock(i,j+1))} \qquad (24)$$

➢ Vertical average difference (VAD)

The Vertical Average Difference measures the average difference between each sample (X) and the next vertical sample (Y) of the original frame in an NxN block, as shown in Figure 57. This average is calculated by adding all the difference of each sample from the next vertical sample, and dividing the result by the number of total samples in a block (NxN), as shown in equation (25).

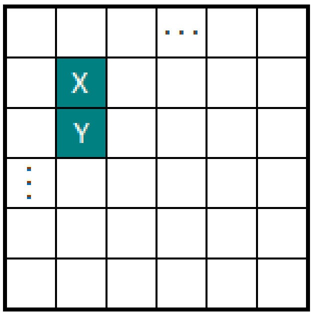


Figure 57: Vertical Average Difference.

$$\boxed{VAD = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(OriginalBlock(i,j) - OriginalBlock(i+1,j))^2} \qquad (25)$$

C. Training

The training is only performed for a predefined number of blocks; in this thesis, the first three rows of blocks are used for this purpose as shown in Figure 58. For instance, if the block size is N = 16, the blocks use for training will be 3 x 48=144 blocks.

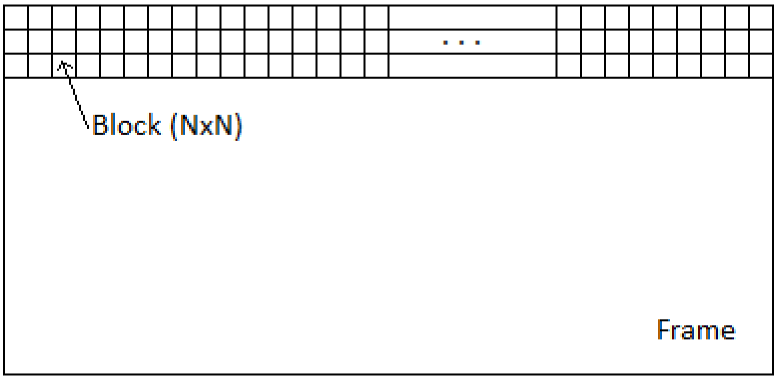


Figure 58: Blocks used for training part.

The prediction block and the SAD (Sum of Absolute Difference) are obtained for each of these blocks using the Mode DC, Planar Mode, Pure Horizontal Mode and Pure Vertical Mode. These modes are the same as the HEVC framework. Then each SAD for each mode is compared in order to choose the mode with lowest SAD. And then the best prediction block is selected for the best SAD. After this, the best residual block is calculated by subtracting the original block to the best prediction block.

The best residual block is transformed, obtaining the transformed coefficients for each block. After this, the transform coefficients are quantised depending on the quantised parameter (QP) and the bit-depth. Finally the quantised coefficients are dequantised and then are inverse transformed obtaining the reconstructed residuals for each block [28]. This inverse transform block is added to the best prediction block in order to obtain the reconstructed block.

The last step for each of these blocks is to classify the features, using the best mode selected with lowest SAD. The best mode selected can be one of the classification modes shown in table 18.

Classification mode	Mode
1	Mode DC
3	Pure Horizontal Mode
4	Pure Vertical Mode

Table 18: Classification modes.

If the best mode selected is equal to the classification mode, each feature is saved in a variable (classifVar, classifHor, classifVer), if not each feature is saved in other variable (classifRESTVar, classifRESTHor, classifRESTVer).

The next step is to combine all the features for the first three rows of encoded blocks in X and also save the class label for each feature in Y. There are different possible combinations of the features between the Variance, Horizontal Average Difference (HAD) and Vertical Average Difference (VAD). The combinations tested are shown in table 19.

Classification mode	**Possible combination of the features**		
1 - Mode DC	Variance	Variance + HAD	Variance + HAD + VAD
3 - Pure Horizontal Mode	HAD	HAD + VAD	Variance + HAD + VAD
4 - Pure Vertical Mode	VAD	HAD + VAD	Variance + HAD + VAD

Table 19: Possible combination of the features.

After the last training block is encoded, the training finishes and the classification begin for the subsequent blocks. Classification was implemented in a MATLAB function referred to as ClassificationDiscriminant.fit(X,Y). This function is a discriminant analysis classifier, which returns a discriminant analysis object based on the feature predictors X and class labels Y. Notice that in addition to a linear discriminant function, a quadratic discriminant function was also tested. This function instead of a linear decision boundary $(w \cdot x + b = 0)$ returns a decision boundary specified by the above expression:

$$\boxed{w \cdot x^2 + bx + c = 0} \text{ for some } w, b \in \mathbb{R}^N \text{ and } c \in \mathbb{R} \tag{26}$$

To train the MATLAB classifier, the fitting function estimates the parameters of a Gaussian mixture distribution for each class (Y). A Gaussian distribution is a normal probability function that tells the probability that any real observation will fall between any two real limits, as the curve approaches zero on either side. For linear discriminant analysis, the classifier has the same covariance matrix for each class, only the means vary. For quadratic discriminant analysis, both means and covariances of each class vary.

To predict the classes of new data, the trained classifier finds the class with the smallest misclassification cost, which is:

$$\boxed{\hat{y} = \underset{y=1,\dots,K}{\arg\min} \sum_{k=1}^{K} \hat{P}(k|x)C(y|k)} \tag{27}$$

where

$\hat{y}$ is the predicted classification.

K is the number of classes.

$\hat{P}(k|x)$ is the posterior probability of class *k* for observation *x*.

$C(y|k)$ is the cost of classifying an observation as *y* when its true class is *k*.

D. Classification

The classification is performed for the remaining blocks in a frame. For instance, if the block size is N = 16 and in case of images extracted from the Kodak test set images, the blocks used for classification will be (512/16 x 768/16) - (3x48) = 1392 blocks.

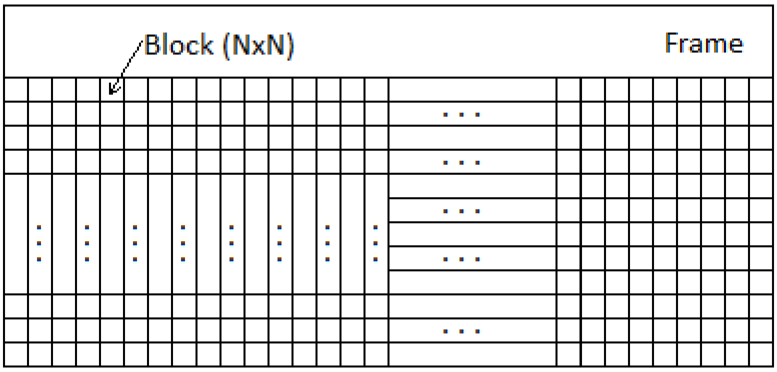


Figure 59: Blocks used for classification part.

The class for the current block is predicted using the MATLAB function predict. This function predicts the class for a new feature of the current block using the discriminant analysis object, which contains the data used for training.

Single Mode

Once the prediction is obtained and is equal to the Classification Mode, the current block is encoded using just one single mode, namely the mode selected by the Classification Mode.

- **When Classification Mode is selected as 1: Mode DC**. The prediction and the SAD are obtained using the same mode DC as the HEVC for comparison. These prediction block and SAD will be the best prediction block and best SAD, respectively for the current block.
- **When Classification Mode is selected as 3: Mode Pure Horizontal (PH).** The prediction and the SAD are obtained using the same mode Pure Horizontal as the HEVC for comparison. These prediction block and SAD will be the best prediction block and best SAD, respectively for the current block.
- **When Classification Mode is selected as 4: Mode Pure Vertical (PV)**. The prediction and the SAD are obtained using the same mode Pure Horizontal as the HEVC for comparison. These prediction block and SAD will be the best prediction block and best SAD, respectively for the current block.

Four Modes

On the other hand, if the prediction is not equal to the Classification Mode, the current block is encoded using the Mode DC, Planar Mode, Pure Horizontal Mode and Pure Vertical Mode. These modes are the same as the HEVC framework for comparison. The prediction block and the SAD are obtained for each of these modes. Then each SAD for each mode is compared in order to choose the mode with lowest SAD. The best mode is used to select the best prediction block.

After obtain the best SAD and the best prediction block in both cases, the best residual block is calculated by subtracting the original block to the best prediction block. The best residual block is transformed and quantised obtaining the transform quantised coefficients. Finally the quantised coefficients are dequantised and then are inverse transformed obtaining the reconstructed residuals for each block [28]. This inverse transform block is added to the best prediction block in order to obtain the reconstructed block.

4.

EXPERIMENTS AND RESULTS

4.1. Combination of features

Different combinations of the features have been performed in order to obtain the best results. The combinations tested between Variance (Var), Horizontal Average Difference (HAD) and Vertical Average Difference (VAD) for each mode are shown in table 20.

Classification mode	Possible combination of the features		
1 - Mode DC	Variance	Variance + HAD	Variance + HAD + VAD
3 - Pure Horizontal Mode	HAD	HAD + VAD	Variance + HAD + VAD
4 - Pure Vertical Mode	VAD	HAD + VAD	Variance + HAD + VAD

Table 20: Combination of the features tested.

In addition a quadratic discriminant function was also tested as mentioned before. The following results are obtained for a fixed block size of 16x16 and a bit depth of 8 bits. The amount of data is very large so only the results for QP = 22 and 37 and for the first two images of the Kodak test set are shown.

A confusion matrix has been used to measure the results of the binary classification. This matrix contains information about the actual value and the original value. The following table shows the confusion matrix for two class classifier.

		Original Value	
		Classification Mode	Class Label 0
Actual Value	Classification Mode	True Positive (TP)	False Negative (FN)
	Class label 0	False Positive (FP)	True Negative (TN)

Table 21: Confusion Matrix.

The entries in the confusion matrix have the following meaning in the context of this research:

- **True Positive (TP):** When the actual value is the classification mode and it should be the classification mode (Hit).
- **True Negative (TN):** When the actual value is one of the other modes and it should be one of the other modes (Hit).
- **False Positive (FP):** When the actual value is one of the other modes and it should be the classification mode (Miss).
- **False Negative (FN):** When the actual value is the classification mode and it should be one of the other modes (Miss).

Other measures of interest:

- **Total Number of Hits (TNH):** It is the sum of the True Positive and True Negative (TNH = TP + TN).
- **Total Number of Miss (TNM):** It is the sum of the False Positive and False Negative (TNM = FP + FN).

- **Total Number of CLASSIFICATION MODE predicted (TNCMP):** It is the sum True Positive and False Negative. (TNCMP = TP + FN).

