ESCUELA DE INGENIERÍA DE FUENLABRADA

GRADO EN INGENIERÍA EN SISTEMAS AUDIOVISUALES Y MULTIMEDIA

TRABAJO FIN DE GRADO

ESTUDIO DEL NUEVO ESTÁNDAR DE COMPRESIÓN DE VÍDEO H.266 Y COMPARACIÓN CON SU PREDECESOR

Autor: Yolanda Lillo Mata
Tutor: David Gualda Gómez

Curso académico 2023/2024

“Nadie te va a recordar por tu currículum, sino por tu forma de ser”

Víctor Kuppers

AGRADECIMIENTOS

Parecía que no iba a llegar nunca, pero ahora si puedo decir que soy INGENIERA, y en este camino que no ha sido siempre de rosas como no agradecer a quien me ha soportado estos largos años.

En primer lugar, agradecer a mis padres suena típico, pero sin ellos esto no podría haber sido posible, gracias por vuestro esfuerzo, comprensión y paciencia.

A mi hermana, por guiarme en este largo camino de la ingeniería y estar siempre cerca. Como no, a la tía Tere, por aguantar esta andadura, por todos esos años que me dio hogar y me enseñó a salir de casa siempre con el pie derecho.

Continuamos por mi tutor David, a quien quisiera dar las gracias por guiarme en este proyecto, animarme cuando parecía que no iba a acabar nunca, por su infinita paciencia y por todos los conocimientos que me has ayudado a conseguir.

Gracias a todos los compañeros que he conocido en esta andadura y que han formado parte de ella, por hacer todo este proceso más ameno. Sobre todo, gracias, Elena y María por ser el mejor equipo, porque juntas no había asignatura que se nos resistiese, bueno alguna sí. Gracias por sostenerme cuando la cosa no iba bien, por los ánimos, aunque a veces todas estuviésemos igual de cansadas, por las celebraciones, por seguir ahí tras finalizar esta etapa.

A esos amigos/as de siempre los que me apoyaron en algún momento de este recorrido por no dejarme abandonar esta ingeniería.

A mi fiel amigo Carlos, por alegrarte de cada aprobado como si fuese tuyo, por las charlas eternas pasando el día por Madrid que no arreglaban el mundo, pero si nos daban vida y fuerzas para seguir, pero sobre todo por tu apoyo de principio a fin.

Para finalizar, nunca fui partidaria de añadir esto en unos agradecimientos de TFG, pero aquí estoy, el amor llegó a mi vida y cambio un poco todo.

Gracias a ti Iván, mi amor, mi compañero de vida por llegar al final de esta andadura para darme las fuerzas necesarias para terminar, por aguantar mi mal humor y darme calma en la desesperación estos casi tres años. Por tu apoyo y paciencia, ojalá seguir cumpliendo objetivos a tu lado.

RESUMEN

En este Trabajo Fin de Grado (TFG) se ha llevado a cabo el estudio del último estándar de compresión de video H.266, desarrollado en conjunto por la Unión Internacional de Telecomunicaciones y la Organización Internacional de Estandarización.

En dicho estudio, se ha profundizado en la arquitectura del estándar comparándolo cada parte con su antecesor, el estándar H.265.

Por último, se han obtenido resultados comparativos del estándar H.266 con respecto a H.265 y H.264 a partir de videos con distintas resoluciones y valores de *bitrate*, utilizando una profundidad de color de 10 bits, mostrando los resultados con las métricas objetivas más habituales (PSNR, VMAF y SSIM).

PALABRAS CLAVE

Estándar, códec, *bitrate*, PSNR (*Peak Signal-to-Noise Ratio*), compresión, video, resolución, VMAF (*Video Multimethod Assessment Fusion*), SSIM (*Structural Similarity Metric*).

ABSTRACT

In this Final Degree Project (TFG), a study of the latest video compression standard H.266, developed jointly by the International Telecommunication Union and the International Organization for Standardization has been carried out.

In this study, the architecture of the standard has been studied in depth, comparing each part with its predecessor, the H.265 standard.

Finally, comparative results of the H.266 standard with respect to H.265 and H.264 have been obtained from videos with different resolutions and bitrate values, using a color depth of 10 bits, showing the results with the most common objective metrics (PSNR, VMAF and SSIM).

KEYWORDS

Standard, codec, bitrate, PSNR (*Peak Signal-to-Noise Ratio*), compression, video, resolution, VMAF (*Video Multimethod Assessment Fusion*), SSIM (*Video Multimethod Assessment Fusion*).

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

INDICE DE TABLAS

ACRÓNIMOS Y SIGLAS

ALF	*Alfa Laval Filter*
AVI	*Audio Video Interleave*
CABAC	*Context Adaptative Binary Arithmetic Coder*
CAVLC	*Context-Adaptive Variable Length Coding*
CD	*Compact Disc*
CIF	*Common Interchange Format*
CTU	*Coding Tree Units*
DBF	*Deblocking Filter*
DCT	*Discrete Cosine Transform*
FIFO	*First In, First Out*
FPS	*Frames per Second*
GOP	*Group of pictures*
HD	*High Definition*
ISO	*International Standards Organization*
ITU	*International Telecommunications Union- Telecommunications*
LFNST	*Low Frequency Non Separable Transforms*
MPEG	*Moving Pictures Experts Group*
MVD	*Motion Vectors Difference*
PSNR	*Peak Signal-to-Noise Ratio*
QCIF	*Quarter Common Intermediate Format*
RLE	*Run Length Encoding*
SAO	*Sample Adaptive Offset*
SSIM	*Structural Similarity Metric*
SIMD	*Single Instruction Multiple Data*

UHD	*Ultra High-Definition*
VCEG	*Video Coding Experts Group*
VHS	*Video Home System*
VMAF	*Video Multimethod Assessment Fusion*
VVC	*Versatile Video Coding*
VVenC	*Fraunhofer Versatile Video Encoder*
WMV	*Windows Media Video*
YUV	*Intensity, Hue and Value*

1. INTRODUCCIÓN

1.1. PRESENTACIÓN

En este Trabajo de Fin de Grado se lleva a cabo un proyecto que consiste en estudiar el último estándar de video (H.266) y compararlo con su antecesor.

Lo que se encuentra a modo introductorio es una evolución histórica de los códecs para finalmente centrarnos en H.266, sus características, su arquitectura y una herramienta para poder realizar compresión con este estándar.

Tras ello, se comparan los resultados con su antecesor, es decir, también se analizan videos en H.265 con una herramienta similar para poder realizar una comparativa de ambos resultados.

1.2. ESTRUCTURA DE LA MEMORIA

En este punto se analizarán cada uno de los capítulos que contiene este Trabajo Fin de Grado para facilitar el entendimiento y orden a los lectores:

- **Introducción**, donde se muestra una presentación rápida del proyecto, así como la estructura de esta memoria.
- **Objetivos**, donde se explica lo que se quiere conseguir con este trabajo, así como las fases en la que se ha estructurado y el tiempo aproximado de cada una de las fases.
- **Definiciones y conceptos**, se describen algunos términos que se consideran importante tener claro para la compresión del trabajo.
- **Desarrollo del trabajo**, aquí se empieza con la evolución histórica de todos los estándares desde los años 90, se continua con las características del códec h.266 que es en el que se centra en este TFG, así como su arquitectura. Por último, dentro de este punto se verá la comparativa entre los dos últimos códecs conocidos, donde veremos cada herramienta usada para comprimir y descomprimir los videos, así como sus comparaciones en tamaño, *bitrate*, PSNR (*Peak Signal-to-Noise Ratio*) …
- **Resultados,** se describen los resultados que se han obtenido tras ambas codificaciones de forma gráfica.
- **Conclusiones,** se recogen las conclusiones obtenidas durante la realización del proyecto, así como las competencias que se han adquirido y empleado y finalmente se proponen algunas líneas de mejora.
- **Bibliografía**, se encuentran las referencias a las fuentes consultadas para la realización de este proyecto.
-

2. OBJETIVOS

En este punto se pueden ver los objetivos con los que se realiza este trabajo, así como cada una de las fases que se ha llevado a cabo para su elaboración. Por último, se observa un diagrama de Gantt.

2.1. OBJETIVOS DEL TRABAJO

Entre los objetivos de este trabajo está:

- Conocer la historia y evolución de los códecs de video a lo largo de la historia.
- Profundizar en el conocimiento del estándar H.266, en sus características y arquitectura.
- Aprender a usar herramientas de codificación y decodificación para H.265 y H.266.
- Comparar los dos últimos estándares de vídeo a partir de métricas objetivas.

2.2. FASES DEL TRABAJO

2.2.1. Fase de planificación

En esta etapa inicial de planificación, lo que se lleva a cabo es el análisis del proyecto que se desarrolla, se forma la idea principal del proyecto. Posteriormente se definen los objetivos generales que representan la meta a alcanzar y se comienza a dar forma al proyecto, se estructura la memoria a grandes rasgos. Una vez se tienen estos, se puede continuar con la siguiente fase.

2.2.2. Fase de investigación

El objetivo de esta fase es la obtención de información y conceptos relevantes sobre los códecs y sus herramientas para poder realizar el trabajo. Se investigó sobre todos los códecs y más en profundidad sobre el H.266 y tras ello se empezó a seleccionar las herramientas necesarias para las codificaciones. Una vez se comenzó con el desarrollo del proyecto. También fue necesario seguir profundizando en la investigación de las herramientas que se usan, para conseguir información, solucionar dudas y problemas.

2.2.3. Fase de documentación

Esta fase se ha realizado en numerosas ocasiones a lo largo del proyecto. Al principio para redactar los objetivos y adentrarse en la estructura de la memoria. Se continúo documentando toda la información relativa a los códecs de vídeo. Además, se tuvo que documentar todo lo relativo a las herramientas que se utilizan para codificar los videos, su características, instalación y funcionamiento. Y, por último, en la parte final del proyecto se comparan los resultados obtenidos para poder obtener las conclusiones, así como los apartados finales como resumen, anexos…

2.2.4. Fase de desarrollo

Esta fase se podría dividir en dos, ya que para realizar la comparativa se necesita usar dos herramientas de codificación de vídeo diferentes: una para H.265 y otra para H.266.

- Herramienta FFmpeg

  En este paso, se ha descargado e instalado FFmpeg en un ordenador y se han realizado numerosas pruebas para comprimir vídeos con diferente tasa de *bitrate*.
- Herramienta FFmpeg VVCEasy

  En este punto, se ha descargado e instalado una herramienta dentro de FFmpeg como es, VVCeasy. Una vez lo teníamos se comienzan con las pruebas para comprimir los mismos videos que para el estándar anterior, video en formato YUV (*Intensity*, *Hue and Value*), pero en este caso los codificaremos en H.266.

2.2.5. Fase de análisis de resultados

En esta fase, se calculan diferentes métricas para los diferentes vídeos, codificado tanto con H.265 como con H.266. Una vez se disponen de los resultados de dichas codificaciones con videos HD (*High Definition*), FULL HD y UHD (Ultra High-Definition) en nuestros dos estándares, se calculan las métricas necesarias para compararlos y se representan para que sea más fácil observar sus diferencias. Adicionalmente y aunque no sea el centro del trabajo, también se comparará con H.264 para obtener mejores conclusiones.

2.3. DIAGRAMA DE GANTT

El objetivo de este diagrama es exponer el tiempo de dedicación aproximado para cada una de las tareas que engloba el proyecto a lo largo del año. En el eje horizontal se pueden observar los doce meses y en el vertical cada una de las fases del proyecto.

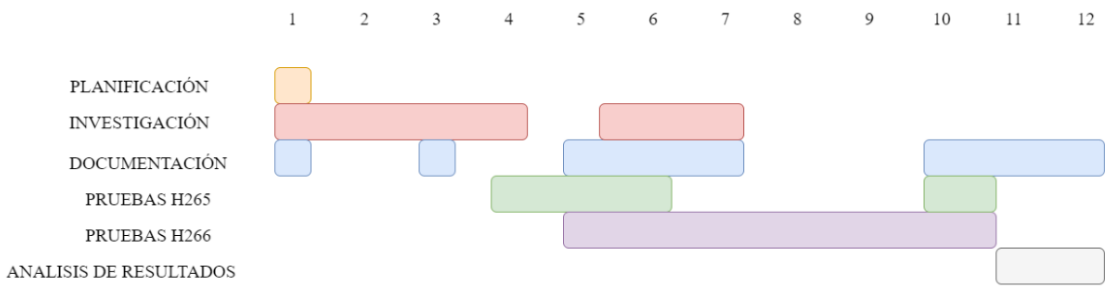


Figura 1. Diagrama de Gantt.

3. DEFINICIONES Y CONCEPTOS

Aquí se explican definiciones y conceptos usadas a lo largo de la memoria, que se considera importante de entender con claridad para facilitar la compresión del trabajo.

3.1. BITRATE

La tasa de bits, más conocido como *bitrate* [1] de un vídeo, es la tasa de tráfico o de datos, o en el mismo sentido, la cantidad de información que un ordenador reproduce por segundo. En consecuencia, a medida que aumenta la tasa de transferencia de datos, se mejora la calidad del contenido. Esta medida se expresa en kilobytes por segundo (kbps), de modo que cuanto mayor sea este valor, mayor será la calidad del vídeo resultante.