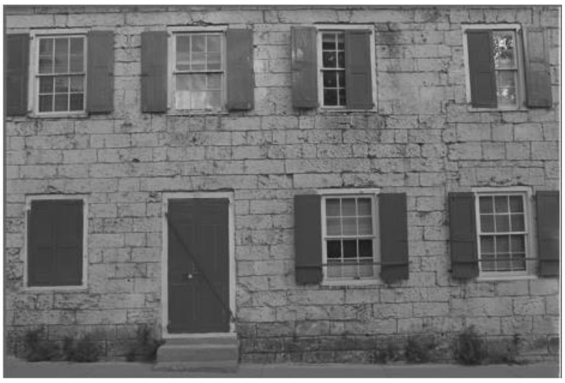
- **Total Number of CLASSIFICATION MODE in the best SAD (TNCMBS):** It is the sum of the times that the classification mode appears in the best SAD for the first three rows of blocks.

The results for the first two images of the Kodak test set are:

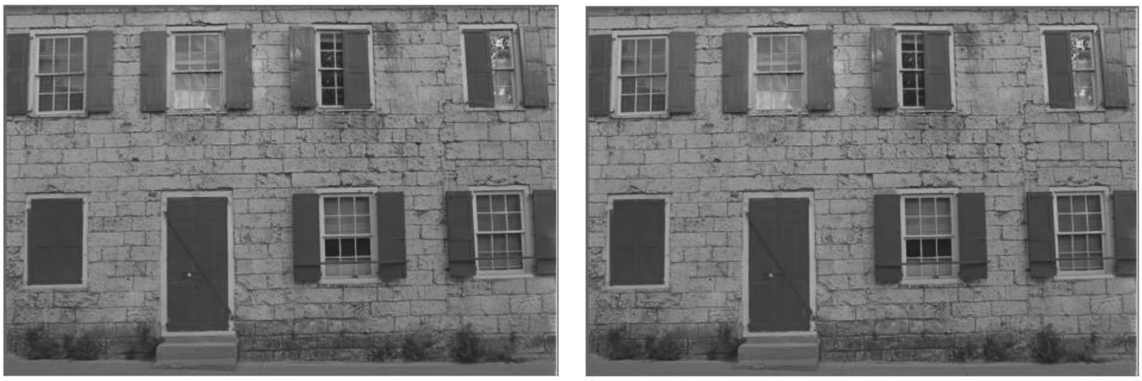
➢ FIRST IMAGE:

For instance, the original frame and the reconstructed frame using all the features for a QP of 22 and a linear decision boundary are:

Original image



Reconstructed frame using Mode DC as Classification Mode

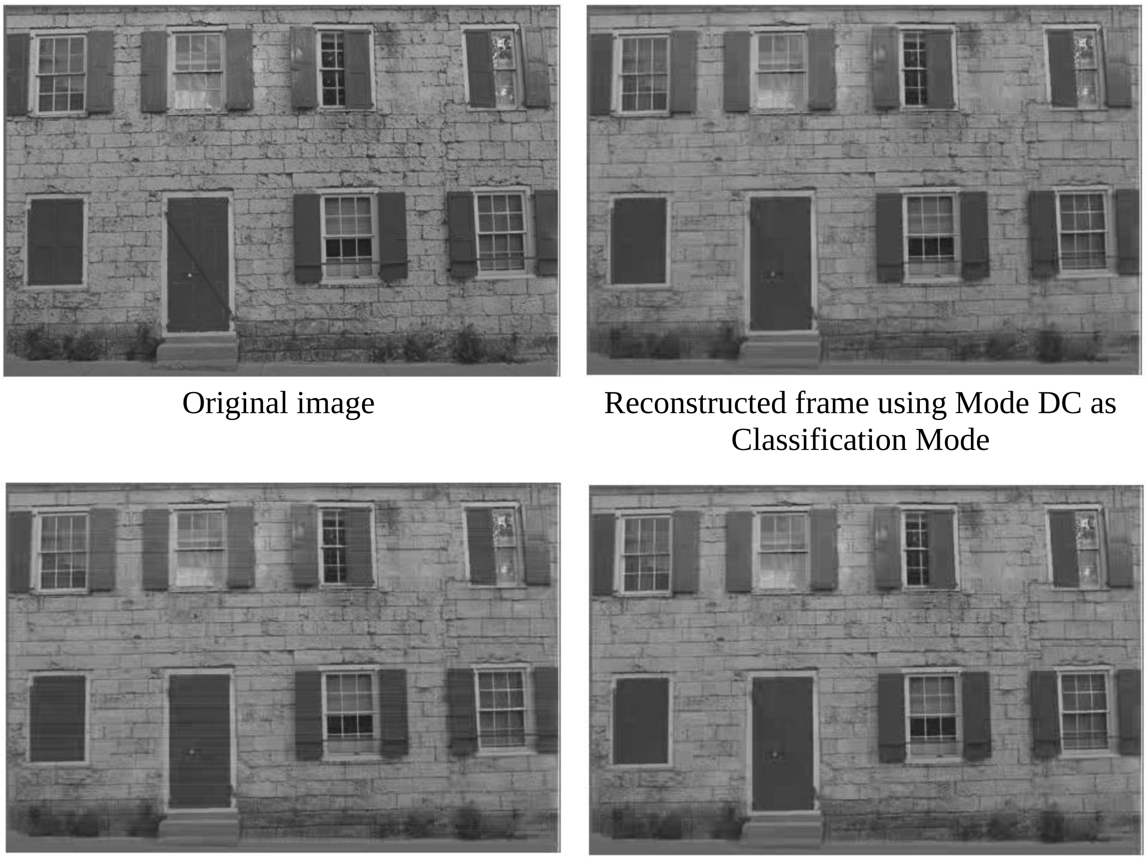


Reconstructed frame using Mode Pure Horizontal as Classification Mode

Reconstructed frame using Mode Pure Vertical as Classification Mode

Figure 60: First image of the Kodak test set with QP=22.

For instance, the original frame and the reconstructed frame using all the features for a QP of 37 and a linear decision boundary are:



Original image

Reconstructed frame using Mode DC as Classification Mode

Reconstructed frame using Mode Pure Horizontal as Classification Mode

Reconstructed frame using Mode Pure Vertical as Classification Mode

Figure 61: First image of the Kodak test set with QP=37.

When QP=37 the image is more degraded and it is easy to see the way that each mode predicts, for example in Figure 61, in the reconstructed frame using Mode Pure Horizontal, it is possible to see horizontal lines as a result of the prediction.

RESULTS WHEN CLASSIFICATION MODE = 1 CORRESPONDING TO MODE DC:

Linear Boundary Decision	QP = 22			QP = 37		
	Var	Var + HAD	Var + HAD + VAD	Var	Var + HAD	Var + HAD + VAD
TN1sP	**0**	**78**	**160**	**0**	**18**	**37**
TN1sBS	**347**	**347**	**347**	**237**	**237**	**237**
TP	0	34	81	0	0	10
TN	1045	1001	966	1155	1137	1128
TNH	**1045**	**1035**	**1047**	**1155**	**1137**	**1128**
FP	347	313	266	237	237	227
FN	0	44	79	0	18	27
TNM	**347**	**357**	**345**	**237**	**255**	**254**

Quadratic Boundary Decision	QP = 22			QP = 37		
	Var	Var + HAD	Var + HAD + VAD	Var	Var + HAD	Var + HAD + VAD
TN1sP	**0**	**166**	**335**	**0**	**0**	**158**
TN1sBS	**347**	**347**	**347**	**237**	**237**	**237**
TP	0	70	131	0	0	54
TN	1045	949	841	1155	1155	1051
TNH	**1045**	**1019**	**972**	**1155**	**1155**	**1051**
FP	347	277	216	237	237	183
FN	0	96	204	0	0	104
TNM	**347**	**373**	**420**	**237**	**237**	**287**

Table 22: Results first image MODE DC with linear and quadratic function.

RESULTS WHEN CLASSIFICATION MODE = 3 CORRESPONDING TO MODE PH:

Linear Boundary Decision	QP = 22			QP = 37		
	HAD	HAD + VAD	Var + HAD + VAD	HAD	HAD + VAD	Var + HAD + VAD
TN3sP	**1246**	**1159**	**1009**	**1246**	**1217**	**1094**
TN3sBS	**568**	**568**	**568**	**685**	**607**	**607**
TP	536	530	448	561	566	488
TN	114	195	263	100	134	179
TNH	**650**	**725**	**711**	**661**	**700**	**667**
FP	32	38	120	46	41	119
FN	710	629	561	685	651	606
TNM	**742**	**667**	**681**	**731**	**692**	**725**

Quadratic Boundary Decision	**QP = 22**			**QP = 37**		
	HAD	**HAD + VAD**	**Var + HAD + VAD**	**HAD**	**HAD + VAD**	**Var + HAD + VAD**
TN3sP	**1183**	**1196**	**907**	**1246**	**1229**	**974**
TN3sBS	**568**	**568**	**568**	**607**	**607**	**607**
TP	519	530	422	561	563	466
TN	160	158	339	100	119	277
TNH	**679**	**688**	**761**	**661**	**682**	**743**
FP	49	38	146	46	44	141
FN	664	666	485	685	666	508
TNM	**713**	**704**	**631**	**731**	**710**	**649**

Table 23: Results first image MODE PH with linear and quadratic function.

RESULTS WHEN CLASSIFICATION MODE = 4 CORRESPONDING TO MODE PV:

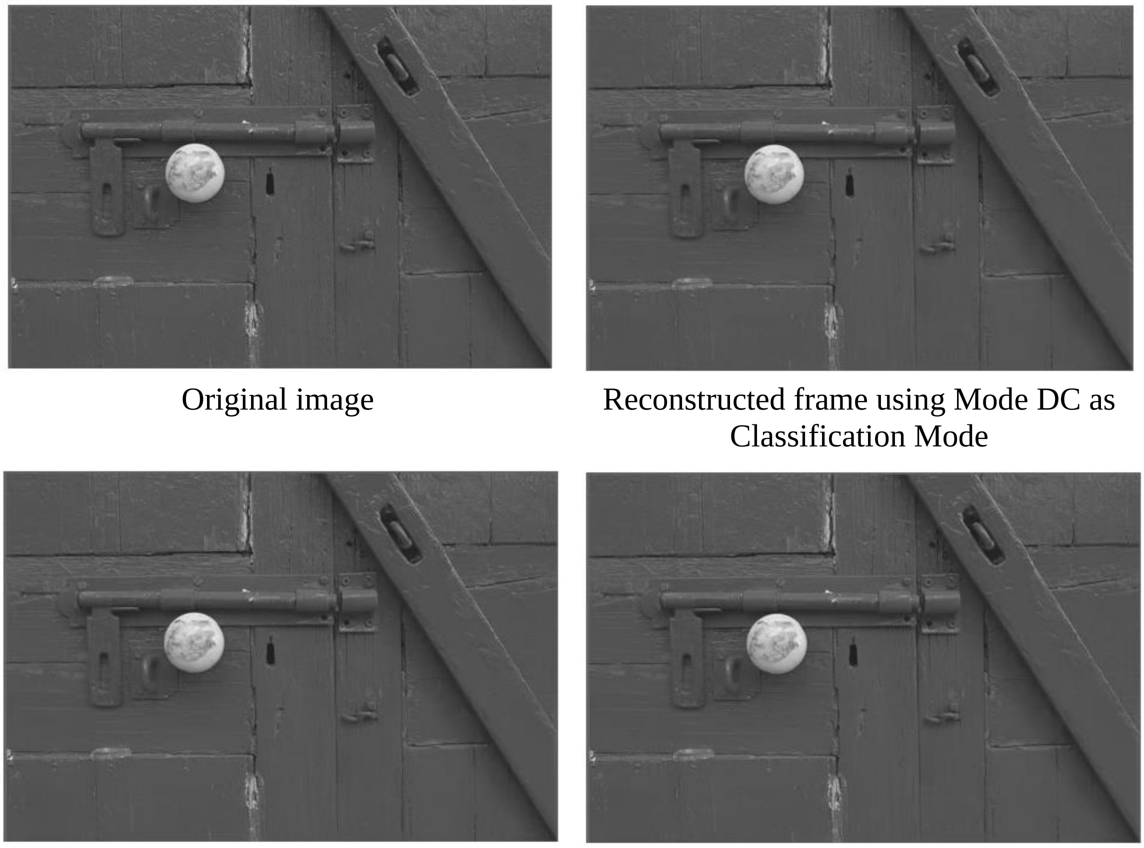
Linear Boundary Decision	**QP = 22**			**QP = 37**		
	VAD	**HAD + VAD**	**Var + HAD + VAD**	**VAD**	**HAD + VAD**	**Var + HAD + VAD**
TN4sP	**0**	**21**	**45**	**0**	**68**	**80**
TN4sBS	**383**	**383**	**383**	**446**	**446**	**446**
TP	0	19	43	0	61	71
TN	1009	1007	1007	946	939	937
TNH	**1009**	**1026**	**1050**	**946**	**1000**	**1008**
FP	383	364	340	446	385	375
FN	0	2	2	0	7	9
TNM	**383**	**366**	**342**	**446**	**392**	**384**

Quadratic Boundary Decision	**QP = 22**			**QP = 37**		
	VAD	**HAD + VAD**	**Var + HAD + VAD**	**VAD**	**HAD + VAD**	**Var + HAD + VAD**
TN4sP	**0**	**51**	**122**	**0**	**50**	**111**
TN4sBS	**383**	**383**	**383**	**446**	**446**	**446**
TP	0	46	111	0	40	98
TN	1009	1004	998	946	936	933
TNH	**1009**	**1050**	**1109**	**946**	**975**	**1031**
FP	383	337	272	446	406	348
FN	0	5	11	0	10	13
TNM	**383**	**342**	**283**	**446**	**416**	**361**

Table 24: Results first image MODE PV with linear and quadratic function.

➢ <u>SECOND IMAGE:</u>

For instance the original frame and the reconstructed frame using all the features for a QP of 22 and a linear decision boundary are:



Original image

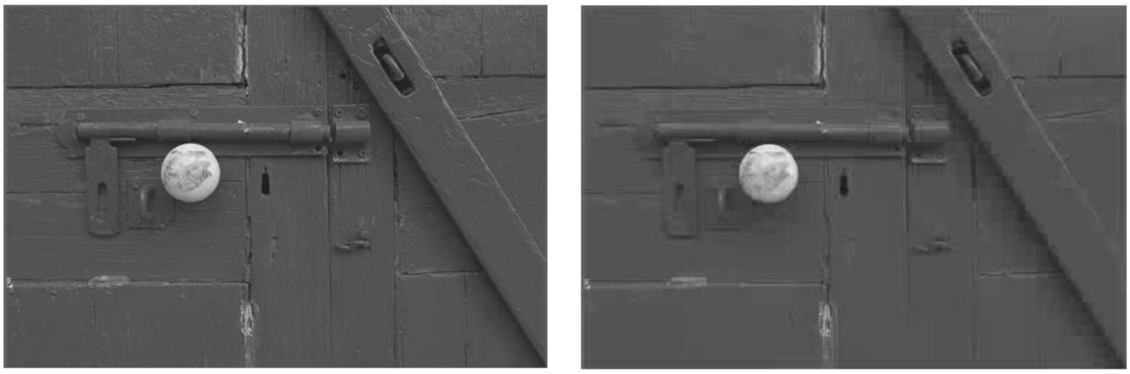
Reconstructed frame using Mode DC as Classification Mode

Reconstructed frame using Mode Pure Horizontal as Classification Mode

Reconstructed frame using Mode Pure Vertical as Classification Mode

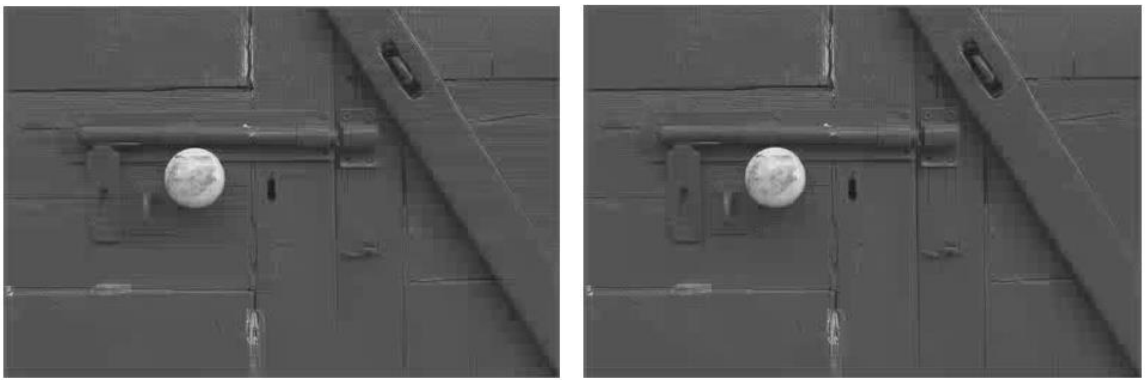
Figure 62: Second image of the Kodak test set with QP=22.

For instance, the original frame and the reconstructed frame using all the features for a QP of 37 and a linear decision boundary are:



Original image

Reconstructed frame using Mode DC as Classification Mode



Reconstructed frame using Mode Pure Horizontal as Classification Mode

Reconstructed frame using Mode Pure Vertical as Classification Mode

Figure 63: Second image of the Kodak test set with QP=37.

Results when CLASSIFICATION MODE = 1 corresponding to MODE DC:

Linear Boundary Decision	**QP = 22**			**QP = 37**		
	Var	**Var + HAD**	**Var + HAD + VAD**	**Var**	**Var + HAD**	**Var + HAD + VAD**
TN1sP	**152**	**14**	**0**	**0**	**0**	**0**
TN1sBS	**522**	**522**	**522**	**252**	**252**	**252**
TP	51	4	0	0	0	0
TN	769	860	870	1140	1140	1140
TNH	**820**	**864**	**870**	**1140**	**1140**	**1140**
FP	471	518	522	252	252	252
FN	101	10	0	0	0	0
TNM	**572**	**528**	**522**	**252**	**252**	**252**

Quadratic Boundary Decision	**QP = 22**			**QP = 37**		
	Var	**Var + HAD**	**Var + HAD + VAD**	**Var**	**Var + HAD**	**Var + HAD + VAD**
TN1sP	**1035**	**720**	**542**	**0**	**0**	**171**
TN1sBS	**522**	**522**	**522**	**252**	**252**	**252**
TP	452	351	258	0	0	21
TN	287	501	586	1140	1140	990
TNH	**739**	**852**	**844**	**1140**	**1140**	**1011**
FP	70	171	264	252	252	264
FN	583	369	284	0	0	284
TNM	**653**	**540**	**548**	**252**	**252**	**548**

Table 25: Results second image MODE DC with linear and quadratic function.

RESULTS WHEN CLASSIFICATION MODE = 3 CORRESPOND TO MODE PH:

Linear Boundary Decision	**QP = 22**			**QP = 37**		
	HAD	**HAD + VAD**	**Var + HAD + VAD**	**HAD**	**HAD + VAD**	**Var + HAD + VAD**
TN3sP	**0**	**340**	**414**	**1236**	**947**	**785**
TN3sBS	**427**	**427**	**427**	**474**	**474**	**474**
TP	0	229	265	425	383	357
TN	965	854	816	107	354	490
TNH	**965**	**1083**	**1081**	**532**	**737**	**847**
FP	427	198	162	49	91	117
FN	0	111	149	811	564	428
TNM	**427**	**309**	**311**	**860**	**655**	**545**

Quadratic Boundary Decision	**QP = 22**			**QP = 37**		
	HAD	HAD + VAD	Var + HAD + VAD	HAD	HAD + VAD	Var + HAD + VAD
TN3sP	**0**	**314**	**503**	**1025**	**788**	**1214**
TN3sBS	**427**	**427**	**427**	**474**	**474**	**474**
TP	0	211	275	358	339	409
TN	965	862	737	251	469	113
TNH	**965**	**1073**	**1012**	**609**	**808**	**522**
FP	427	216	152	116	135	65
FN	0	103	228	667	449	805
TNM	**427**	**319**	**380**	**783**	**584**	**870**

Table 26: Results second image MODE PH with linear and quadratic function.

RESULTS WHEN CLASSIFICATION MODE = 4 CORRESPONDING TO MODE PV:

Linear Boundary Decision	**QP = 22**			**QP = 37**		
	VAD	**HAD + VAD**	**Var + HAD + VAD**	**VAD**	**HAD + VAD**	**Var + HAD + VAD**
TN4sP	**0**	**86**	**139**	**0**	**111**	**224**
TN4sBS	**340**	**340**	**340**	**601**	**601**	**601**
TP	0	56	83	0	61	121
TN	1052	1022	996	791	741	688
TNH	**1053**	**1078**	**1079**	**791**	**802**	**809**
FP	340	284	257	601	540	480
FN	0	30	56	0	50	103
TNM	**340**	**314**	**313**	**601**	**590**	**583**

Quadratic Boundary Decision	QP = 22			QP = 37		
	VAD	HAD + VAD	Var + HAD + VAD	VAD	HAD + VAD	Var + HAD + VAD
TN4sP	**77**	**94**	**153**	**301**	**43**	**104**
TN4sBS	**340**	**340**	**340**	**601**	**601**	**601**
TP	5	57	75	166	18	42
TN	980	1015	974	656	766	729
TNH	**985**	**1072**	**1049**	**822**	**784**	**771**
FP	335	283	265	435	583	559
FN	72	37	78	135	25	62
TNM	**407**	**320**	**343**	**570**	**608**	**621**

Table 27: Results second image MODE PV with linear and quadratic function.

First, it should be mentioned that the previous results are different for each image depending on the content of the image.