3.2. CODEC

Un códec se responsabiliza de cifrar y condensar los datos de un archivo de audio o vídeo para facilitar su transferencia y disminuir su tamaño. Cuando se reproduce o edita dicho archivo, se lleva a cabo la descompresión correspondiente.

El códec cumple dos funciones principales: comprimir y descomprimir. La compresión se ejecuta para hacer el archivo más manejable, mientras que la descompresión posibilita el acceso a todos los datos contenidos en el archivo.

3.3. RESOLUCIÓN

La definición de una pantalla se relaciona con la cantidad de píxeles que puede mostrar y se obtiene multiplicando el número de píxeles en anchura y altura [2]. Por lo general, a mayor cantidad de píxeles, mayor será la resolución. Entre las resoluciones podemos destacar:

- HD, que se refieren a una resolución de pantalla de 1280 x 720 píxeles.
- Full HD, es una resolución de alta definición de pantalla que consta con 1920 píxeles horizontales por 1080 píxeles verticales.
- Ultra HD o UHD, describe aquellas pantallas con una resolución de 3840 x 2160 píxeles. Estas pantallas a menudo se promocionan como “4K” debido a que tienen aproximadamente 4000 píxeles de ancho, aunque esta denominación puede no ser la más precisa.

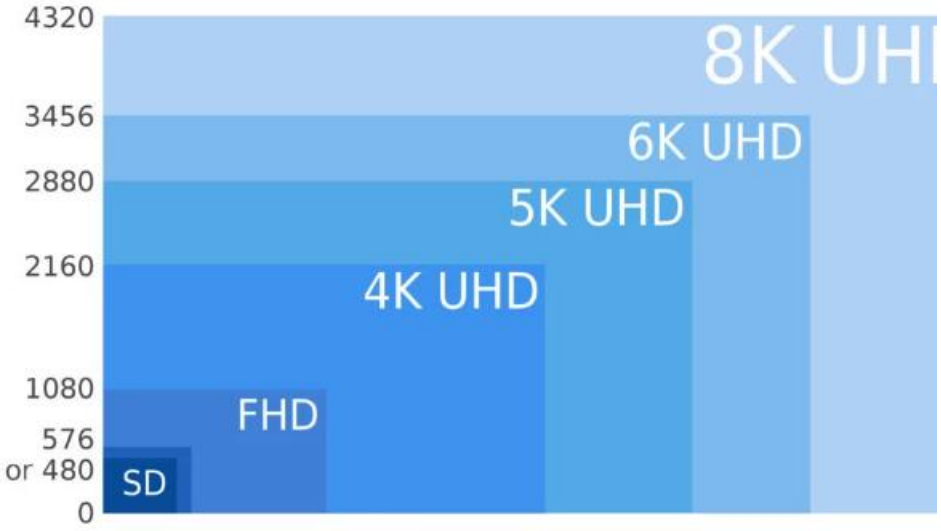


Figura 2. Tipos de resoluciones.

3.4. PSNR

En líneas generales, la relación señal a ruido de pico [3], es un concepto de ingeniería que describe la relación entre la máxima energía alcanzable de una señal y el nivel de ruido presente en su representación. Usualmente, se expresa en una escala logarítmica utilizando el decibelio como unidad de medida. Además, es la métrica más utilizada para la comparación de las prestaciones de los códecs de vídeo y un modo de indicar el error cuadrático medio en dB, indicando una mayor calidad cuanto mayor es su valor.

$$\text{PSNR (dB)} = 10\ log_{10}\left(\frac{[2^b-1]^2}{MSE}\right)$$

Figura 3. Fórmula PSNR.

3.5. RATIO DE COMPRESIÓN

Este concepto se refiere al proceso de transformar la frecuencia de datos de una señal de vídeo digital sin comprimir a una versión comprimida. Se puede entender como la relación entre el tamaño del archivo del vídeo original, sin comprimir, y el tamaño del archivo resultante después de la compresión.

Cuando mayor sea la relación de compresión [4] se utilizará menos ancho de banda para mantener un cierto número de imágenes por segundo. O bien, si el ancho de banda se mantiene constante, se podrá aumentar el número de imágenes por segundo.

Sin embargo, es importante tener en cuenta que un mayor nivel de compresión conlleva a una menor calidad de imagen para cada imagen individual.

3.6. FRAMERATE O TASA DE IMÁGENES POR SEGUNDO

Su definición más sencilla es cómo de rápido o de lento se muestran esas imágenes, es decir, la velocidad a la cual un dispositivo puede mostrar los cuadros o fotogramas por segundo. Generalmente se mide en fps (*Frames per Second*). Se puede apreciar una mayor fluidez y suavidad en el movimiento cuando se aumenta la tasa de cuadros por segundo.

Por ejemplo, si un video se tiene 10 fps, significa que en un segundo se mostrarán 10 imágenes, es decir, 10 *frames* en video. Cuantas más imágenes se vean, más fácil es engañar al cerebro para que vea un vídeo más fluido.

Si queremos saber hasta cuantos fotogramas por segundo es capaz de percibir el ser humano no podemos saberlo ya que la percepción del movimiento varía dependiendo del individuo.

Con respecto a los vídeos, si tenemos uno con una tasa de *frames* muy alta, hay riesgo de que la imagen se vea plana y sin vida y, por tanto, pierda el efecto natural. Los *framerate* más usados son:

- 24 fps: es el estándar para la industria televisiva y el cine.
- 30 fps: se usa más para eventos deportivos en televisión.
- 60 fps: para crear vídeos en cámara lenta, pues al poner el efecto en los editores de vídeo, al tener *frames* extra se seguirá viendo fluido, aunque lento.

3.7. VMAF (*Video Multimethod Assessment Fusion*)

Se trata de una medida objetiva [5] de calidad de video de referencia completa desarrollada por Netflix en colaboración con la Universidad del Sur de California, el laboratorio IPI/LS2N de la Universidad de Nantes y el Laboratorio de Ingeniería de Imagen y Video de la Universidad de Texas en Austin.

Su función principal es predecir la calidad subjetiva del vídeo utilizando tanto una referencia de calidad como una secuencia de vídeo distorsionada. Esta medida es útil para evaluar la calidad de varios códecs de vídeo, codificadores, configuraciones de codificación o variantes de transmisión. Además, tiene en cuenta artefactos de escalado y compresión, y cuenta con un modelo específicamente entrenado para el consumo de video móvil [6].

VMAF combina diversas métricas espaciales y temporales mediante modelos de aprendizaje automático. Ha demostrado estar más alineado con la percepción visual humana que métricas como el PSNR, ya que proporciona una puntuación que va de 0

a 100, lo que facilita su interpretación, a diferencia del PSNR, que opera en una escala logarítmica [7].

3.8. SSIM (ÍNDICE DE SIMILITUD ESTRUCTURAL)

El SSIM es un parámetro comúnmente empleado para evaluar la calidad de imágenes y secuencias de vídeo. Se utiliza como un método predictivo para estimar la calidad percibida de imágenes digitales en televisión, cine y otros medios visuales. Originalmente concebido para medir la calidad visual de una imagen comprimida en relación con la imagen original sin comprimir o sin distorsiones, el SSIM se ha extendido a diversas formas de imágenes y vídeos digitales[8].

Su idea es medir la similitud estructural entre las dos imágenes, en lugar de una diferencia de píxel a píxel como lo hace PSNR, además que está métrica no estima errores absolutos [9] .

4. DESCRIPCIÓN DEL TRABAJO DESARROLLADO

4.1. EVOLUCIÓN HISTÓRICA

La compresión y la descompresión de video se realiza mediante un codificador y un decodificador, lo que normalmente se denomina códec [10]. En la siguiente figura [10] se observa el funcionamiento de dicho códec.

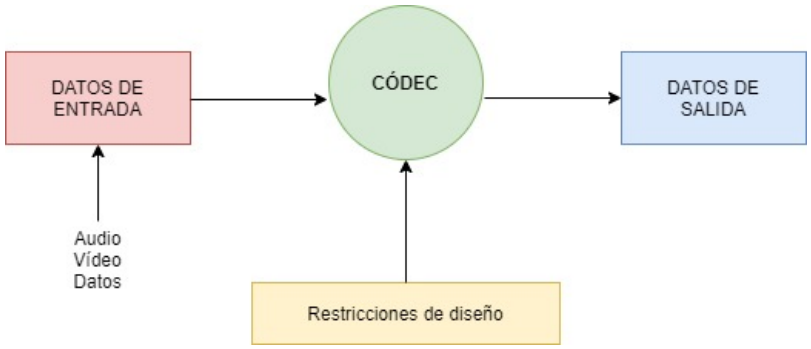


Figura 4. Diagrama general de un códec [11].

El códec se encarga de codificar y comprimir los datos de un archivo de video en este caso para agilizar su transferencia y disminuir su tamaño.

Según han ido transcurriendo los años, se han diseñado diferentes estándares o algoritmos para realizar dicha compresión. Los dos organismos más relevantes encargados de su realización han sido:

- ITU (*International Telecommunications Union- Telecommunications*, antes conocida como International *Telegraph and Telephone Consultative Committee*), responsable de la regulación de las telecomunicaciones a nivel global entre las diversas administraciones y compañías operadoras.
- ISO (*International Standards Organization*) es una entidad que se dedica a establecer normas a nivel mundial y está conformada por distintas organizaciones nacionales de normalización.

De manera general, se pueden diferenciar dos tipos de códecs[13]:

- Códecs con pérdidas matemáticas: funcionan eliminando información redundante de las imágenes con el objetivo de minimizar la reducción de calidad perceptible para el espectador. Sin embargo, esta información eliminada no puede ser recuperada y la cantidad de información perdida está directamente relacionada con el nivel de compresión, lo que a su vez afecta la calidad. Para obtener la máxima calidad en un archivo con un tamaño determinado, se debe aplicar una compresión mínima. Por otro lado, si se prioriza reducir el tamaño del archivo, se puede aplicar una compresión más agresiva, aunque esto conlleve una pérdida de calidad más significativa. En la práctica, una compresión efectiva que minimice la pérdida de calidad es muy útil, siempre y cuando no se necesite acceder a los datos perdidos en el futuro. A nivel prácticos, resulta muy útil una buena compresión con una mínima

pérdida de calidad, siempre y cuando no necesitemos utilizar los datos perdidos en un futuro.

- Códecs sin pérdidas matemáticas: mantienen intactos los datos originales y garantizan que después de la compresión las imágenes sean las mismas, por tanto, son capaces de recuperar toda la información tal y como era originalmente. Entre estos codificadores se destacan los siguientes métodos:
    - LZ, son una familia de técnicas de codificación sin pérdidas basadas en diccionarios. La idea principal fue el sustituir cadenas de texto por un puntero a la zona de texto ya codificada, que se almacena como un diccionario, donde se había producido esa misma cadena de texto. En las imágenes, donde los símbolos son los valores que toman cada una de las muestras que componen un píxel.
    - Codificación *Huffman*, es una técnica de compresión y codificación diseñada para minimizar la cantidad media de bits necesarios para transmitir un símbolo cuando se trata de transmitir múltiples copias independientes y estadísticamente equivalentes de ese símbolo. Este método establece cómo representar los diferentes valores del símbolo como cadenas binarias, asignando códigos más cortos a los símbolos más probables y códigos más largos a los símbolos menos probables.
    - Codificación aritmética, es un método donde una cadena de caracteres se representa con un número fijo de bits por carácter, como se hace en el código ASCII. Sin embargo, cuando se aplica la codificación aritmética, los caracteres más frecuentes se almacenan con menos bits, mientras que los menos frecuentes se almacenan con más bits. Esto reduce el número total de bits utilizados. En la codificación aritmética, todo el mensaje se convierte en un solo número que está en el rango de 0 a 1.

  Estos códecs usan normalmente el sistema RLE (*Run Length Encoding*), que recordemos que es una compresión en la que las secuencias consecutivas con el mismo valor son almacenadas como un valor único, es decir, que intenta descartar las áreas que hay entre imagen e imagen y que son de color similar. Cabe destacar, que RLZ se usa tanto en códecs con pérdidas como sin perdidas.

  A efectos prácticos las técnicas de compresión sin perdidas no son muy efectivas en el video ya que, este tiene pocas áreas de color continuo y está formado por numerosas variaciones de color, además tienen poca aplicación y solo se usa en datos que estamos seguros que vayan a sufrir tratamientos posteriores.

Si en lugar de esta clasificación se tiene en cuenta la velocidad de codificación y decodificación se puede diferenciar:

- Códecs con simetría, misma velocidad de codificación y decodificación.
- Códecs sin simetría, grandes diferencias entre sus velocidades de codificación y decodificación.

Una vez se introduce de forma rápida en la definición de códec, así como algunos tipos de estos y los organismos que lo regulan, vemos algo más importante, el paso del tiempo. Sabemos que los estándares van cambiando a lo largo del tiempo al igual

que la tecnología de ahí que en la siguiente figura se pueda observar la evolución de algunos de los estándares más importantes a lo largo del tiempo, en concreto desde 1988 hasta el más actual en 2020.

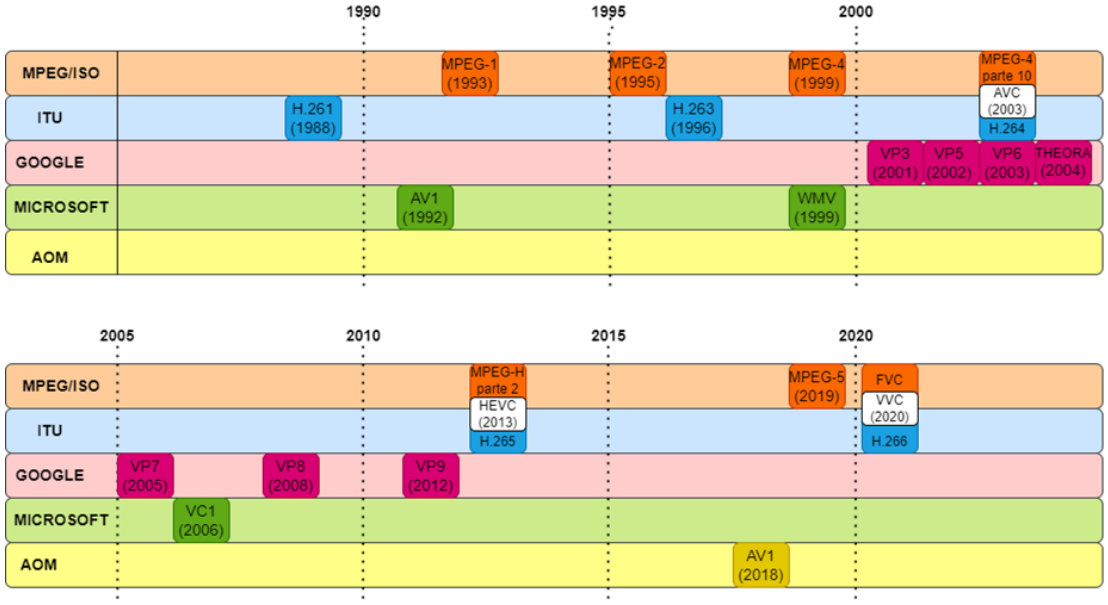
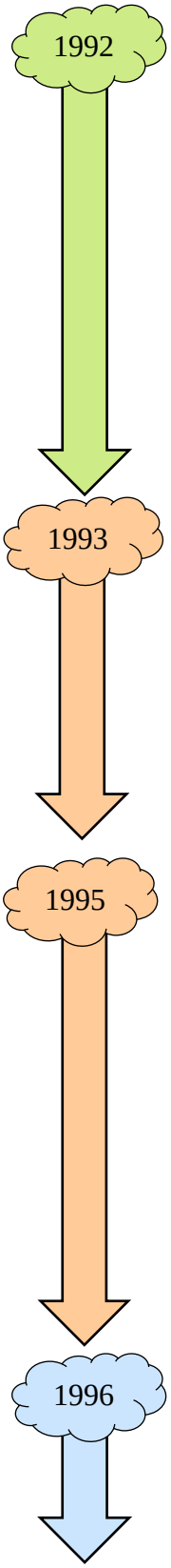


Figura 5. Evolución histórica estándares de codificación.

Una vez vista la figura se procede a la explicación simplificada de cada uno de los estándares para tener un ligero conocimiento de cada uno de ellos.

1988

En primer lugar, debemos mencionar **H.261**, un estándar desarrollado por la ITU y aprobado en noviembre de 1988, originalmente diseñado para su aplicación en videoconferencias. Este estándar fue concebido para operar dentro de un rango de tasas de bits que van desde 40 kbit/s hasta 2 Mbit/s. Soporta dos tamaños de fotograma de vídeo: CIF (*Common Interchange Format*) con una resolución de 352x288 píxeles y QCIF (*Quarter Common Interchange Format*) con una resolución de 176x144 píxeles, empleando un esquema de muestreo de 4:2:0. H.261 integra un mecanismo para optimizar la utilización del ancho de banda, buscando un equilibrio entre la calidad de la imagen y la fluidez del movimiento. Por consiguiente, un vídeo con una amplia variabilidad en sus imágenes presentará una calidad inferior en comparación con uno que exhiba imágenes más homogéneas y con menos movimiento. Esto se debe a que H.261 se basa en un flujo constante de información (*constant bit-rate*), en lugar de mantener una calidad de imagen constante (*variable bit-rate*).

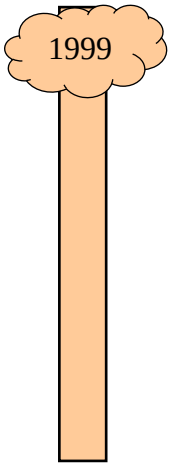


AVI (*Audio Video Interleave*), es un formato contenedor de audio y vídeo desarrollado por Microsoft en noviembre de 1992. Su función principal es permitir el almacenamiento simultaneo de un flujo de datos de vídeo y varios flujos de audio. El formato del flujo no es objetivo del AVI ya que, es interpretado por un códec externo, el audio y video que contiene AVI pueden estar en cualquier formato de ahí que se considere formato contenedor. Para reproducir simultáneamente ambos flujos se necesita que se vayan almacenando de manera entrelazada así, cada fragmento de archivo tiene la información necesaria para reproducir unos pocos fotogramas junto al sonido que corresponda. Hoy en día, este formato sigue siendo de los más populares entre los formatos de video tanto para aficionado como profesiones de la producción de vídeo. Además de su fácil reproducción en Windows Media Player o Google Drive.

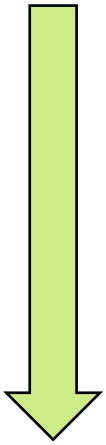
MPEG-1 nació en 1993, fue el primer estándar de codificación tanto de audio como de video del grupo MPEG (*Moving Pictures Experts Group).* Se utilizaba para el formato de video CD (Compact Disc), es decir, para sustituir las cintas de video VHS (*Video Home System*) usando CD, aunque su calidad es parecida a la de las cintas de vídeo domésticas. Permitía una resolución de 352x288 píxeles y empleaba un método de compresión que consistía en comparar cada imagen con la siguiente y almacenar únicamente las diferencias entre ellas. Esto permitía alcanzar niveles de compresión muy altos.

Si continuamos por el año 1995, encontramos **MPEG-2**, también conocido como SVCD. Este estándar proporciona al vídeo una mayor calidad y resolución (720x576) en comparación con su predecesor, que se ha vuelto obsoleto para la distribución de películas a gran escala.
MPEG-2 fue aceptado para la transmisión de vídeo digital comprimido con velocidades mayores de 1MB/s. Es comúnmente empleado para codificar tanto audio como vídeo en señales de transmisión como televisión digital terrestre, por satélite o por cable. Aunque con ciertas mejoras, sigue siendo un estándar vigente en la televisión de alta definición en algunos países. Utiliza soporte de video entrelazado además de conseguir tasa de compresión muy elevadas de hasta 100:1 dependiendo del vídeo. MPEG-2 es el estándar utilizado en los DVD-Video, recordamos que el DVD tiene una capacidad 8 veces mayor que los CD.

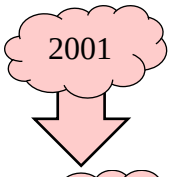
H.263 desarrollado por la ITU en 1996, se utiliza para codificar vídeos con compresión. Este códec fue diseñado principalmente para videoconferencias ya que, tiene una muy baja tasa de bits y un movimiento muy reducido. Es ideal para secuencias de vídeo con pocos cambios entre imágenes.



Se continúa con **MPEG-4** [14], nacido en 1999 con el principal objetivo de videoconferencias, pero pronto se vio claro que su uso principal aplicaciones sería en Internet, es decir, en la multimedia en general. Fue creado para velocidades de bits muy bajas, pero hoy en día admite hasta 4Mbps. Este estándar de codificación de video es utilizado en transmisión, en entornos conversacionales e interactivos, también lleva a cabo una escalabilidad del contenido ya que, puede usarse tanto en entornos web como en televisión. Entre sus ventajas destaca: integra contenido sintético y natural en un mismo objeto, soporta 2D y 3D, puede codificar desde velocidades muy bajas (5kbps en vídeos) hasta velocidades muy altas (5Mbps). Tiene un gran impacto en producción, programación y archivo y en la reutilización de activos de video digital.

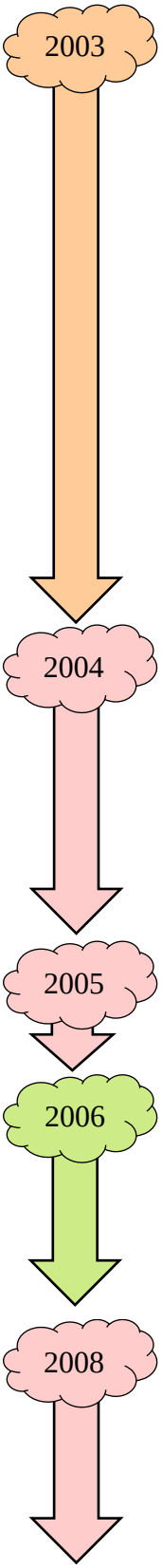


En este mismo año se encuentra **WMV (***Windows Media Video***)** [15] que además de un archivo de video del formado *Windows Media Video*, muy utilizado y más antiguo, y un contenedor también es conocido como un códec. Lo que esto significa es que mientras el usuario ve un archivo WMV, en realidad hay tres archivos en uso: un archivo de audio, un archivo de video y el archivo contenedor. Este códec comprime los datos de video en un formato más pequeño e indica al reproductor de video como descomprimirlo. Presenta ventajas como la gran cantidad de datos que conservan a pesar de su pequeño tamaño, su uso en sitio web sin aumento de tiempo de carga. Con respecto a sus inconvenientes el principal es su incompatibilidad con las plataformas que no son de Windows.



Le sigue en 2001, **VP3.** Originalmente, un códec de vídeo propietario creado por On2 Technologies. En términos de calidad y velocidad de bits, es comparable al códec de vídeo MPEG-4.

En 2002 le sigue el códec **VP5** similar al códec anterior (VP3).



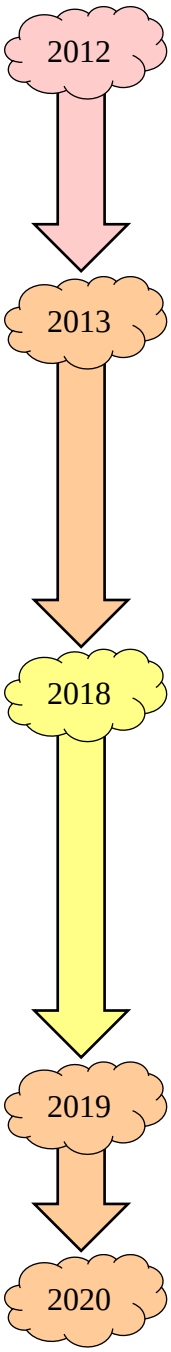
A este le sigue en 2003 **VP6**, un códec más avanzado y con una calidad superior, pero con poco cambio respecto a sus antecesores.
En el mismo año, se introduce **H.264**, también conocido como **MPEG-4 parte 10**, un estándar que establece un códec de vídeo altamente comprimido. Su objetivo principal era ofrecer una calidad de imagen superior a tasas de bits más bajas que las versiones anteriores, sin aumentar la complejidad del diseño ni los costos. Para acelerar su desarrollo, la ITU y la ISO se unieron, lo que resultó en el nombre híbrido H.264/MPEG-4 AVC. En cuanto a su estructura, hubo pocos cambios significativos. Algunas técnicas de codificación de vídeo que previamente se habían descartado debido a su complejidad y costos fueron reconsideradas para su inclusión en este estándar.
Actualmente este estándar es el más utilizado para la grabación, compresión y distribución de video, se encuentra en su versión 26 que se lanzó en 2019. Empresas como Netflix, *YouTube* y software como Adobe Flash Player utilizan este códec, así como algunas transmisiones de HDTV por tierra, cable o satélite.

Hasta el momento, el único competidor significativo de H.264 en la web era este formato, desarrollado en 2004 a partir del código fuente de VP3. **Theora** actúa como el códec de vídeo, mientras que OGG sirve como el contenedor. Theora es un formato de compresión de vídeo de código abierto y gratuito, lo que significa que está disponible para su uso por cualquier persona. Aunque su principal desventaja es que los codificadores y decodificadores no están tan optimizados como su antecesor y necesitan un 25% más de recuso, además de los ficheros que genera son más grandes.

En 2005, encontramos **VP7** es el sucesor de VP6 y está desarrollado por Google destaca que está preparado para competir con códec como H.264 o MPEG-4.

En abril de 2006, se publicó el estándar **VC-1**, también conocido como *Windows Media Video 9*. El objetivo principal de este estándar es proporcionar soporte para la compresión de contenido entrelazado sin necesidad de convertirlo previamente a progresivo. Esto lo convierte en una opción más atractiva para los profesionales de la industria de la difusión y el vídeo. *Blu-ray* y videoconsolas como Xbox 360 utilizan este estándar.

VP8 es una especificación desarrollada en 2008 para codificar y decodificar video de alta definición, ya sea como un archivo de video o como un flujo de bits para la visualización. Parece que es muy novedosos, pero usa las mismas ideas que los formatos anteriores. Su diseño está hecho para manejar imágenes de los videos con submuestreo de croma 4:2:0 con color de 8 bits por canal, escaneo progresivo y grandes dimensiones de imagen.