Analysing the above results it is obtained that the number of true positive and the total number of hits and miss is usually better using three features than using just one feature. And comparing those numbers between two and three features it is discovered that using three features it is also usually better than using two features.

Comparing the results between the linear and quadratic decision boundary for the three features in terms of the TNH, TNM and TP; both behaviours are interesting depending on the application but in this case the linear decision boundary is chosen.

Also as we can see in the previous figures, the Figures 61 and 63 are more degraded losing quality of the image because the parameter (QP = 37) is bigger than the parameter QP in Figures 60 and 62 (QP = 22). For three features, it is usually higher the number of TP with a QP =37 than QP = 22; and it is usually higher the number of TNH (TNM) with a QP = 22 than a QP = 37.

In conclusion using three features (Variance, Horizontal Average Difference and Vertical Average Difference) and linear boundary decision is the best case. In the following section an objective video quality evaluation is performed by two videos for the HEVC framework for comparison and the proposed approach framework. Two or three classification features with a linear decision boundary is used in order to corroborate the above conclusion.

4.2. Objective Video Quality Evaluation

After obtain the previous results, the approach was tested on encoding two video sequences in yuv format, namely BQSquare_416x240.yuv (video 1) and BasketballPass_416x240.yuv (video 2) in order to analyse the coding performance. Only the first 50 frames in each video are considered, where each frame has a size of 426x240 (width and height respectively). It also used a size block of 16x16 and a bit depth of 8 bits.

4.2.1. Sum of Absolute Difference

The Sum of Absolute Differences (SAD) is a measure of the similarities between frame blocks. In this case measure the energy of the residual block by taking the summed of absolute difference between the original block and the prediction block of a frame.

$$\boxed{SAD = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}|OriginalBlock(i,j) - PredictionBlock(i,j)|} \tag{28}$$

This measure is calculated in order to choose the mode with lowest SAD, called best SAD. In the following table are shown the results of the average for all 50 frames, of the average of the best SADs of a frame for the original HEVC framework for comparison and the proposed approach for the three different classification modes. These results are obtained for two different decision boundaries and for just two and three combination of the features corresponding with table 20.

					SAD (Sum of Absolute Difference)			
					QP = 22	QP = 27	QP = 32	QP = 37
Video 1(BQSquare)	**HEVC framework**				**6252**	**6273.6**	**6334.8**	**6453.4**
	Proposed approach framework	**Two features**	**Linear Decision boundary**	**Mode DC**	6256.2	6277.4	6338.4	6456.8
				Mode PH	6670.8	6726.4	6796.8	6897.4
				Mode PV	6347.2	6366.4	6423.2	6535.8
			Quadratic Decision boundary	**Mode DC**	6273.2	6289.2	6349.4	6465.6
				Mode PH	6794.8	6831	6902	7029.6
				Mode PV	6362	6385.6	6437.6	6548.4
		Three features	**Linear Decision boundary**	**Mode DC**	6273.6	6292.2	6347.6	6465.6
				Mode PH	6602.6	6648	6716.4	6838
				Mode PV	6336	6355	6409.8	6523.8
			Quadratic Decision boundary	**Mode DC**	6302.6	6319.6	6368.2	6495.4
				Mode PH	6945	6957.2	7021.2	7174.2
				Mode PV	6320.4	6340.8	6392.8	6505

					SAD (Sum of Absolute Difference)			
					QP = 22	QP = 27	QP = 32	QP = 37
Video 2 (BasketBall)	HEVC framework				2322	2352	2404.2	2491.2
	Proposed approach framework	Two features	Linear Decision boundary	Mode DC	2322	2361.4	2414.4	2498.2
				Mode PH	2536.4	2538.2	2586.6	2679.6
				Mode PV	2444.4	2465.4	2495.4	2565.6
			Quadratic Decision boundary	Mode DC	2460.6	2481	2497	2559
				Mode PH	2820.8	2840	2841.6	2952.8
				Mode PV	2479.2	2502.8	2551.6	2640.4
		Three features	Linear Decision boundary	Mode DC	2325.6	2361	2413.8	2501
				Mode PH	2501.6	2476.4	2538.8	2641
				Mode PV	2466.2	2481.6	2504.6	2539.8
			Quadratic Decision boundary	Mode DC	2462.2	2484.6	2501.4	2563
				Mode PH	3023.6	3024.2	3019	3099.2
				Mode PV	2434.8	2464.4	2517.8	2566.2

Table 28: Sum of Absolute Difference.

The average of SADs is bigger as the parameter QP increased. So the difference between the original block and the prediction block is bigger, there are less similar, as the parameter QP increased.

For both videos, the classification mode DC is the mode with lowest average of SADs then classification mode Pure Vertical and finally classification mode Pure Horizontal. The average of SADs of classification mode DC is very similar to the average of SADs of conventional HEVC framework, because the binary classifier mostly of the times predicts class 0.

Also for both videos, the average of SADs most similar to the average of SADs of the original HEVC framework is when classification mode is Mode DC using a combination of two features and linear decision boundary. But comparing each group of modes for the different cases, the case with the lowest average of SADs similar to the average of SADs of conventional HEVC framework is the group of classification modes when the decision boundary is linear and the combination of features used is three (Var, HAD and VAD).

4.2.2. Number of Operations

In this case, the number of operations is a measure of the number of times the SAD is compute in a framework for a given video content.

The average number of operations that the SAD is computed in conventional HEVC framework for a video:

4 modes x Num of blocks (15 x 26) = 1560 operations

The average number of operations that the SAD is computed in the proposed approach framework for each QP and for a video:

4 modes x Number of blocks for the Three first row (3 x 26) +
1 mode x Number of times single mode is chosen +
4 modes x Number of times all modes is chosen

In the following table are the results of the average number of operations of the SAD of each QP and all frames for the original HEVC framework for comparison and the proposed approach for the three different classification modes. These results are obtained for two different decision boundaries and for two and three combination of the features corresponding with table 20.

					Number of operations			
					QP = 22	QP = 27	QP = 32	QP = 37
Video 1 (BQSquare)	HEVC conventional framework				1560 for each QP			
	Proposed approach framework	Two features	Linear Decision boundary	Mode DC	1553.76	1553.88	1554.72	1555.86
				Mode PH	1118.16	1108.44	1112.1	1115.9
				Mode PV	1513.8	1514.1	1513.38	1518
			Quadratic Decision boundary	Mode DC	1533.36	1538.88	1541.1	1545.6
				Mode PH	1072.92	1071.48	1081.8	1093.26
				Mode PV	1508.1	1503.8	1506.2	1513
		Three features	Linear Decision boundary	Mode DC	1539.7	1541.2	1546.6	1548.3
				Mode PH	1191.3	1173.2	1174.3	1202.6
				Mode PV	1515.7	1515.5	1516	1519.7
			Quadratic Decision boundary	Mode DC	1505.3	1513.6	1526.6	1513.6
				Mode PH	988.98	985.2	1004.6	1010.1
				Mode PV	1518.1	1515.2	1520	1524.8

Video 2 (BasketBall)	**HEVC conventional framework**				**1560 for each QP**			
	Proposed approach framework	**Two features**	**Linear Decision boundary**	**Mode DC**	1560	1550.94	1552.14	1553.46
				Mode PH	1391.22	1399.74	1398.72	1399.74
				Mode PV	1395.9	1403.16	1402.33	1424.76
			Quadratic Decision boundary	**Mode DC**	1076.52	1091.1	1170.96	1301.22
				Mode PH	1342.4	1341.3	1351.3	1344.3
				Mode PV	1353.6	1355.7	1341.3	1344.8
		Three features	**Linear Decision boundary**	**Mode DC**	1545.6	1548.5	1552.5	1548.4
				Mode PH	1411.2	1428	1425.8	1426.8
				Mode PV	1361.9	1383.7	1393	1474.5
			Quadratic Decision boundary	**Mode DC**	1056.8	1066.4	1134.8	1242.2
				Mode PH	1259.9	1266.4	1281.3	1272
				Mode PV	1385.9	1396.9	1357.2	1420.7

Table 29: Number of operations for each QP.

In the following table are the results of the number of operation for all the videos and all QP for the different cases.

	Number of operations: HEVC conventional framework: 1560				
	Number of operations: Proposed approach framework				
Video 1 (BQSquare)	**Features**	**Decision Boundary**	**Mode DC**	**Mode PH**	**Mode PV**
	Two features	Linear	1554.5	1113.6	1514.8
		Quadratic	1539.7	1079.8	**1507.7**
	Three features	Linear	1543.9	1185.3	1516.7
		Quadratic	**1514.7**	**997.2**	1519.5
	Number of operations: HEVC conventional framework: 1560				
	Number of operations: Proposed approach framework				
Video 2 (BasketBall)	**Features**	**Decision Boundary**	**Mode DC**	**Mode PH**	**Mode PV**
	Two features	Linear	1554.1	1397.3	1406.5
		Quadratic	1159.9	1344.8	**1348.8**
	Three features	Linear	1548.7	1422.9	1403.2
		Quadratic	**1125.05**	**1269.9**	1390.2

Table 30: Elapsed Time.

For each block, the number of operations computing the SAD for all modes is four and for a single mode is just one. As it is expected the number of SAD computation in the proposed approach is lower than the number of operation of the SAD in conventional HEVC framework.

For both videos, the less number of operations is performed by the quadratic decision boundary and a combination of three features for classification mode DC and Pure Horizontal, and for the other classification mode the less number of operations is performed by the quadratic decision boundary and a combination of two features for both videos.

An average for both videos and for all the classification modes is performed in order to obtain a percentage that show the number of operations saved using the proposed approach compared to the conventional HEVC implemented.

Features	Decision Boundary	Percentage
Two features	Linear	8.75 %
	Quadratic	14.74 %
Three features	Linear	7.89 %
	Quadratic	**16.49 %**

Table 31: Percentage saved of operations.

This approach is capable of encoding the two videos using in average 16.5% less number of operations than conventional HEVC for a quadratic decision boundary and a combination of three features.

4.2.3. Peak Signal to Noise Ratio

Peak Signal to Noise Ratio (PSNR) is the most commonly used measure to quantify quality between the original frame and the reconstructed frame of lossy compression codecs. It measures on a logarithmic scale the mean squared error (MSE) between the original and the reconstructed video frame, relative to the square of the highest-possible signal value in the frame ($(2^{\text{bit-depth}}$ $-1)^2$, in this case 255^2).

The PSNR in dB is defined as:

$$\boxed{PSNR = 10\log_{10}\frac{\left(2^{bit-depth}-1\right)^2}{MSE}} \tag{29}$$

Where MSE is the Mean Square Error defined as:

$$\boxed{MSE = \frac{1}{M\cdot N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(OriginalBlock(i,j) - ReconstructedBlock(i,j))^2} \tag{30}$$

In the following table are the results of the average PSNR for the original HEVC framework for comparison and for the proposed approach for the three different classification modes. These results are obtained for two different decision boundaries and for two and three combination of the features corresponding with table 20.