En 2012, encontramos **VP9** [16], se utiliza para comprimir y descomprimir video, fue desarrollado por Google de ahí que sea gratuito y puede ser utilizado por cualquier fabricante. Con respecto al anterior, es un 50 % más eficiente. Su diseño está hecho para resoluciones HD, 4k y 8k aunque necesita muchos datos. Es el más eficiente a la hora de comprimir video de *YouTube*, ya que, con la misma conexión es capaz de reproducir un vídeo con el doble de resolución.

En el siguiente año está **H.265**, un estándar importante puesto que más adelante en este documento se verá una comparación de este con el protagonista de este trabajo. También conocido como HEVC reduce a la mitad el espacio que ocupa un contenido ya que, al codificar con una tasa más baja, tiene menor ancho de banda y más calidad. Sin embargo, hay pocas plataformas que lo usan debido a su complejidad y a la extensión de su antecesor, no es óptimo para 8k de ahí que nazcan otro mejores. He de destacar que se utiliza por ejemplo en sistemas de videovigilancia donde se utilizan cada vez más las cámaras con resolución superior a Full HD.

Dando un salto al 2018, encontramos **AV-1**[17] un códec de vídeo de alto rendimiento y sin licencia que también puede usarse en imágenes o audios, está diseñado principalmente para ayudar a las empresas a transmitir videos online con alta calidad y eficiencia. Entre sus ventajas aparece que sus vídeos tienen mayor calidad y rapidez, así como su compresión que es dos veces más rápida que la competencia. Sus usuarios pueden disponer de gran calidad, aunque su ancho de banda de internet sea muy bajo. Su uso está muy extendido, por ejemplo, en navegadores como Chrome, Firefox, distribuidores de contenido multimedia como Netflix, Amazon, Apple, compatibilidad con televisores Samsung…

Poco tiempo después, en el año 2019, le sigue **MPEG-5** cuya característica principal es que posee la misma calidad que H.265 pero con menor tasa de bits. Sus usos principales incluyen compresión de datos en *streaming*, difusión de aplicaciones de televisión y distribución de archivos.

Por último y terminando con la evolución historia encontramos el novedoso estándar **H.266** del que hablamos más extensamente.

4.2. CARACTERÍSTICAS GENERALES CODIFICADOR DE VÍDEO H.266

Este estándar de vídeo data [18] de julio de 2020, de ahí que sean tan novedoso y difícil de encontrar información, el Equipo Conjunto de Expertos en Vídeo (JVET), un proyecto de colaboración del Grupo de Expertos en Vídeo (VCEG (*Video Coding Experts Group*)) del UIT-T y el Grupo de Expertos en Imágenes en Movimiento (MPEG) de la ISO/CEI, finalizó esta nueva norma de codificación de vídeo denominada nueva norma de codificación de vídeo denominada *Versatile Video Coding* (VVC) más conocida como “Codificación de Video Versátil”. VVC es la sucesora de la norma de codificación de vídeo de alta eficiencia (HEVC) y ha sido

publicada por el ITU-T como H.266 y por ISO/IEC como MPEG-I Parte 3 (ISO/IEC 23090-3).

Sus principales objetivos a la hora de diseñarlo fueron, en primer lugar, crear una nueva tecnología de codificación de vídeo con una capacidad de compresión por encima de los estándares anteriores, por ejemplo, ofrece un 50% más de compresión con respecto a su antecesor el estándar H.265. El otro objetivo, no menos importante que el anterior, es que se trata de una tecnología muy variable o versátil para usar eficazmente con una amplia gama de aplicaciones que los estándares anteriores no podían plantear.

Su característica principal es ofrecer codificación de calidad mejorada en vídeo de ultra alta definición (UHD), es decir, con resolución de imagen de 3840x2160 o 7620x4320 y una profundidad de bits de 10 bits. Aunque también cabe destacar su uso en video con alto rango dinámico (con gran capacidad de que la cámara capte el detalle en las luces y sombra de la imagen). Este estándar tiene la capacidad de habilitar una amplia variedad de aplicaciones emergentes, tales como multimedia inmersiva omnidireccional de 360 grados, realidad aumentada, uso compartido de pantalla remota, colaboración basada en la nube, juegos en la nube y extracción y fusión basadas en regiones.

expande el conjunto de opciones técnicas disponibles para respaldar el video, sin embargo, los estándares anteriores seguirán siendo relevantes y permitirán la provisión de aplicaciones y servicios de video en todo el mundo [29].

4.3. ARQUITECTURA DEL CODIFICADOR DE VÍDEO H.266

El diseño de VVC [21] sigue el concepto general de codificación de video híbrida basada en bloques. Las imágenes de vídeo se dividen en bloques rectangulares y a cada bloque se le resta otro resultante de aplicar una técnica de predicción (*intra-frame* o *inter-frame)*..

El bloque resultante, también denominado de error o residuo, se transforma al dominio de la frecuencia y se cuantifica, para posteriormente codificarlo mediante un codificador de tipo entrópico..

Los artefactos que resultan del proceso de cuantificación se atenúan aplicando los llamados filtros en bucle a las imágenes reconstruidas antes de que se emitan o se utilicen como referencias para la predicción entre imágenes de las siguientes imágenes.

Aunque VVC usa el mismo marco de codificación que sus predecesores, incluye varias mejoras que finalmente dan como resultado un rendimiento de compresión sustancialmente mejorado. Uno de los cambios más destacados en comparación con sus predecesores es el concepto de partición de bloques muy flexible que admite bloques no cuadrados para la selección del modo de codificación, la predicción dentro de la imagen, la predicción entre imágenes y la codificación de transformación y, por lo tanto, afecta el diseño de muchos otros aspectos.

A continuación, se desarrollan cada uno de los bloques de su arquitectura, pero para entenderlo mejor y no perdernos ningún paso se sigue el diagrama inferior donde se observan cada una de sus partes.

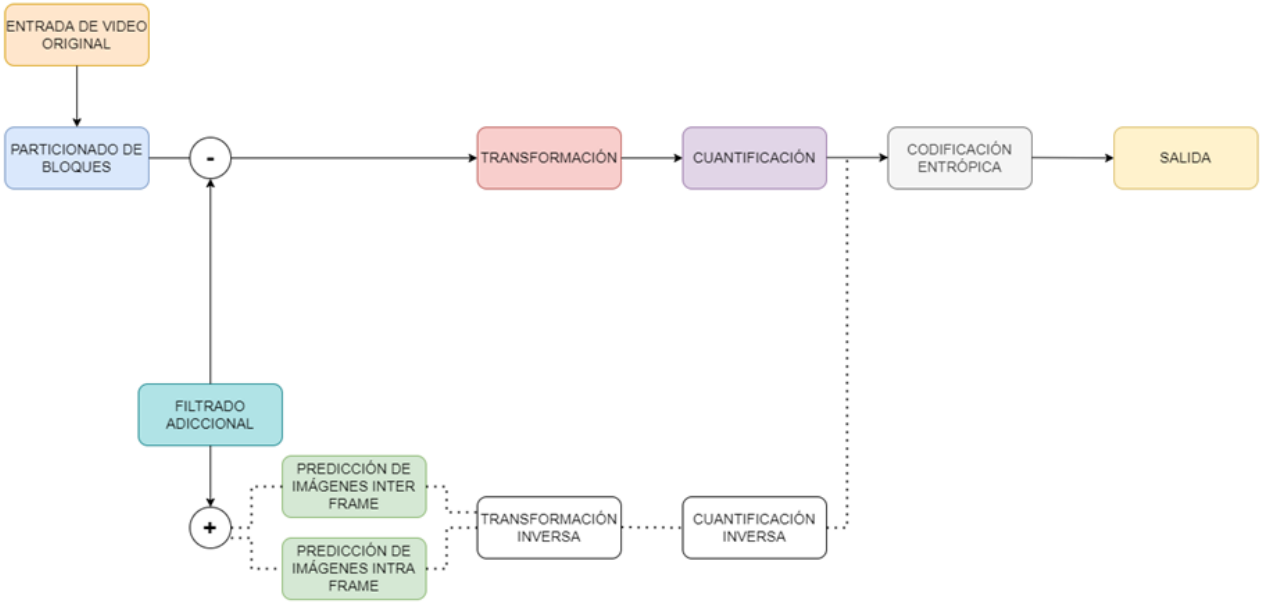


Figura 6. Diagrama de bloques arquitectura VCC.

4.3.1. Particionado de bloques

En la primera etapa de este diagrama aparece el particionado de bloques [24], a grandes rasgos consiste en la división de la imagen en bloques más pequeños para poder realizar los procesos de predicción y transformación.

Los primeros estándares usaban bloques con tamaño fijo, normalmente 16x16 para la predicción de luma. A partir de H.263 esta etapa empezó a convertirse en la parte más importante del diseño, de este modo la partición de bloque ha ido evolucionando para ser más flexible agregando diferentes tamaños y formas de bloque.

En la etapa de predicción, esto permite que un codificador intercambie una alta precisión para la predicción (usando bloques pequeños) versus una velocidad de datos baja para la información de predicción a señalizar (usando bloques grandes).

En la codificación, los bloques pequeños permiten codificar detalles más finos mientras que los bloques grandes pueden codificar regiones suaves de manera eficiente.

Con el aumento de las posibilidades de dividir una imagen en bloques, la complejidad de un codificador que necesita probar las posibles combinaciones y decidir cuál seleccionar también aumenta en comparación con un tamaño fijo o un conjunto de particiones limitado.

Sin embargo, los algoritmos de particionamiento rápidos y los avances en la potencia informática han permitido que los estándares recientes proporcionen un alto grado de flexibilidad.

En el contexto del VVC, se observa que conserva el enfoque de particionado del HEVC, pero también permite una partición más flexible y el aumento en los tamaños de bloque. La nueva técnica de particionado introducida por VVC implica dividir los bloques en fragmentos de formas no necesariamente cuadradas tanto para el plano de luminancia como para los de crominancia.

Las técnicas más importantes de particionado de bloques que se usan en VCC son [18]:

- *Quadtree plus multi-type tree* (QT+MTT): Este enfoque implica la extensión del particionado cuaternario del HEVC a un árbol multi-tipo, donde se utilizan particionados binarios y ternarios. Además, VVC aumenta el tamaño máximo de bloque de HEVC de 64x64 a 128x128.

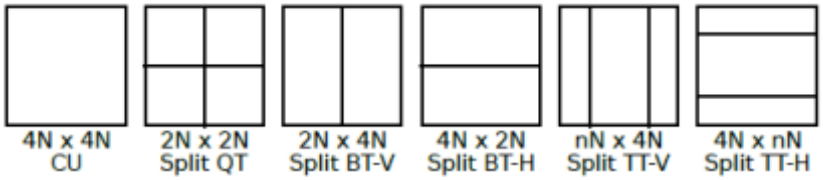


Figura 7. Particionado VCC para una unidad de codificación.

  En la Figura 7 se presentan los diferentes tipos de particionado que ofrece VVC para una unidad de codificación de tamaño 4Nx4N, donde N representa el número de muestras o píxeles de luminancia o crominancia.

  El particionado QT+MTT permite obtener una imagen residual con una energía mínima, resultante de la diferencia entre la imagen actual y las imágenes previas o siguientes, junto con la aplicación de la estimación y compensación de movimiento. Esto conduce a una compresión de vídeo más eficiente.

- *Chroma separate tree* (CST): Dado que el plano de luminancia tiende a tener una textura más fina y bordes más definidos en comparación con el plano de crominancia, se requiere el uso de unidades de codificación de tamaño más pequeño para la luminancia. Por otro lado, en los planos de crominancia, donde los detalles de textura son menores, se pueden utilizar unidades de codificación de mayor tamaño para mejorar la velocidad de codificación. En resumen, VCC permite el uso de un particionado diferente, el que hemos indicado en este punto, para los planos de luminancia y los de crominancia.

- *Virtual pipeline data units* (VPDUs): En los decodificadores VVC implementados en hardware, se encuentran regiones de bloques dentro de la unidad de árbol de codificación (CTU) denominadas VPDUs. Estos bloques

pueden ser decodificados en paralelo con el fin de aumentar la velocidad de decodificación.

A esta etapa le sigue la predicción de imágenes donde aparecen nuevas técnicas para mejorar la compresión de vídeo, dentro de esta etapa está la predicción de imágenes intra e inter frame.

4.3.2. Predicción intra-frame

Esta predicción de imágenes [23] aprovecha la redundancia espacial que existe dentro de una imagen, de ahí el nombre de "intra" [24], al derivar la predicción para un bloque a partir de muestras de referencia especialmente las vecinas ya codificadas o decodificadas. Este tipo de predicción en el dominio de la muestra espacial se introdujo con AVC, mientras que los estándares anteriores usaban una predicción de dominio de transformación simplificada.

VVC usa 67 modos intra, que constan de 65 modos angulares frente a los 33 que tenía HEVC, DC y plano. Para que nos quede más claro se repasan cada uno de estos tres modos. Podemos apreciar, en la Figura 8 los 67 modos de predicción intra-frame de VVC.