					PSNR (dB)			
					QP = 22	**QP = 27**	**QP = 32**	**QP = 37**
Video 1(BQSquare)	**HEVC framework**				35.88	31.786	27.976	24.65
	Proposed approach framework	**Two features**	**Linear Decision boundary**	**Mode DC**	35.88	31.786	27.976	24.65
				Mode PH	35.868	31.762	27.94	24.608
				Mode PV	35.88	31.786	27.976	24.65
			Quadratic Decision boundary	**Mode DC**	35.88	31.786	27.978	24.65
				Mode PH	35.87	31.758	27.936	24.594
				Mode PV	35.88	31.786	27.976	24.65
		Three features	**Linear Decision boundary**	**Mode DC**	35.88	31.786	27.976	24.65
				Mode PH	35.872	31.77	27.954	24.624
				Mode PV	35.88	31.786	27.976	24.65
			Quadratic Decision boundary	**Mode DC**	35.882	31.786	27.976	24.65
				Mode PH	35.864	31.754	27.926	24.58
				Mode PV	35.88	31.786	27.976	24.65

					PSNR (dB)			
					QP = 22	QP = 27	QP = 32	QP = 37
Video 2 (BasketBall)	HEVC framework				38.578	35.256	32.128	29.468
	Proposed approach framework	Two features	Linear Decision boundary	Mode DC	39.178	35.256	32.13	29.468
				Mode PH	38.578	35.258	32.124	29.464
				Mode PV	38.576	35.256	32.124	29.456
			Quadratic Decision boundary	Mode DC	38.592	35.272	32.132	29.47
				Mode PH	38.572	35.252	32.124	29.464
				Mode PV	38.574	35.254	32.12	29.442
		Three features	Linear Decision boundary	Mode DC	38.578	35.256	32.13	29.468
				Mode PH	38.578	35.258	32.122	29.46
				Mode PV	38.576	35.254	32.124	29.462
			Quadratic Decision boundary	Mode DC	38.594	35.27	32.132	29.468
				Mode PH	38.566	35.242	32.112	29.45
				Mode PV	38.574	35.254	32.116	29.45

Table 32: PSNR.

For a given image, high PSNR usually indicates high quality and low PSNR usually indicates low quality. The PSNR of the proposed approach usually is lower than the PSNR of the HEVC framework indicating that the proposed approach has lower quality. The PSNR decrease as the parameter QP increase, because the quality of the frame decreases.

The PSNR for each case of the proposed approach is really similar to the PSNR of the HEVC framework, so it is difficult to find the best case. But comparing each group of modes for the different cases, usually the case with higher or similar PSNR to the PSNR of conventional HEVC framework is the case of classification modes when the decision boundary is linear and the combination of features used is three (Var, HAD and VAD).

4.2.4. Bit-rate

Bit-rate is the number of bits which are transmitted or processed per unit of time. In this case the data is encoded using a CAVLC encoder (Context-Adaptive Variable Length Coding) instead of a CABAC encoder (Context Adaptive Binary Arithmetic Coding) in order to obtain a bit stream. Then the bit rate is estimated using a 16x16 estimator from H.264/AVC [28].

In the following table are the results of the average bit rate for the original HEVC framework for comparison and for the proposed approach for the three different classification modes. These results are obtained for two different decision boundaries and for two and three combination of the features corresponding with table 20.

					Bit-rate (bits)			
					QP = 22	**QP = 27**	**QP = 32**	**QP = 37**
Video 1 (BQSquare)	**HEVC framework**				255145	165851	89654	32268
	Proposed approach framework	**Two features**	**Linear Decision boundary**	**Mode DC**	255134	165833	89643	32262
				Mode PH	256694	166821	90116	32441
				Mode PV	255179	165875	89637	32293
			Quadratic Decision boundary	**Mode DC**	255090	165800	89622	32248
				Mode PH	257071	167122	90369	32595
				Mode PV	255184	165889	89638	32279
		Three features	**Linear Decision boundary**	**Mode DC**	255100	165803	89640	32257
				Mode PH	256233	166581	90011	32405
				Mode PV	255154	165863	89625	32283
			Quadratic Decision boundary	**Mode DC**	255033	165756	89614	32224
				Mode PH	257427	167370	90612	32752
				Mode PV	255166	165882	89665	32260

					Bit-rate (bits)			
					QP = 22	**QP = 27**	**QP = 32**	**QP = 37**
Video 2 (BasketBall)	**HEVC framework**				61229	29261	10482	2912
	Proposed approach framework	**Two features**	**Linear Decision boundary**	**Mode DC**	61229	29254	10453	2911
				Mode PH	61342	29359	10615	2991
				Mode PV	61467	29398	10482	2916
			Quadratic Decision boundary	**Mode DC**	60806	29091	10520	2942
				Mode PH	62144	30003	10883	3031
				Mode PV	61553	29438	10532	2935
		Three features	**Linear Decision boundary**	**Mode DC**	61223	29259	10460	2908
				Mode PH	61241	29285	10562	2967
				Mode PV	61515	29422	10495	2912
			Quadratic Decision boundary	**Mode DC**	60831	29090	10519	2955
				Mode PH	63014	30557	11010	3058
				Mode PV	61463	29387	10560	2947

Table 33: Bit-rate.

The classification mode DC has similar bit rate as the HEVC framework because the binary classifier mostly of the times predicts class 0. Usually for both videos, the classification mode DC has the lowest bit rate followed by the classification mode Pure Vertical and then Pure Horizontal.

4.2.5. Bjontegaard Distortion-rate

Bjontegaard Distortion-rate (BD-rate) computes the area of the average gain in PSNR and the bit rate between two rate-distortion curves (RD). A negative BD-rate implies that the proposed approach brings coding gains, while a positive BD-rate implies that the proposed approach brings coding losses [34].

In the following table are the results of the BD-rate for each video.

Videos	Features	Decision Boundary	BD-rate Proposed approach framework		
			Mode DC	Mode PH	Mode PV
Video 1	Two	Linear	-0.0114 %	1.1578	0.0111
		Quadratic	-0.0491 %	1.2037	0.0067
	Three	Linear	-0.0238 %	0.7975	-0.0013
		Quadratic	-0.0638 %	1.9074	0.0102
Video 2	Two	Linear	-1.1797 %	0.9392	0.4592
		Quadratic	-0.3921 %	3.1292	0.8821
	Three	Linear	-0.1125 %	0.6041	0.4054
		Quadratic	-0.3091 %	5.0288	0.9190

Table 34: BD-rate.

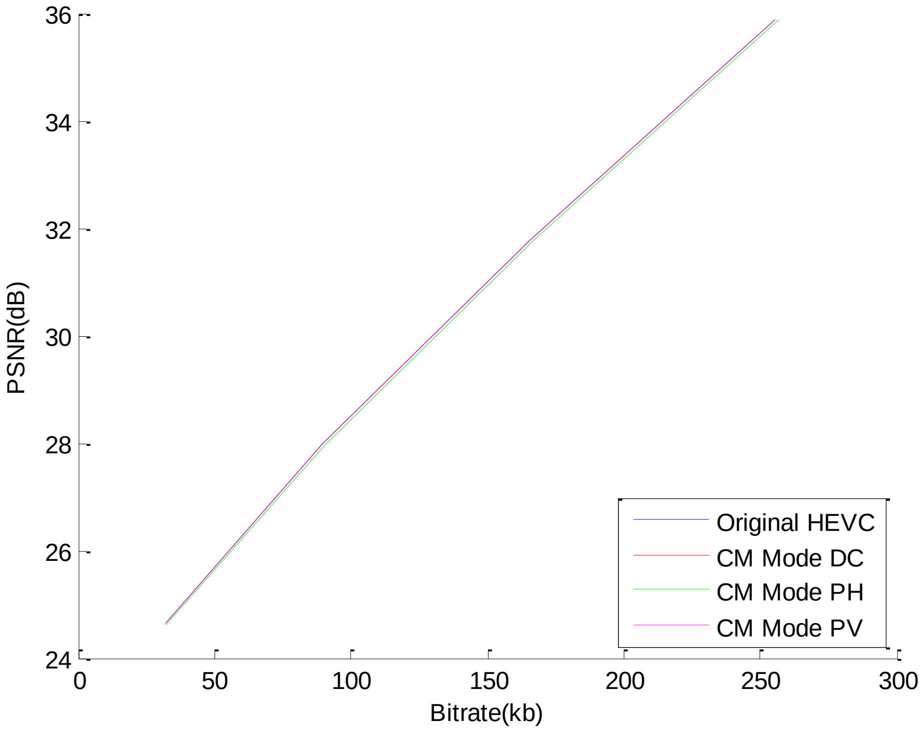
The rate-distortion curves between the original HEVC framework and the proposed approach for each video and each classification mode are shown in the following figures.

When the fitted curves are close to straight lines it means that the BD-rates have good accuracy. In figure 64 and 65 it is possible to see that there is no space between the different rate-distortion curves.

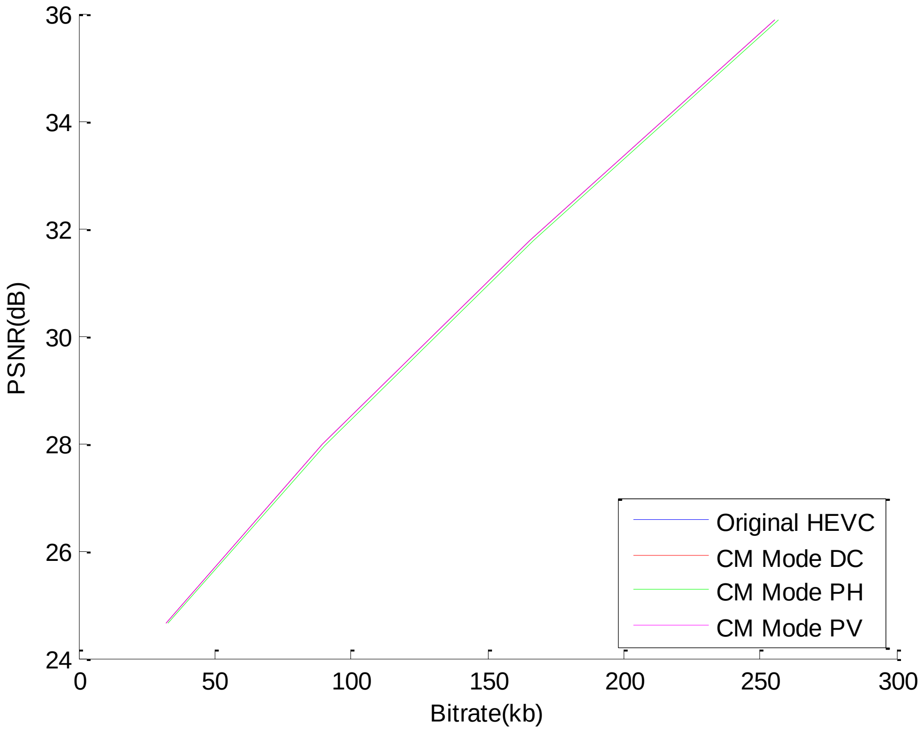
The BD-rate was computed using the previous PSNR and bit-rate calculated in previous subsections. Comparing each case of table 33, it is possible to see that the best case is when the decision boundary is linear and the combination of features used is three (Var, HAD and VAD), corroborating the previous premise.

- **VIDEO 1**

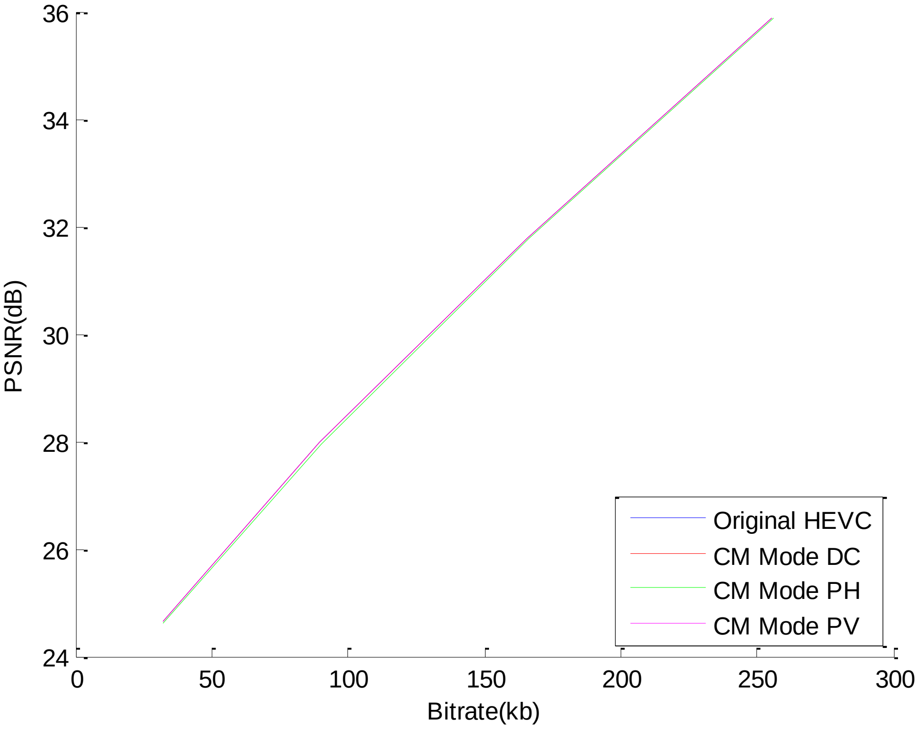
TWO FEATURES AND LINEAR DECISION BOUNDARY



TWO FEATURES AND QUADRATIC DECISION BOUNDARY



THREE FEATURES AND LINEAR DECISION BOUNDARY



THREE FEATURES AND QUADRATIC DECISION BOUNDARY

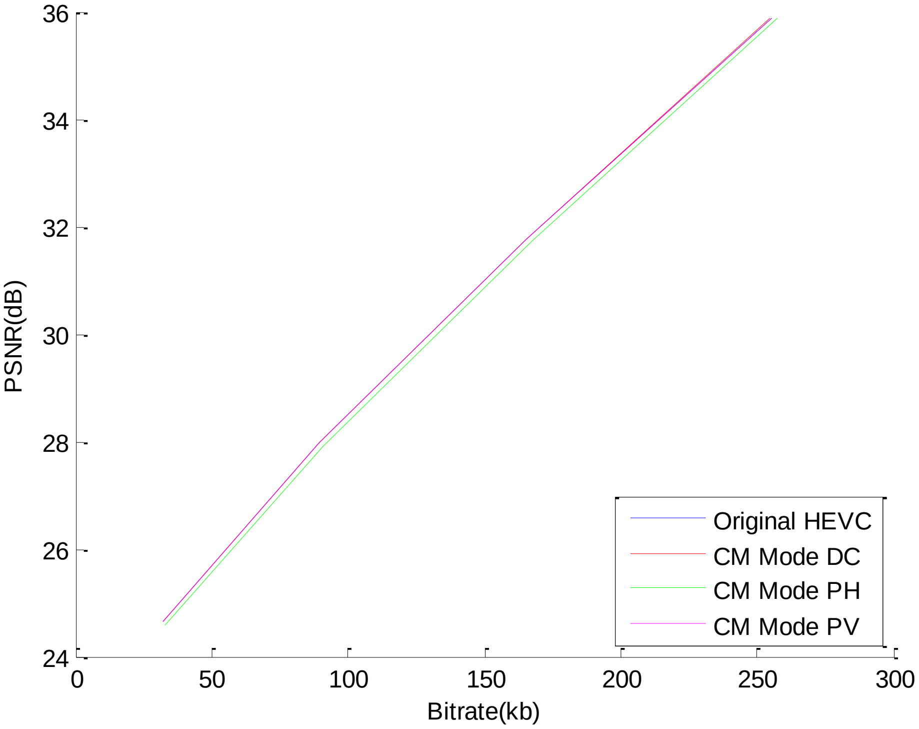
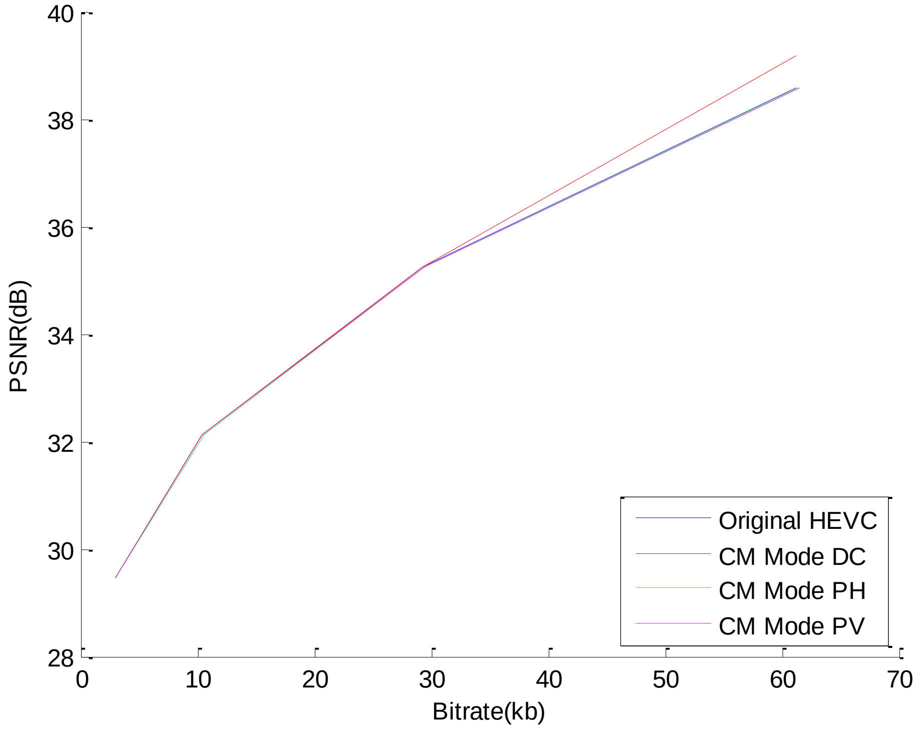


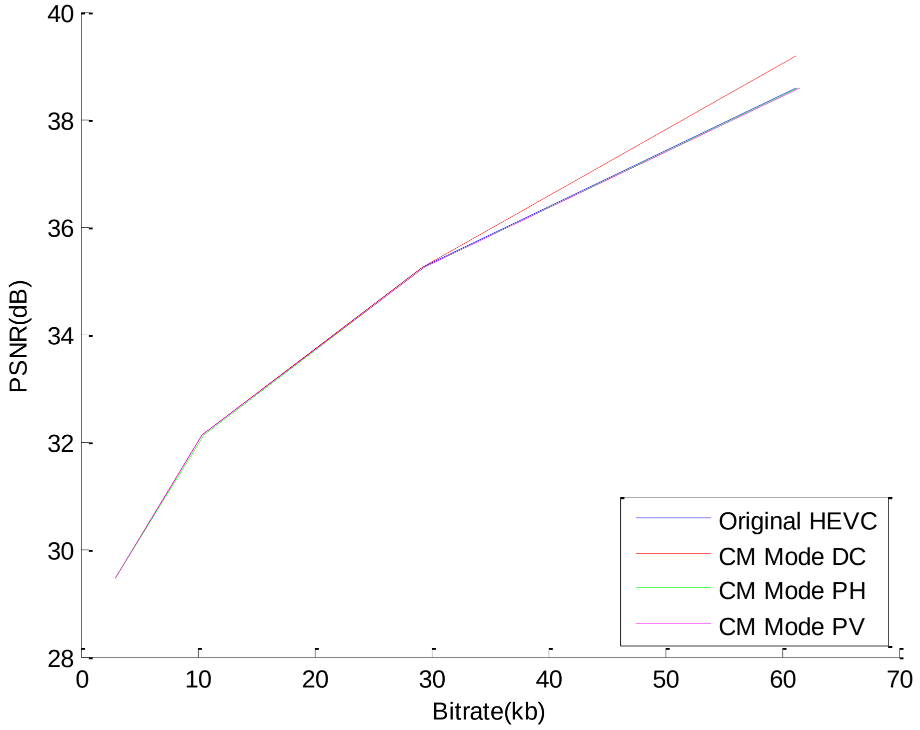
Figure 64: Rate-distortion curves for video 1.