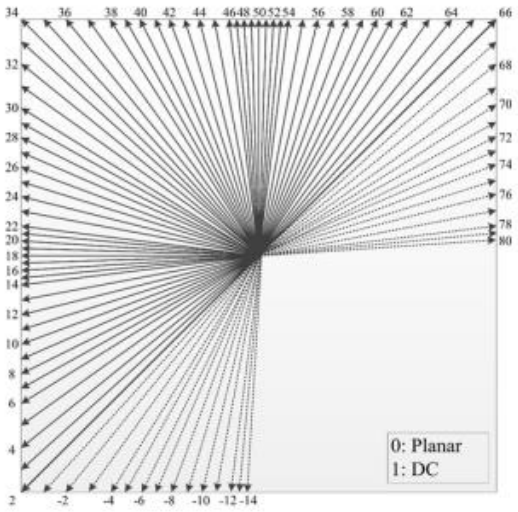


Figura 8. Modos de predicción intra frame en VCC [19]

El modo DC, promedia las muestras de referencia vecinas y utiliza este valor como predicción para todo el bloque, es decir, para cada muestra. El plano, modela las muestras para predecirse como un plano mediante combinaciones lineales dependientes de la posición de las muestras de referencia. Y, por último, los modos angulares interpolan las muestras de referencia a lo largo de una dirección / ángulo específico.

VVC más reciente no solo amplió aún más el número de modos, sino que también incorpora nuevos métodos, como una predicción intra *frame* basada en matrices (MIP), que fue diseñado utilizando aprendizaje automático.

Incrementa el tamaño del bloque en el modelo temporal a 128x128, con opciones de particionado binario o ternario en contraste con el particionado cuaternario de HEVC. Además, permite diferentes particionados para los planos de luminancia y crominancia, y facilita la aceleración del hardware mediante el procesamiento en paralelo. Mientras que los modos intra angulares admiten direcciones de 45 ° a -135 °, para admitir bloques no cuadrados se utiliza un esquema adaptativo para usar modos de gran angular.

Además, se utiliza un filtro de suavizado interno que elige de forma adaptativa un filtro de cuatro tomas en función del modo direccional.

Para reducir aún más la redundancia de componentes cruzados, VVC emplea un esquema de modo de predicción lineal que permite predecir las muestras de croma basándose en muestras de luma reconstruidas. Otra mejora interesante es que VVC amplía las muestras de referencia al permitir el uso de varias líneas de referencia, lo que mejora la calidad de la predicción.

4.3.3. Predicción inter-frame

A grandes rasgos la predicción inter *frame* [22] se refiere a predecir los *frames* actuales a partir de los *frames* anteriores o posteriores. La predicción entre imágenes se aprovecha de la redundancia que existe entre imágenes de una secuencia de video codificada.

Un concepto clave es la compensación de movimiento basada en bloques, donde la imagen se divide en bloques para cada uno de los bloques se usa un área correspondiente de una imagen previamente decodificada, es decir, la imagen de referencia, como predicción para el bloque actual.

Suponiendo que el contenido de un bloque se mueve entre imágenes con movimiento de traslación, el desplazamiento entre el bloque actual y el área correspondiente en la imagen de referencia se denomina comúnmente vector de movimiento de traslación (MV) 2-D [23].

La búsqueda de la mejor correspondencia generalmente se realiza en el codificador mediante una búsqueda de coincidencia de bloques que se conoce como estimación de movimiento. Luego, el codificador envía una señal de los datos de MV estimados al decodificador. H.261 usaba solo MV con valores enteros, y este concepto de compensación de movimiento traslacional se generalizó más tarde mediante el uso de precisión de MV de muestra fraccionaria con interpolación, promediando dos predicciones de una imagen anterior temporalmente y una posterior (predicción bidireccional en videos MPEG-1 y MPEG-2) o de múltiples imágenes de referencia con posiciones temporales relativas arbitrarias (en estándares desde AVC). Además, el uso de múltiples imágenes de referencia desde diferentes posiciones temporales

permite estructuras de predicción jerárquicas dentro de un grupo de imágenes (GOP *(Group of pictures)*), lo que mejora aún más la eficiencia de la codificación [24].

Sin embargo, cuando se utilizan imágenes sucesivas, se introduce un retraso estructural al requerir un orden diferente de las imágenes para la codificación y visualización. El estándar más reciente, VVC, incluso va más allá del modelo de movimiento de traslación.

El movimiento en VVC se puede señalar a través de la transmisión explícita de parámetros de movimiento, a través del modo de omisión o mediante el modo de combinación, que incluye derivar parámetros de movimiento de candidatos espaciales y temporales. El modo de combinación utiliza candidatos de vector de movimiento (MV) espacial, temporal y cero como HEVC, además de un vector promedio por pares y un vector basado en el historial de una tabla FIFO *(First In, First Out*).

Se introduce un nuevo esquema llamado modo de fusión con diferencia de vector de movimiento (MMVD) que refina el movimiento derivado a través de una diferencia de vector de movimiento (MVD *(Motion Vectors Difference*)).

Además, se introduce la codificación MVD simétrica, donde los índices de imagen de referencia de lista-0 y lista-1, y el MVD de lista-1 se obtienen en el decodificador.

Para compensar diferentes tipos de movimiento no traslacional tanto en el modo de fusión como en la Predicción avanzada de vectores de movimiento (AMVP) afín, VVC emplea una predicción de compensación de movimiento afín.

El movimiento se señala mediante información de movimiento de dos o tres puntos de control desde las esquinas del bloque. En el lado del decodificador, el movimiento de cada ubicación se deduce en función de esta información.

Por último, VVC introduce la opción de realizar un particionado geométrico de los bloques, lo que significa que se pueden dividir los bloques de codificación de forma no horizontal durante la predicción. Esta técnica de particionado geométrico contribuye a encontrar bloques similares en la imagen de referencia, lo que resulta en una mejora significativa en la compresión del vídeo.

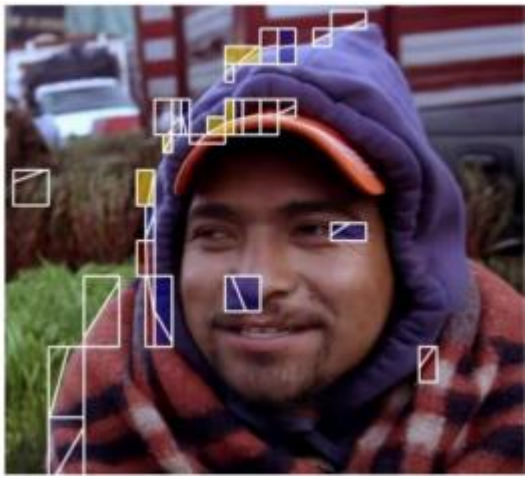


Figura 9. Ejemplo de bloques utilizando el modo de partición geométrica [22]

4.3.4. Filtrado adicional, filtros para reducción de artefactos

En la etapa de filtrado adicional [21], encontramos 3 filtros en bucle, SAO, DBF (*Deblocking Filter*) y ALF (*Alfa Laval Filter*). En comparación con HEVC, que incorpora un filtro de desbloqueo o *deblocking filter* (DBF) y un filtro de compensación adaptativa o *Sample Adaptive Offset Filter* (SAO), VVC introduce un tercer filtro en el codificador, denominado filtro de bucle adaptativo o *Adaptive loop filter* (ALF). Este último filtro se aplica después de DBF y SAO, en bloques de 4×4 píxeles.

ALF utiliza un esquema de clasificación de bloques para elegir entre 25 filtros diferentes para cada bloque de luminancia, según la dirección y el nivel de los gradientes locales. Se aplica una forma de diamante de 7 × 7 para el componente de luminancia y una forma de diamante de 5 × 5 para los componentes de croma [24].

En el decodificador, la selección del filtro luma se basa, como bien se menciona con anterioridad, en la clasificación local de un bloque de 4×4 en 25 clases. La señalización de un índice en un conjunto que contiene 25 filtros de luminancia y un índice para uno de los 8 filtros de crominancia en el nivel de CTU permite un alto grado de adaptabilidad local. El codificador determina los coeficientes de filtro y los parámetros de recorte, y se pueden señalizar múltiples conjuntos por secuencia de video codificada usando un APS (metodología que integra el aprendizaje de contenidos).

En conjunto con los filtros preexistentes en HEVC, se consiguen mejoras en las imágenes reconstruidas. Esto se traduce en una mejora en la precisión de los predictores, lo que a su vez reduce la cantidad de información residual que necesita ser transformada y cuantificada. [25].

4.3.5. Transformación

La Transformación descorrelaciona una señal transformándola del dominio espacial a un dominio de frecuencia, utilizando una base de transformación adecuada.

En los estándares de codificación de video hibrido aplican una transformación que consiste en la diferencia entre la predicción y la señal de video de entrada original. En el dominio de la transformación, la información esencial normalmente se concentra en un pequeño número de coeficientes. En el decodificador, se debe aplicar la transformada inversa para reconstruir las muestras residuales. Un ejemplo es la transformada de coseno discreta (DCT (*Discrete Cosine Transform*)), que se ha utilizado desde H.261 para la compresión de video híbrida.

En estándares posteriores que comienzan con H.263 versión 3 y AVC, se utilizan transformaciones de complejidad reducida basadas en enteros que a menudo se denominan informalmente DCT, aunque una verdadera DCT utiliza una base trigonométrica.

Para tener en cuenta las diferentes estadísticas en la señal de origen, puede ser beneficioso elegir entre múltiples transformaciones como en HEVC y VVC. Además,

la aplicación de una transformación adicional en los coeficientes de transformación como en VVC puede descorrelacionar aún más la señal.

Centrados en nuestro estándar, VVC usa transformaciones de hasta 64 × 64 para luma y 32 × 32 para muestras de croma que se adaptan a muestras de mayor resolución. Además usa DCT-II ,que se emplea también en H.265/HEVC, además H.266 adopta un esquema de selección de transformación múltiple (MTS) que incluye algunas transformadas recientemente introducidas DST-VII y DCT-VIII, y las matrices de transformación se cuantifican con mayor precisión que las matrices de transformación en H.265/HEVC.

H.266/VVC también introduce una transformada no separable de baja frecuencia (LFNST (*Low Frequency Non Separable Transforms*)) que se aplica entre la cuantificación inversa y la transformada primaria inversa. En LFNST, se aplica una transformación no separable de 4x4 u 8x8 según el tamaño del bloque de transformación principal

4.3.6. Cuantificación

Su principal objetivo [21] es reducir la precisión de un valor de entrada para disminuir la cantidad de datos necesarios para representar dichos valores, en otras palabras, aproximar los valores de entrada de manera que se minimice la tasa de bits requerida para transmitir los índices de cuantificación mientras no se exceda un cierto error de reconstrucción. Cabe destacar que el proceso de cuantificación se encuentra dentro de los esquemas de compresión con pérdidas, debido a que es un proceso que representa un conjunto grande de valores (posiblemente infinito) con un conjunto más pequeño

En la codificación de vídeo híbrida, la cuantificación se aplica normalmente a las muestras residuales transformadas individuales, es decir, en los coeficientes de transformación, lo que da lugar a niveles de coeficientes enteros.

Este proceso se conoce como cuantificación inversa o escalado y se lleva a cabo en el codificador, que restablece el rango de valores original sin recuperar la precisión.

La pérdida de precisión hace que la cuantificación sea el componente principal del diagrama de bloques de la codificación de vídeo híbrida que introduce la distorsión. La cuantificación junto con el escalado puede verse como una operación de redondeo con un tamaño de paso que controla la precisión. En los recientes estándares de codificación de vídeo el tamaño del paso se deriva del llamado parámetro de cuantificación (QP) que controla la fidelidad y la tasa de bits [24].

Un tamaño de paso mayor (QP más grande) reduce la tasa de bits, pero también deteriora la calidad, lo que, por ejemplo, provoca de vídeo con artefactos de bloqueo y detalles borrosos.

Normalmente, cada muestra se cuantifica de forma independiente, lo que se denomina cuantificación escalar. En cambio, la cuantificación vectorial procesa un conjunto de muestras de forma conjunta, por ejemplo, mapeando un bloque en un vector de un libro de códigos. Al menos desde la perspectiva del decodificador, todos los estándares de codificación de vídeo estándares de codificación de vídeo anteriores a

HEVC han empleado sólo cuantificación escalar. HEVC incluye un truco conocido como ocultación de datos de signo que puede verse como una forma de cuantificación vectorial, y VVC introduce la cuantificación dependiente (DQ), que puede interpretarse como un tipo de cuantificación vectorial de bloque deslizante porque la cuantificación de una muestra depende de los estados de las muestras anteriores.

En resumen, el diseño del cuantificador en VCC se basa en la cuantificación escalar al igual que en AVC y HEVC, pero en nuestro protagonista destaca que basa en la cuantificación escalar con cuantificadores de reconstrucción uniforme además de incluir dos extensiones que pueden mejorar la eficiencia de la codificación a costa de una mayor complejidad del codificador.

4.3.7. Codificación entrópica

La codificación entrópica [24] asigna palabras de código a un conjunto de símbolos fuente de valores discretos teniendo en cuenta sus propiedades estadísticas, es decir, la frecuencia relativa.

Todos los estándares de codificación de video recientes utilizan tablas de codificación de longitud variable (VLC) que asignan palabras de código más cortas a símbolos con una mayor frecuencia de ocurrencia para acercarse a la entropía. La forma de diseñar tablas de palabras clave en estándares anteriores se basaba en la codificación *Huffman*.

VLC se aplica para codificar y decodificar la gran la mayoría de los datos, incluidos los datos de control, los datos de movimiento y los niveles de coeficiente. AVC mejoró aún más el esquema VLC para la codificación de nivel de coeficiente mediante el uso de un VLC adaptable al contexto (CAVLC (*Context-Adaptive Variable Length Coding*)).

Un contexto está determinado por el valor o una combinación de valores de símbolos anteriores, que se pueden usar para cambiar a una tabla VLC diseñada para ese contexto.

Además, AVC fue el primer estándar de codificación de video que introdujo la codificación aritmética binaria adaptable al contexto (CABAC) como un segundo método de codificación de entropía más eficiente. CABAC (*Context Adaptative Binary Arithmetic Coder*) todavía usa tablas VLC para mapear símbolos. Sin embargo, las cadenas binarias no se escriben directamente en el flujo de bits, sino que cada bit de la cadena binaria se codifica adicionalmente mediante codificación aritmética binaria con modelos de probabilidad adaptables al contexto.

Debido a su alta eficiencia, CABAC se ha convertido en el único método de codificación de entropía en los estándares HEVC y VVC posteriores.

Con respecto a nuestro estándar VCC como bien se comenta en el párrafo anterior, utiliza codificación aritmética binaria adaptativa al contexto (CABAC), al igual que su antecesor el estándar HEVC, aunque presenta algunos cambios.

Pero ¿en qué consiste CABAC y cuáles son esos cambios con respecto a HEVC?

Se trata de una técnica de compresión sin pérdidas, aunque generalmente se utiliza en estándares de codificación de vídeo diseñados para aplicaciones de compresión con pérdidas. CABAC sobresale por ofrecer una compresión considerablemente mejor que la mayoría de los otros algoritmos de codificación de entropía utilizados en la codificación de vídeo, y proporciona al esquema de codificación una capacidad de compresión superior a la de sus predecesores.

CABAC emplea múltiples modos de probabilidad para diferentes contextos. En primer lugar, convierte todos los símbolos no binarios en binarios. Luego, para cada bit, el codificador elige qué modelo de probabilidad utilizar, y utiliza información de elementos cercanos para optimizar la estimación de probabilidad. Finalmente, aplica la codificación aritmética para comprimir los datos.

El modelado de contexto en CABAC proporciona estimaciones de probabilidades condicionales de los símbolos de codificación. Mediante la utilización de modelos de contexto apropiados, se puede explotar la redundancia entre símbolos al cambiar entre diferentes modelos de probabilidad según los símbolos ya codificados en la vecindad del símbolo actual. Esta técnica de modelado de contexto es responsable de la mayor parte de los ahorros, aproximadamente un 10%, en la tasa de bits de CABAC en comparación con el método de codificación de entropía CAVLC.

En la siguiente figura se puede observar cada una de las etapas de esta codificación y describirlas con algo más de detalle

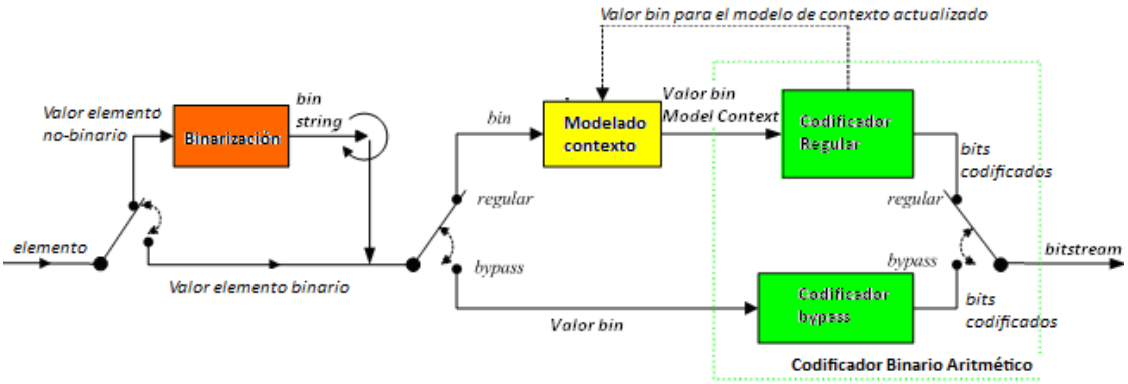


Figura 10. Proceso CABAC [26]

- Binarización es el proceso de transformar los símbolos en palabras código o bins. Cada símbolo de valor no binario se "binariza" o se convierte en un código binario antes de la codificación aritmética.
- Modelado de contexto: implica la selección de un modelo de contexto a partir del cual se calcula la probabilidad de aparición de cada *bin*, basándose en las estadísticas de los símbolos previamente transmitidos. El modelo de contexto almacena la probabilidad asociada a cada *bin* de ser "1" o "0".
- Codificación Binaria Aritmética: un codificador aritmético codifica cada bit conforme al modelo de contexto seleccionado, y finalmente, actualiza el modelo de probabilidad elegido según el valor real del *bin* codificado.

A diferencia de HEVC, el orden de escaneo no depende del modo de predicción intra, ya que se descubrió que dicho escaneo dependiente del modo proporciona solo mejoras insignificantes y complicaría innecesariamente el diseño. VVC utiliza una restricción basada en bloques de transformación en la cantidad de contenedores codificados por contexto para mantener una complejidad similar en el peor de los casos que HEVC

4.4. COMPARATIVA ESTÁNDAR H.265 Y H.266

A continuación, se puede observar una tabla comparativa del estándar principal del trabajo, H.266 y su antecesor el H.265. De este modo, se comprende más fácilmente sus diferencias y similitudes.

Bloques	H.265	H.266
Particionado en bloques	Máximo 64x64	Máximo 128x128
Predicción intra-frame	33 modos angulares	65 modos angulares
Filtrado adicional	DBF SAO	DBF SAO ALF
Transformación	DCT-II	DCT-II DCT-VIII DST-VII
Cuantificación	Escalar	Escalar Vectorial Dependiente
Codificación entrópica	CABAC	CABAC

Tabla 1. Resumen comparativa arquitectura H.265 vs H.266

Ahora se procede a realizar una comparación implementando los estándares H.265 y H.266. Para ello lo primero que se hace es obtener un vídeo raw sin copyright, para ellos utilizamos la página pexels.com y obtenemos un vídeo en formado HD cuyo estándar es h.264. Se recuerda que en un vídeo raw tenemos los datos obtenidos directamente del sensor antes de realizar cualquier proceso en la imagen, es decir, sin procesar. En definitiva, las imágenes del vídeo completamente descomprimida.

Una vez tenemos el vídeo se hacen las modificaciones para cada estándar respectivamente. Para H.265 utilizaremos la herramienta FFmpeg y para codificar H.266 utilizaremos la implementación del codificador VVC de Fraunhofer (Versatile Vídeo Encoder; VVenC).

4.4.1. Codificación y decodificación en H.265 con FFmeg

FFmpeg es una colección de software de código abierto que posibilita la grabación, conversión (transcodificación) y transmisión de audio y video. Es una herramienta multiplataforma por lo que puede instalarse en cualquier sistema ya sea Mac, *Window* o Linux con facilidad. Puede ser utilizado en una amplia lista de códec a excepción de algunos como por ejemplo el actual H.266.

FFmpeg es un programa de línea de comandos, su uso no es muy complejo, aunque la gran cantidad de parámetros, combinaciones, formatos y características hacen que su uso sea poco intuitivo cuando empezamos a usarlo de cero.
Incluye tres herramientas para tratar los vídeos, aunque se centrará el trabajo en FFmpeg.

- ffmpeg: herramienta principal y más utilizada. Incluye las herramientas de conversión y procesado de vídeo.
- ffplay: reproductor de multimedia básico.
- ffprobe: analizador de contenido multimedia que proporciona información técnica.

FFmpeg nos permite entre otras muchas opciones, especificar la relación de aspecto del video, limitar su duración, rotar o girar los videos, así como realizar numerosas modificaciones en el audio del video como cambios de volumen, insertar o eliminar audios…

Una vez realizada la introducción sobre la herramienta, se comienza con su uso.
Lo primero que se hace es obtener la herramienta en nuestro ordenador, para ver cómo se debe realizar su instalación ir al anexo.
Cuando ya tenemos nuestra herramienta lista para usarse, se busca un video sin *copyright*, lo ideal es encontrar un video sin comprimir, en crudo, pero esto es un poco difícil de encontrar debido a su tamaño. Por tanto, cuando ya tenemos nuestro video, se pueden observar sus características con el siguiente comando:

- *ffmpeg -i video-original.mp4*

De este modo se observa la resolución de nuestro video original, así como los *frames* y numerosas características.
A continuación, lo que se realiza es una conversión de nuestro video original en formato MP4 a video en crudo en formato YUV con 8 bits por píxel y también con 10 bits por píxel. Se recuerda que en crudo es sin compresión ni edición, es decir, sin procesar. Estos dos videos se usarán tanto para la compresión con H.265 como para H.266. Se usa el siguiente comando para 8 bits y 10 bits respectivamente:

- *ffmpeg -i video-HD.mp4 video_HD_raw_8b.yuv*
- *ffmpeg -i video-HD.mp4 -pix_fmt yuv420p10le video_HD_raw_10b.yuv.*

Se repite el comando anterior con cada uno de los vídeos que vamos a usar, es decir, en este caso son tres, video HD, Full HD y UHD.
Para un mejor entendimiento se explica cada elemento del comando anterior:

- *ffmpeg*: herramienta que usamos
- *-i*: indica que se va a especificar un archivo de entrada
- *video.mp4*: archivo de entrada
- *-pixel_format yuv420p10le*: indica que el formato de submuestreo es 4:2:0 de 10 bits.
- *video.yuv*: archivo de salida.

Ahora, se procede a codificar el vídeo en crudo con 4 *bitrates* diferentes y tres resoluciones. Se indica el comando que se utiliza para cada una de las resoluciones y sus *bitrates* respectivamente.

Se empieza con resolución HD, que equivale a 1280x720 píxeles:

- *Bitrate* 128k
  - *ffmpeg -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libx265 -b:v 128k -pix_fmt yuv420p10le video_HD_h265_128k_codif.mp4*
- *Bitrate* 256k
  - *ffmpeg -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libx265 -b:v 256k -pix_fmt yuv420p10le video_HD_h265_256k_codif.mp4*
- *Bitrate* 512k
  - *ffmpeg -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libx265 -b:v 512k -pix_fmt yuv420p10le video_HD_h265_512k_codif.mp4*
- *Bitrate* 1024k
  - *ffmpeg -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libx265 -b:v 1024k -pix_fmt yuv420p10le video_HD_h265_1024k_codif.mp4*

Se continua con resolución Full HD, equivale a 1920x1080 píxeles:

- *Bitrate* 128k
  - *ffmpeg -video_size 1920x1080 -i video_FullHD_raw_8b.yuv -c:v libx265 -b:v 128k -pix_fmt yuv420p10le video_FullHD_h265_128k_codif.mp4*
- *Bitrate* 256k
  - *ffmpeg -video_size 1920x1080 -i video_FullHD_raw_8b.yuv -c:v libx265 -b:v 256k -pix_fmt yuv420p10le video_FullHD_h265_256k_codif.mp4*
- *Bitrate* 512k
  - *ffmpeg -video_size 1920x1080 -i video_FullHD_raw_8b.yuv -c:v libx265 -b:v 512k -pix_fmt yuv420p10le video_FullHD_h265_512k_codif.mp4*
- *Bitrate* 1024k
  - *ffmpeg -video_size 1920x1080 -i video_FullHD_raw_8b.yuv -c:v libx265 -b:v 1024k -pix_fmt yuv420p10le video_FullHD_h265_1024k_codif.mp4*

Por último, se hace la conversión a UHD, que equivale a 3840x2160 píxeles:

- *Bitrate* 128k
  - *ffmpeg -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libx265 -b:v 128k -pix_fmt yuv420p10le video_UHD_h265_128k_codif.mp4*
- *Bitreate* 256k
  - *ffmpeg -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libx265 -b:v 256k -pix_fmt yuv420p10le video_UHD_h265_256k_codif.mp4*

- *Bitrate* 512k
    - *ffmpeg -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libx265 -b:v 512k -pix_fmt yuv420p10le video_UHD_h265_512k_codif.mp4*
- *Bitrate* 1024k
    - *ffmpeg -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libx265 -b:v 1024k -pix_fmt yuv420p10le video_UHD_h265_1024k_codif.mp4*

Como se puede observar se utiliza siempre el mismo comando, pero se va variando su resolución y *bitrate*. En cualquier caso, se explica a continuación el comando genérico:

- *-f rawvideo*: formato de conversion, rawvideo significa sin conversión.
- *-pix_fmt yuv420p10le*: indica que el formado de submuestreo es 4:2:0 de 10
- bits.
- *-video_size*: tamaño del video original, que debe de ir seguido de su resolución. Por ejemplo, 1280x720.
- *-c:v libx265:* codificador utilizado, en este caso va seguido de libx265, porque estamos codificando en H.265
- *-i*: indica que se va a especificar un archivo de entrada, debe de ir seguido del nombre del archivo con su extensión, por ejemplo video.yuv
- *-b:v*: bitrate del video de salida, irá seguido de 2048k o 128k, es decir, del valor de bitrate que queremos que tenga el video de salida

Tras esto, se debe decodificar el video obtenido anteriormente para poder comparar y realizar las métricas correspondientes. Además, se anotan todas las características de estos videos codificados a H.265 para comparar los resultados más adelante.