- **VIDEO 2**

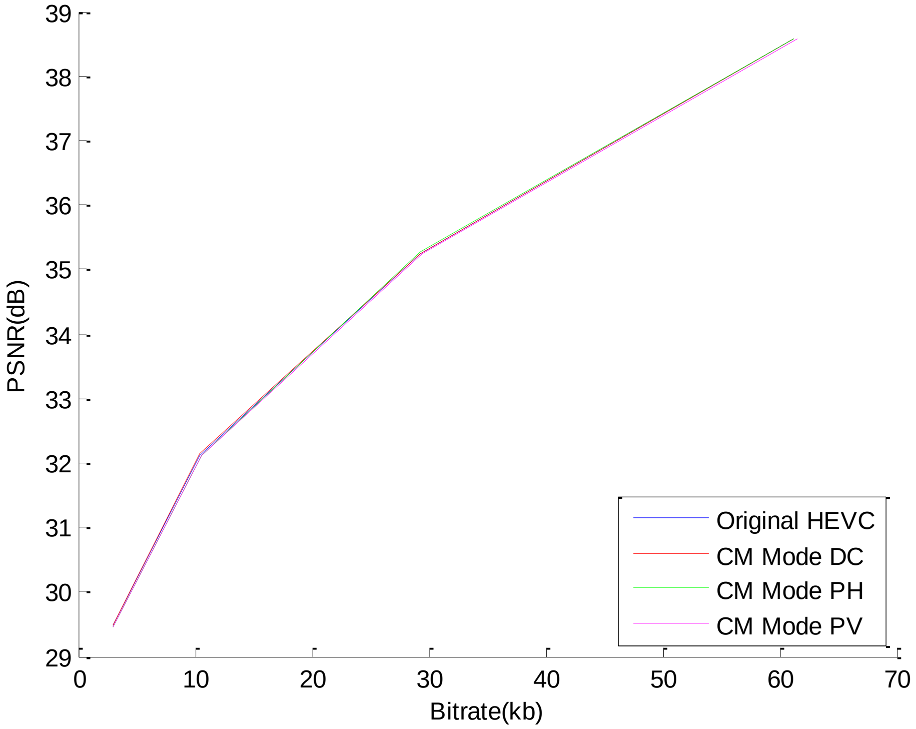
TWO FEATURES AND LINEAR DECISION BOUNDARY



TWO FEATURES AND QUADRATIC DECISION BOUNDARY



THREE FEATURES AND LINEAR DECISION BOUNDARY



THREE FEATURES AND QUADRATIC DECISION BOUNDARY

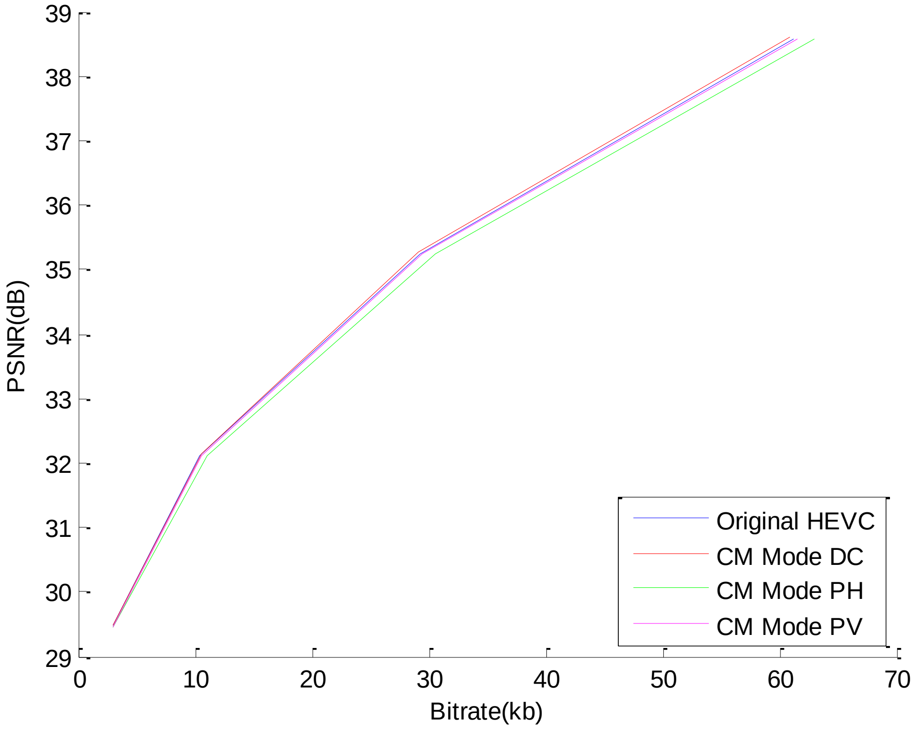


Figure 65: Rate-distortion curves for video 2.

5.

CONCLUSIONS AND FUTURE WORK

5.1. Conclusions

High Efficient Video Coding standard (HEVC) has emerged as a technology covering the increasingly need for higher coding efficiency and compression of video content. This comes at the cost of a generally much higher computational complexity with respect to previous standards. So, the main goal of this project was to reduce this computational complexity and at the same time trying to maintain its compression efficiency, by improving the Intra-Picture prediction block.

The computational complexity in case of intra-prediction mode is due to the larger number of block size and prediction modes. Therefore in this thesis a novel framework was defined and implemented to avoid an exhaustive search among all possible intra-modes. Consequently different experiments were performed in order to find the case that gives the best values.

The results of the objective video quality evaluation prove that the implemented framework reduces the computational complexity, mainly using classification mode Pure Horizontal for the two videos tested.

The BD-rate, based on the PSNR and bit rate calculated, corroborates the thesis that the best results are obtained using a linear decision boundary and a combination of three features (Var, HAD and VAD) for all the classification modes. The values obtained for this case and mainly for classification mode Pure Horizontal and Pure Vertical are lower than the other cases indicating that it produced relatively small coding losses, obtaining coding efficiency similar to the conventional HEVC framework.

Due to the reduction of the computational complexity, the number of operations of the SAD computed is lower for the proposed approach than for conventional HEVC. The results shows that the proposed approach, for a quadratic decision boundary and a combination of three features, is capable of encoding the two videos using in average 16.5% less number of operations than conventional HEVC, at the cost of relatively small compression efficiency losses calculated by the BD-rate.

5.2. Future work

Some extensions can be implemented based on the work done in this thesis, for instance:

- An implementation of a similar approach on other prediction modules in HEVC, for instance in the case of inter-prediction. An evaluation of the objective video quality should be also performed in order to obtain the best results for the computational complexity reduction.
- Different features and/or combination of features may be considered which could possibly bring more accurate discrimination between the classes.
- Finally, a multi-class classifier (classify the data into more than two classes) may be implemented to select the optimal mode for each block, instead of selecting between Single mode and Four modes.

BIBLIOGRAPHY

[1]. Gary J. Sullivan and Thomas Wiegand. ***Rate-Distortion Optimization for Video Compression.*** IEEE Signal Processing Magazine. November 1998.

[2]. Jens-Rainer Ohm, *Member, IEEE*; Gary J. Sullivan, *Fellow, IEEE*; Heiko Schwarz; Thiow Keng Tan, *Senior Member, IEEE,* and Thomas Wiegand, *Fellow, IEEE*. ***Comparasion of the Coding Efficiency of Video Coding Standards-Including High Efficiency Video Coding (HEVC)***. IEEE Transactions on Circuits and System for Video Technology. December 2012.

[3]. Gary J. Sullivan, *Fellow, IEEE*, Jens-Rainer Ohm, *Member, IEEE*, Woo-Jin Han, *Member, IEEE,* and Thomas Wiegand, *Fellow, IEEE*. ***Overview of the High Efficiency Video Coding (HEVC) Standard.*** IEEE Transactions on Circuits and System for Video Technology. December 2012.

[4]. Philippe Bordes, Gordon Clare, Félix Henry, Mickael Raulet, Jérome Viéron**.** ***An overview of the emerging HEVC standard.*** Technicolor, Orange Labs, IETR/INSA Rennes, Ateme, France.

[5]. Iain Richardson. ***HEVC: An introduction to High Efficiency Video Coding.*** Vcodex Video compression. 2013

[6]. Iain E.G. Richardson. ***H.264 and MPEG-4 Video Compression Video Coding for Next-generation Multimedia.*** 2003. ISBN 978-0-470-84837-1.

[7]. Jorn Ostermann, Jan Bormans, Peter List, Detlev Marpe, Matthias Narroschke, Fernando Pereira, Thomas Stockhammer, and Thomas Wedi. ***Video coding with H.264/AVC: Tools, Performance, and Complecity*. IEEE Circuits and Systems Magazine.

[8]. John R. Smith, IBM, Hari Kalva. ***The H.264 Video Coding Standard*.** Published by the IEEE Computer Society

[9]. Vivienne Sze, *Member, IEEE*, and Madhukar Budagavi, *Senior Member, IEEE*. ***High Throughput CABAC Entropy Coding in HEVC.*** IEEE Transactions on Circuits and System for Video Technology. December 2012.

[10]. Joel Sole, Rajan Joshi, Nguyen Nguyen, Tianying Ji, Marta Karczewicz, Gordon Clare, Félix Henry, and Alberto Dueñas. ***Transform Coefficient Coding in HEVC.*** IEEE Transactions on Circuits and System for Video Technology. December 2012.

[11]. Ian E. Richardson. ***The H.264 Advanced Video Compression Standard.*** Second Edition. 2010. ISBN 978-0-470-51692-8.

[12]. Jani Lainema, Frank Bossen, *Member*, *IEEE*, Woo-Jin Han, *Member*, *IEEE*, Junghye Min, and Kemal Ugur. ***Intra Coding of the HEVC Standar.*** IEEE Transactions on Circuits and System for Video Technology. December 2012.

[13]. Jani Lainema, Kemal Ugur. ***Angular Intra Prediction in High Efficiency Video Coding (HEVC).*** Nokia Research Center, Visiokatu 1, Tampere Finland.

[14]. Kiran Misra, *Member, IEEE,* Andrew Segall*, Member, IEEE,* Michael Horowitz, Shilin Xu, Arild Fuldseth, and Minhua Zhou. ***An overview of tiles.*** IEEE Journal of Selected Topics in Signal Processing, Vol 7, December 2013

[15]. Benjamin Bross, *Fraunhofer HHI*, Woo-Jin Han, *Gachon University*, Jens-Rainer Ohm, *RWTH Aachen*, Gary J. Sullivan, *Microsoft*, Ye-Kui Wang, *Qualcomm*, Thomas Wiegand, *Fraunhofer HHI / TU Berlin*. ***High efficiency video coding (HEVC). Recommendation ITU-T H.265***. Document: JCTVC-L1003_v34. January 2013.

[16]. Philipp Helle, Simon Oudin, Benjamin Bross, *Student Member, IEEE,* Detlev Marpe, Senior Member, IEEE, M. Oguz Bici, Kemal Ugur, Joel Jung, Gordon Clare, and Thomas Wiegand, *Fellow, IEEE*. ***Block Merging for Quadtree-Based Partitioning in HEVC.*** IEEE Transactions on Circuits and System for Video Technology. December 2012.

[17]. Detlev Marpe, *Member, IEEE*, Heiko Schwarz, and Thomas Wiegand. ***Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard.*** IEEE Transactions on Circuits and System for Video Technology, Vol 13, No 7 July 2003.

[18]. ***High Efficiency Video Coding (HEVC).*** Директор по развитию TELCO Group Г.Авдеев

[19]. Hsueh-Ming Hang, Wen-Hsiao Peng, Chia-Hsin Chan and Chun-Chi Chen. ***Towards the Next Video Standard: High Efficiency Video Coding***. Proceedings of the Second APSIPA Annual Summit and Conference, pages 609–618, Biopolis, Singapore, 14-17 December 2010.