Para la decodificación se usa el siguiente comando:

- *ffmpeg -i video_HD_h265_128k_codif.mp4 -pix_fmt yuv420p10le video_HD_deco_h265_128k.yuv*

En el cual se debe ir modificando el video de entrada, es decir, el vídeo que se ha codificado anteriormente, y dar un nombre especifico al vídeo de salida, se debe repetir este comando con todas las resoluciones y *bitrate*, es decir, con un total de 12 videos.

Se van anotando los valores de todos estos videos, es decir, el tiempo que tardan en ejecutarse los comandos, el tamaño de los videos resultantes para las diferentes resoluciones y *bitrates*, así después serán útiles para comparar y analizar los resultados.

4.4.2. Codificación y decodificación en H.266 con FFmpeg

Hoy en día, ya es posible realizar la codificación en H.266 con la herramienta FFmpeg. Lo que usaremos es una herramienta de FFmpeg llamada ffmpeg_vvc. Esta herramienta está creada usando el codificador y decodificador de Franhoufer,

pero será más fácil su instalación, integración y ejecución desde la consola de comandos.

Antes de explicar cómo se ha usado, es necesario ver en qué consiste el codificador principal del que se basa, *Fraunhofer Versatile Video Encoder* (VVenC) [5]. Se puso en marcha para ofrecer una implementación de codificador VVC rápida, eficiente y de acceso público. Fraunhofer presenta dos herramientas VVenC y VVdeC que se utilizan respectivamente para codificar y decodificar vídeo en VCC o H.266, en este caso nos centraremos en VVenC para codificar.

El software VVenC se basa en VTM (*VCC test model*) con optimizaciones que incluyen el rediseño del software para mitigar los cuellos de botella de rendimiento (se recuerda que el cuello de botella surge cuando hay un componente que frena el rendimiento general del sistema porque sus especificaciones están descompensadas.), amplias optimizaciones SIMD ("*Single Instruction Multiple Data"* pretende conseguir con una sola instrucción trabajar con múltiples datos, o dicho de otra forma, busca paralelizar tareas.), algoritmos de búsqueda de codificadores mejorados y soporte multihilo para explotar la paralelización.

Además, VVenC admite funciones de codificación del mundo real, como control de velocidad a nivel de fotograma y codificación optimizada perceptualmente para proporcionar una solución de codificación de vídeo flexible, rápida y fácil de usar para el estándar VVC.

Los flujos de bits codificados con VVenC pueden descodificarse con cualquier descodificador compatible con el estándar VVC, por ejemplo, el rápido *Fraunhofer Versatile Video Decoder* (VVdeC)

Resumen de características:

- Implementación de codificador fácil de usar con cinco preajustes de calidad/velocidad predefinidos.
- Codificador VVC optimizado que proporciona aumentos de velocidad para secuencias de prueba HD y UHD en función del preajuste de calidad/velocidad elegida.
- Optimización perceptual para mejorar la calidad subjetiva del vídeo.
- Control de velocidad a nivel de fotograma compatible con codificación VBR.
- Interfaz de codificador en modo experto, que permite un control preciso del proceso de codificación.

A continuación, se procede a instalar VVCEasy para ffmepeg que ofrece un comando sencillo de FFmpeg, VVC Tools, VLC o266player, *VVDEC Web Player*, *Bitmovin VVDec Player*, *libvvdec* y más; para ello se siguen todas las instrucciones de construcción que se puede ver en el anexo.

Tras ello se empiezan a realizar las pruebas, al igual que se había hecho con ffmpeg, se deben hacer todas las pruebas desde la consola de comandos donde se encuentra nuestro ejecutable.

Una vez aquí se inician nuestras pruebas, para ello se utilizan tres videos en diferentes resoluciones HD, Full HD y UHD siempre videos en crudo, es decir,

los vídeos que se obtenían en el apartado anterior y, al igual que antes se va modificando el *bitrate* y la resolución. A continuación, se observan los comandos utilizados, similares a los anteriores, pero como se verá a continuación, aquí se usa el video en crudo de 10 bits.

Se empieza con resolución HD, que equivale a 1280x720 píxeles:

- Vídeo HD, *bitrate* 128k
    - *ffmpeg_vvc -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libvvenc -b:v 128k video_HD_266_128k_codif.mp4*
- Vídeo HD, *bitrate* 256k
    - *ffmpeg_vvc -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libvvenc -b:v 256k video_HD_266_256k_codif.mp4*
- Vídeo HD, *bitrate* 512k
    - *ffmpeg_vvc -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libvvenc -b:v 512k video_HD_266_512k_codif.mp44*
- Vídeo HD, *bitrate* 1024k
    - *ffmpeg_vvc -video_size 1280x720 -i video_HD_raw_8b.yuv -c:v libvvenc -b:v 1024k video_HD_266_1024k_codif.mp4*

Se continua con resolución Full HD, equivale a 1920x1080 píxeles:

- Vídeo Full HD, *bitrate* 128k
    - *ffmpeg_vvc -video_size 1920x1080 -i video_fullHD_raw_8b.yuv -c:v libvvenc -b:v 128k video_FullHD_266_128k_codif.mp4*
- Vídeo Full HD, *bitrate* 256k
    - *ffmpeg_vvc -video_size 1920x1080 -i video_fullHD_raw_8b.yuv -c:v libvvenc -b:v 256k video_FullHD_266_256k_codif.mp4*
- Vídeo Full HD, *bitrate* 512k
    - *ffmpeg_vvc -video_size 1920x1080 -i video_fullHD_raw_8b.yuv -c:v libvvenc -b:v 512k video_FullHD_266_512k_codif.mp4*
- Vídeo Full HD, *bitrate* 1024k
    - *ffmpeg_vvc -video_size 1920x1080 -i video_fullHD_raw_8b.yuv -c:v libvvenc -b:v 1024k video_FullHD_266_1024k_codif.mp4*

Por último, se hace la conversión a UHD, equivale a 3840x2160 píxeles:

- Vídeo UHD, *bitrate* 128k
    - *ffmpeg_vvc -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libvvenc -b:v 128k video_UHD_266_128k_codif.mp4*
- Vídeo UHD, *bitrate* 256k
    - *ffmpeg_vvc -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libvvenc -b:v 256k video_UHD_266_256k_codif.mp4*
- Vídeo UHD, *bitrate* 512k
    - *ffmpeg_vvc -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libvvenc -b:v 512k video_UHD_266_512k_codif.mp4*
- Vídeo UHD, *bitrate* 1024k
    - *ffmpeg_vvc -video_size 3840x2160 -i video_UHD_raw_8b.yuv -c:v libvvenc -b:v 1024k video_UHD_266_1024k_codif.mp4*

Los valores del comando genérico son iguales que en H.265 por tanto, a continuación, solo se explican los valores nuevos, se debe ir al paso anterior si surgen dudas.

- -*libvvenc*: codificador utilizado, en este caso va seguido de libx265 porque estamos codificando en H.265

Al igual que con el estándar anterior, se debe decodificar el video obtenido anteriormente para poder comparar y realizar las métricas correspondientes posteriormente. Además, se anotan todas las características de estos videos codificados a H.266 para comparar los resultados más adelante.
Para la decodificación se usa el siguiente comando:

- *ffmpeg_vvc -i video_HD_266_128k_codif.mp4 video_HD_deco_h266_128k.yuv*

En el cual se debe ir modificando el video de entrada, es decir, el vídeo que se ha codificado anteriormente, y dar un nombre especifico al vídeo de salida, se debe repetir este comando con todas las resoluciones y *bitrate*, es decir, con un total de 12 videos. Al igual que para el estándar anterior, se van anotando los valores de todos estos videos para poder realizar la comparativa.

4.4.3. Análisis de métricas

4.4.3.1. Análisis del PSNR, VMAF y SSIM

Una vez realizados los videos, se empieza con el cálculo de las métricas necesarias para realizar la comparación. Nótese que la comparación a diferencia de otras es de 10 bits por píxel.

Para calcularlas se hace uso del programa *MSU Video Quality Measurement Tool*, que permite calcular fácilmente cada una de las métricas. Es un programa profesional pero en este caso se usa la versión gratuita. Se instala fácilmente (ver anexo A.3), y para su cálculo es necesario disponer del video original, sin comprimir, en crudo de 10 bits por pixel y cada uno de los videos descomprimidos con el formato y bitrate necesario. Además, la herramienta permite realizar comparaciones entre varios videos. Su uso es sencillo, tras seleccionar los videos con todas sus características seleccionamos la métrica que se quiere calcular y a los segundos se obtiene el resultado.

4.4.3.2. Análisis del ratio de compresión

Una vez se obtiene este dato se pasa a calcular el ratio de compresión, para ello lo que se hace es dividir el tamaño del video sin comprimir entre el video comprimido, si el valor obtenido es por ejemplo, 500 se expresará como 500:1, lo que significa que al comprimir el video se ha reducido en 500 veces su tamaño original. Muy importante cuando se hace la división utilizar las mismas unidades.

5. RESULTADOS

En este apartado se muestran los resultados comparativos del códec H.266 con respecto a H.265 y H.264 (se ha incluido este último debido a que sigue siendo el más utilizado en la actualidad), utilizando las métricas objetivas de PSNR, VMAF y SSIM, para resoluciones de HD, FHD y UHD, con *bitrates* de 128 KB, 256 KB, 512 KB y 1024 KB.

Además, también se han calculado los ratios de compresión y los tiempos de ejecución del proceso de codificación.

Dicha comparativa se muestra en diversas tablas y figuras en 3D donde se aprecian las diferencias entre los distintos estándares. Es importante destacar que las comparaciones se han realizado con profundidades de color de 10 bits en vez de 8 bits, que suele ser lo habitual.

5.1. RESULTADOS PSNR

En la Figura 11 se puede observar el funcionamiento del programa para comparar los resultados de las codificaciones en H.265 y H.266, en HD y con un *bitrate* de 1024k.Este proceso se repite para el resto de ejemplos.

Figura 11. Cálculo PSNR en *MSU Video Quality Measurement Tool*

A continuación, se puede observar el resultado del cálculo del PSNR para todas las resoluciones y *bitrates*, incluyendo el estándar H.264, ya que sigue siendo uno de los más utilizados en la actualidad:

BITRATE / VIDEO	128KB	256KB	512KB	1024KB
H.264 (HD)	29.5 dB	32.2 dB	35.4 dB	38.6 dB
H.265 (HD)	29.9 dB	33.7 dB	36.6 dB	39.1 dB

H.266 (HD)	32.2 dB	34.7 dB	37.5 dB	40.2 dB
H.264 (FULL HD)	26.3 dB	28.9 dB	31.7 dB	34.8 dB
H.265 (FULL HD)	27 dB	30.4 dB	33.6 dB	36.1 dB
H.266 (FULL HD)	32.1 dB	33.8 dB	35.9 dB	38.4 dB
H.264 (UHD)	31.6 dB	38.2 dB	42.7 dB	45.6 dB
H.265 (UHD)	34.7 dB	41.8 dB	43.9 dB	47.8 dB
H.266 (UHD)	42.8 dB	45.3 dB	47.3 dB	48.5 dB

Tabla 2. Resultados PSNR H264 vs H.265 vs H.266.

Como se puede apreciar, los valores de PSNR aumentan conforme aumenta el estándar así como con el aumento del valor de *bitrate*. Esto significa que los estándares más recientes, presentan una mayor calidad objetiva, como debe suceder.

Esta progresión del valor del PSNR de acuerdo con el estándar de codificación utilizado y el *bitrate* también se puede observar en las representaciones en 3D de las Figuras 12, 13 y 14.

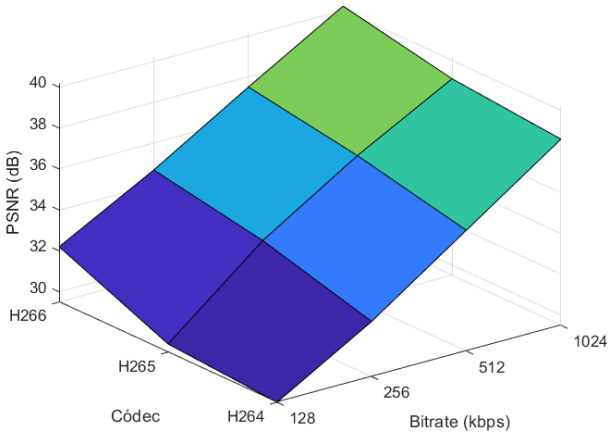


Figura 12. Representación PSNR de vídeos HD en 3D.