[20]. Gary J. Sullivan, *Senior Member, IEEE* and Thomas Wiegand. ***Video Compression—From Concepts to the H.264/AVC Standard.*** Proceedings of the IEEE, Vol 93, No 1, January 2005.

[21]. Goff Hill. ***The Cable and Telecommunications Professionals' Reference: Transport Networks***. Vol 2, Third Edition 2008, ISBN 978-0-240-80748-5.

[22]. S Jayaraman,S Esakkirajan, T Veerakumar. ***Digital Image processing***. First Edition. 2009. ISBN 978-0-07-014479-8.

[23]. ***Lossless Image Compression Systems***. Module 2. Version 2 ECE IIT, Kharagpur.

[24]. Imran Ullah Khan and M.A. Ansari. ***Performance Analysis of H.264 Video Coding Standard and H.263 Video Coding Standard.*** VSRD Technical & Non-Technical Journal, Vol2 (1), 2011, 8-14.

[25]. Sandeep Kanumuri, TK Tan, Frank Bossen. ***Enhancements to Intra Coding Input***. Document to JCT-VC. D235. January 2011.

[26]. ***H.264/ MPEG-4 Part 10: Intra Prediction.*** *Prediction of Intra Macroblocks*. April 2003. Website www.vcodex.com.

[27]. ***Recommendation ITU-T H.262***. Generic Coding of moving pictures ans associated audio information: Video. International Telecommunication Union. November 1996.

[28]. Saverio G. Blasi, Marta Mrak and Ebroul Izquierdo. ***Masking of transformed intra-predicted blocks for high quality image and video coding***. IEEE International Conference on Image Processing, Paris France October 2014

[29]. Images obtained from: http://r0k.us/graphics/kodak/

[30]. G. Bjontegaard. ***Improvements of the BD-PSNR model.*** ITU-T SG16/Q6, 35th VCEG Meeting, VCEG-AI11 2008

[31]. Milos Hauskrecht. Binary classification CS 1571 Introduction to AI Lecture 25. http://people.cs.pitt.edu/~milos/courses/cs1571-Fall2012/Lectures/Class25.pdf

[32]. ***Linear Binary Classification***. https://inst.eecs.berkeley.edu/~ee127a/book/login/l_lqp_apps_class.html

[33]. Mehryar Mohri. ***Introduction to Machine Learning Lecture 8***. Courant Institute and Google Research. http://www.cs.nyu.edu/~mohri/mlu/mlu_lecture_8.pdf

[34]. G. Bjontegaard. ***Improvements of the BD-PSNR model***. ITU-T SG16/Q6, 35th VCEG Meeting, Doc.VCEG-AI11, 2008.

ANEXOS

A.

INTRODUCCIÓN

Motivación

La codificación de video digital se utiliza para comprimir la información para que pueda ser almacenada o transmitida ocupando un espacio mínimo. Este proceso se realiza en un codificador de vídeo y el objetivo principal es optimizar la eficiencia de codificación. La eficiencia de codificación es la capacidad de minimizar la tasa de bits necesaria para representar un vídeo manteniendo al mismo tiempo un nivel de calidad dado.

Los dos estándares de codificación de vídeo anteriores, el H.262/MPEG-2 Video [27] y el H.264/MPEG-4 AVC (Advanced Video Coding) [11], han tenido una influencia significativa en una gran variedad de productos que se encuentran con frecuencia presentes en nuestra vida cotidiana. Sin embargo, la creciente popularidad del vídeo de alta definición, la aparición de nuevos formatos (4k, 8k), la necesidad de una mayor resolución en 3D o multivisión, y el aumento del interés por una mayor calidad y resolución en las aplicaciones móviles, están generando necesidades urgentes de una eficiencia de codificación superior a las capacidades de H.264/MPEG-4 AVC. Además, más del 50% del tráfico actual de las redes es vídeo dirigido a dispositivos móviles y Tablet-PC; este crecimiento del tráfico y las necesidades de transmisión de los servicios de video-on-demand están imponiendo severos desafíos en las redes actuales.

Por estas razones y debido principalmente a la creciente necesidad de una mayor compresión del contenido de los vídeos, el estándar H.265/HEVC (High Efficiency Video Coding) [15] ha sido desarrollado. Ésta Recomendación aborda todas las aplicaciones existentes del estándar H.264/MPEG-4 AVC y también muchas nuevas aplicaciones, como las anteriores mencionadas. Se ha diseñado para prestar especial atención a tres cuestiones clave: el aumento de la resolución de vídeo, la facilidad de integración del sistema de transporte y el aumento del uso de arquitecturas de procesamiento paralelo.

Gracias a las nuevas mejoras y a un esquema de codificación optimizado, el estándar HEVC es capaz de mejorar la eficiencia de compresión reduciendo en un promedio del 50% la tasa de bits para el mismo objetivo y la percepción de la calidad de los vídeos con respecto a su predecesor, H.264 / MPEG-4 AVC [7]. Esto ocurre a costa de una complejidad computacional mucho mayor con respecto a los estándares anteriores.

Objetivos

El objetivo principal de este proyecto es reducir la complejidad computacional y al mismo tiempo lograr una eficiencia de compresión comparable al HEVC convencional. Para lograr esto, el bloque de predicción Intra-Picture ha sido mejorado.

Por lo tanto, el primer paso es analizar el bloque de predicción Intra-Picture del HEVC convencional, que consiste en predecir un bloque en el frame actual utilizando la información de referencia de los bloques vecinos en el mismo frame. Soporta tres modos diferentes, el modo Angular con 33 direcciones diferentes, el modo Planar y el modo DC.

El segundo paso es el diseño de un framework para probar y desarrollar los métodos de predicción Intra del estándar HEVC replicando el procedimiento tal como se encuentra en el software de referencia de HEVC.

Y, por último, el tercer paso es utilizar este framework para mejorar los métodos de predicción Intra del estándar HEVC utilizando clasificación binaria de unas características extraídas de los frames. Una comparación entre ambos frameworks se realizará en términos de la tasa BD y el tiempo de ejecución. Para probar estos diseños, se han utilizado 24 imágenes y varios videos.

Estructura del documento

Este proyecto está estructurado de la siguiente manera:

1. **Introducción:** En este primer capítulo se explica la motivación, específicamente por qué este nuevo estándar ha sido desarrollado, y también se explican los objetivos y el enfoque de este proyecto.

2. **Estado del arte:** En esta parte se explica los fundamentos de la codificación de vídeo para entender la segunda parte que explica el estado del arte, que ahora es el estándar High Efficient Video Coding. Y la tercera parte de este capítulo consiste en explicar en detalle el método de predicción Intra.

3. **Diseño y desarrollo:** Esta parte presenta la formulación del problema y el diseño del HEVC framework así como la mejora propuesta para el bloque de predicción Intra.

4. **Experimentos y resultados:** Esta sección muestra los resultados para todos los experimentos llevados a cabo durante el proyecto, que consisten en evaluar las diferentes combinaciones de las características para 24 imágenes y evaluar la calidad del framework del enfoque propuesto también para 3 videos.

5. **Conclusiones y trabajo futuro:** Las conclusiones y el trabajo futuro son presentadas en detalle.

B.

CONCLUSIONES Y TRABAJO FUTURO

Conclusiones

El estándar High Efficient Video Coding (HEVC) ha surgido como una tecnología que cubre la creciente necesidad de una mayor eficiencia de codificación y compresión de vídeo. Esto ocurre generalmente a costa de una mayor complejidad computacional con respecto a los estándares anteriores. Así que, el objetivo principal de este proyecto era reducir ésta complejidad computacional y al mismo tiempo intentar mantener su eficiencia de compresión, mediante una mejora del bloque de predicción Intra-Picture.

La complejidad computacional en el caso del bloque de predicción Intra-Picture es debido al aumento del número de tamaños del bloque y de modos de predicción. Por lo tanto, en este proyecto se ha definido e implementado un novedoso framework para evitar una búsqueda exhaustiva entre todos los posibles modos Intra. Por consiguiente diferentes experimentos se realizaron con el fin de encontrar el caso que de los mejores resultados.

Los resultados de la evaluación objetiva de la calidad de los vídeos prueban que el framework implementado reduce la complejidad computacional, utilizando principalmente el modo de clasificación Pure Horizontal para los dos vídeos.

El BD-rate, basado en el PSNR y el bit-rate calculado con anterioridad, corrobora la tesis de que los mejores resultados se obtienen utilizando una frontera de decisión lineal y una combinación de tres características (Var, HAD y VAD) para todos los modos de clasificación. Los valores obtenidos para este caso, y principalmente para los modos de clasificación Pure Horizontal y Pure Vertical son más bajos que los otros casos indicando que se producen pérdidas de codificación relativamente pequeñas, obteniendo una eficiencia de codificación similar a la del HEVC convencional.

Debido a la reducción de la complejidad computacional, el número de operaciones del SAD calculado es menor para el enfoque propuesto que para el HEVC convencional. Los resultados muestran que el enfoque propuesto, para una frontera de decisión cuadrática y una combinación de tres características, es capaz de codificar los dos vídeos utilizando en promedio un 16,5% menos de operaciones que el HEVC convencional, a costa de pérdidas relativamente pequeñas de la eficiencia de compresión calculadas por el BD-rate.

Trabajo futuro

Algunas extensiones se pueden implementar basándose en el trabajo realizado en este proyecto, por ejemplo:

- Una implementación con un enfoque similar sobre otros módulos de predicción en HEVC, por ejemplo en el caso de Inter-predicción. Una evaluación objetiva de la calidad de los vídeos debería ser también realizada con el fin de obtener los mejores resultados para dicha reducción de la complejidad computacional.
- Se pueden considerar diferentes características de las utilizadas y/o diferentes combinaciones de las características lo que podría llevar a una discriminación más precisa entre las clases.
- Por último, un clasificador multi-clases (clasifica los datos en más de dos clases) puede ser implementado para seleccionar el modo óptimo para cada bloque, en lugar de seleccionar entre el Modo individual y Cuatro modos.

C.

PRESUPUESTO

1) Ejecución Material

Compra de ordenador personal (Software incluido)	2.200 €
Alquiler de impresora láser durante 6 meses	50 €
Material de oficina	200 €
Total de ejecución material	2.450 €

2) Gastos generales

16 % sobre Ejecución Material	392 €

3) Beneficio Industrial

6 % sobre Ejecución Material	147 €

4) Honorarios Proyecto

1200 horas a 15 € / hora	18000 €

5) Material fungible

Gastos de impresión	70 €
Encuadernación	200 €

6) Subtotal del presupuesto

Subtotal Presupuesto	20720 €

7) I.V.A. aplicable

21% Subtotal Presupuesto	4351.2 €

8) Total presupuesto

Total Presupuesto	25071.2 €

Londres, Julio de 2014

El Ingeniero Jefe de Proyecto

Fdo.: Marta Merlo Serrano

Ingeniero Superior de Telecomunicación

D.

PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de *Improved Intra-prediction for video coding*. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.