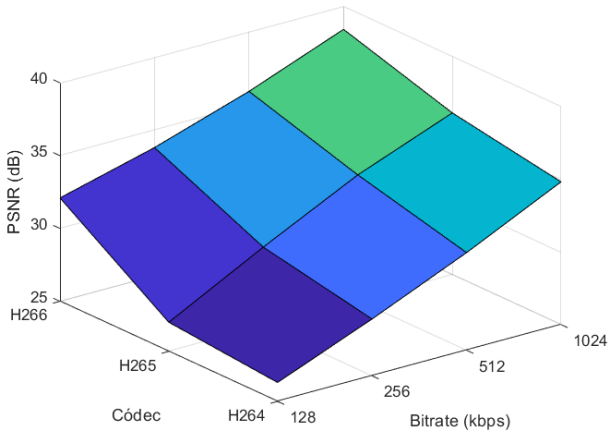


Figura 13. Representación PSNR de vídeos full HD en 3D

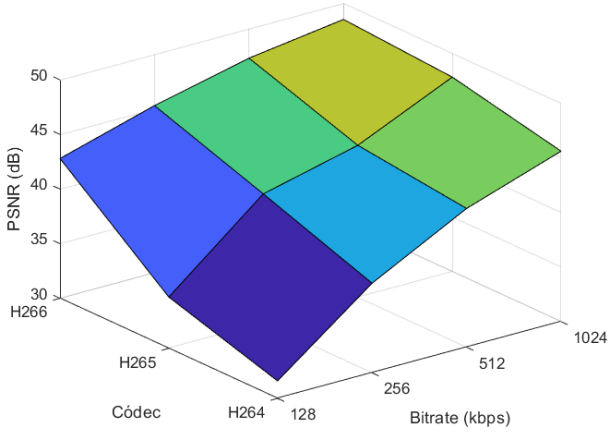


Figura 14. Representación PSNR de vídeos UHD en 3D.

5.2. RESULTADOS VMAF

En lo que respecta a la métrica VMAF, cuya definición ha sido explicada con anterioridad, se presenta un ejemplo de su obtención en la Figura 15, con resolución HD y birate de 1024 KB

Figura 15. Cálculo VMAF en MSU Video Quality Measurement Tool

El resultado de los valores VMAF de acuerdo con las distintas resoluciones, *bitrates* y estándares de codificación probados se pueden observar en la Tabla 3, cuyos valores aumentan conforme aumenta la versión del estándar de codificación y el valor de *bitrate*:

BITRATE	128KB	256KB	512KB	1024KB

VIDEO				
H.264 (HD)	20.4	42.9	67.3	83.3
H.265 (HD)	30.1	58.4	75.7	85.8
H.266 (HD)	50.2	65.2	78	87
H.264 (FULL HD)	6.64	14.6	24.6	34.8
H.265 (FULL HD)	10.2	17.2	41.4	64.5
H.266 (FULL HD)	27.3	43.2	61.4	77.6
H.264 (UHD)	25.3	52.2	77	88.1
H.265 (UHD)	33.9	76.1	90.2	93.6
H.266 (UHD)	74.2	85.8	91.2	94.2

Tabla 3. Resultados VMAF H264 vs H.265 vs H.266.

La visualización de los datos de la tabla para cada una de las resoluciones se puede observar en 3D en las Figuras 16, 17 y 18:

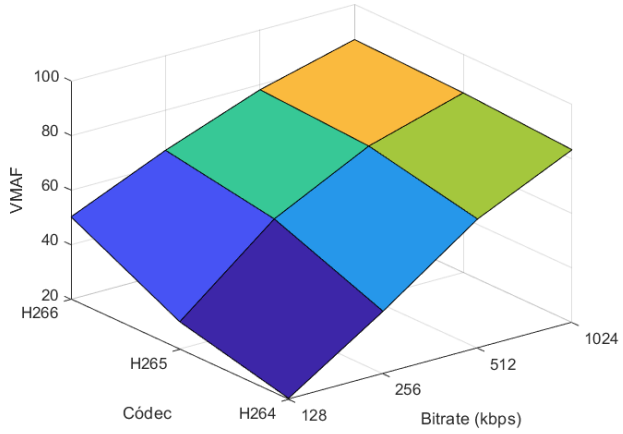


Figura 16. Representación VMAF de vídeos HD en 3D.

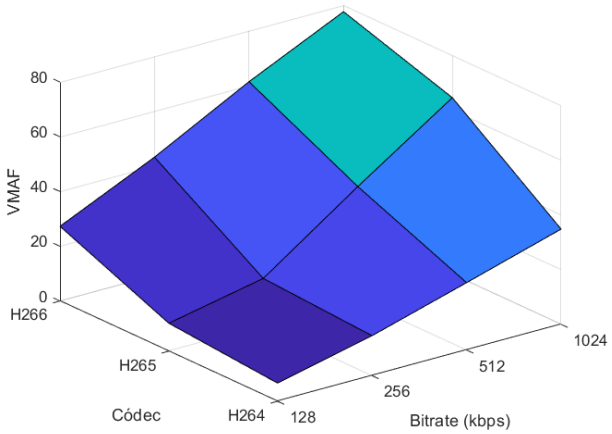


Figura 17. Representación VMAF de vídeos full HD en 3D.

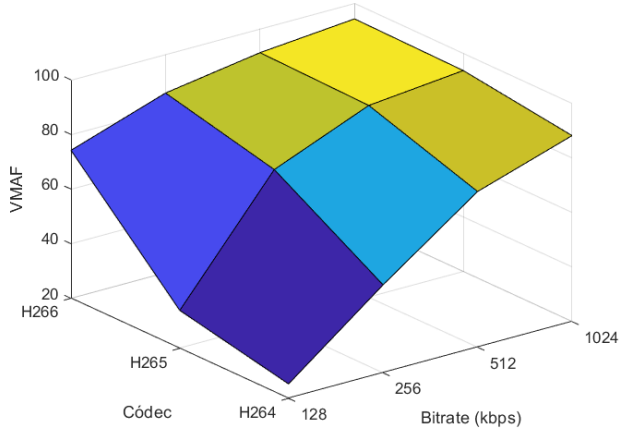


Figura 18. Representación VMAF de vídeos UHD en 3D.

5.3. RESULTADOS SSIM

Por último, para la métrica SSIM, un ejemplo concreto de obtención para HD y con *bitrate* 1024 KB se puede observar en la Figura 19.

Figura 19. Cálculo SSIM en MSU Video Quality Measurement Tool

A continuación, análogamente a las métricas anteriores, se pueden observar los resultados para H.266, H.265 y H.264 con distintas resoluciones y *bitrates* en la Tabla 4.

BITRATE VIDEO	128KB	256KB	512KB	1024KB
H.264 (HD)	0.711	0.813	0.902	0.951
H.265 (HD)	0.763	0.878	0.930	0.959
H.266 (HD)	0.835	0.902	0.946	0.968
H.264 (FULL HD)	0.726	0.695	0.792	0.894
H.265 (FULL HD)	0.730	0.772	0.839	0.896
H.266 (FULL HD)	0.811	0.852	0.898	0.938

H.264 (UHD)	0.953	0.970	0.983	0.989
H.265 (UHD)	0.961	0.983	0.990	0.992
H.266 (UHD)	0.982	0.988	0.991	0.993

Tabla 4. Resultados SSIM H264 vs H.265 vs H.266.

Igual que en los caos anteriores el valor de la métrica va aumentando con respecto al estándar utilizado lo que significa que las pérdidas matemáticas se van reduciendo. Esta información también se puede observar en las gráficas 3D de las Figuras 20, 21 y 22.

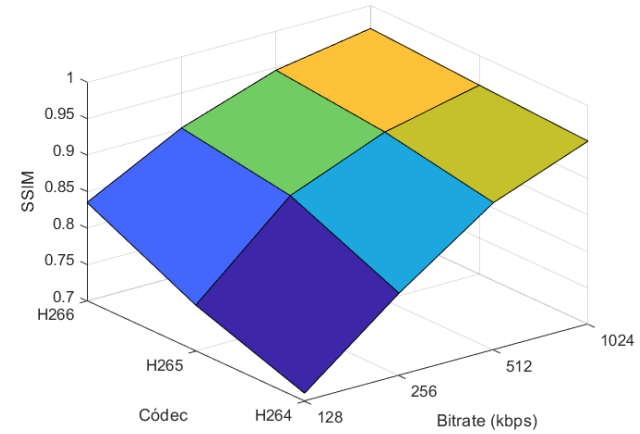


Figura 20. Representación SSIM vídeos HD en 3D.

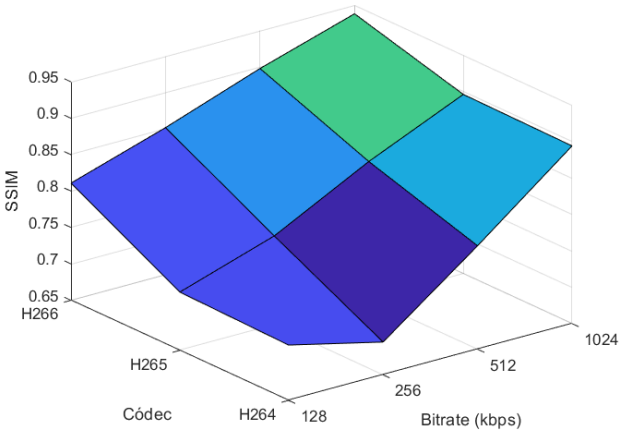


Figura 21. Representación SSIM de vídeos full HD en 3D.

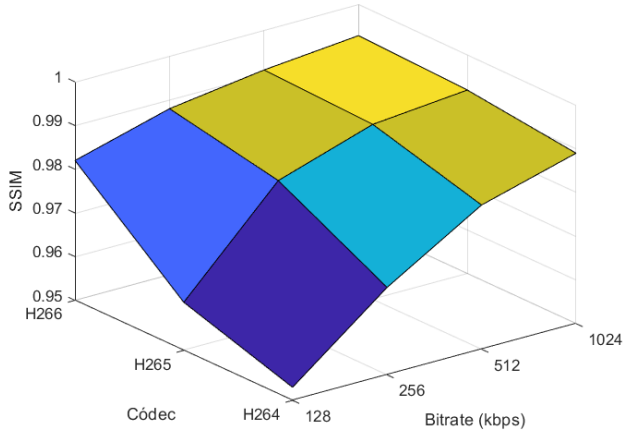


Figura 22. Representación SSIM de vídeos UHD en 3D.

5.4. TAMAÑO DE LOS VIDEOS Y TIEMPOS DE EJECUCIÓN

En la Tabla 5 se pueden observar los tamaños de los videos codificados para cada uno de los estándares. Los tamaños son parecidos para cada resolución y *bitrate*, mostrando cierta reducción en el estándar H.266 para el caso del video en alta resolución y con un alto *bitrate*.

BITRATE / VIDEO	128KB	256KB	512KB	1024KB
H.264 (HD)	249KB	493KB	967KB	1922KB
H.265 (HD)	210KB	472KB	971KB	1941KB
H.266 (HD)	238KB	462KB	914KB	1825KB
H.264 (FULL HD)	141KB	312KB	554KB	1068KB
H.265 (FULL HD)	184KB	254KB	541KB	1106KB
H.266 (FULL HD)	142KB	283KB	556KB	1104KB
H.264 (UHD)	387KB	584KB	1164KB	2360KB
H.265 (UHD)	270KB	516KB	1126KB	2310KB
H.266 (UHD)	321KB	618KB	1232KB	2260KB

Tabla 5. Resultados tamaño codificaciones H.264 vs H.265 vs H.266.

	HD	FULL HD	UHD
TAMAÑO VIDEO RAW	1028700 KB	1458000 KB	12271500 KB

Tabla 6. Tamaño original videos

Otra forma de reflejar el tamaño de los videos codificados es mediante el cálculo de los ratios de compresión, que se pueden observar en la siguiente tabla:

BITRATE / VIDEO	128KB	256KB	512KB	1024KB
H.264 (HD)	4131:1	2086:1	1063:1	535:1
H.265 (HD)	4898:1	2179:1	1059:1	529:1

H.266 (HD)	4322:1	2226:1	1125:1	563:1
H.264 (FULL HD)	10340:1	4673:1	2631:1	1365:1
H.265 (FULL HD)	7923:1	5740:1	2695:1	1318:1
H.266 (FULL HD)	10267:1	5151:1	2622:1	1320:1
H.264 (UHD)	31709:1	21012:1	10542:1	5199:1
H.265 (UHD)	45450:1	23781:1	10898:1	5312:1
H.266 (UHD)	38228:1	19856:1	9960:1	5429:1

Tabla 7. Resultados ratio de compresión H.264 vs H.265 vs H.266

Por otra parte, en la Tabla 7 se pueden observar el tiempo transcurrido en el proceso de codificación para cada estándar evaluado, así como para cada resolución y *bitrate*.

BITRATE / **VIDEO**	128KB	256KB	512KB	1024KB
H.264 (HD)	2min	2min	4min	4min
H.265 (HD)	6seg	7seg	7seg	7seg
H.266 (HD)	2min	2min	4min	4min
H.264 (FULL HD)	1min	1min	2min	3min
H.265 (FULL HD)	3seg	5seg	5seg	6seg
H.266 (FULL HD)	1min	1min	2min	3min
H.264 (UHD)	6min	6min	9min	12min
H.265 (UHD)	1min	1min	1min	2min
H.266 (UHD)	6ming	6min	9min	12min

Tabla 8. Resultados tiempo ejecución codificación H.264 vs H.265 vs H.266

6. COSTES

Para calcular el coste de este trabajo, es necesario tener en cuenta varios puntos. En primer lugar, se necesita medios físicos, es decir poseer de un ordenador portátil, con uno de gama media es más que suficiente, actualmente en el mercado tendría un coste de 800€. Seguimos con los medios digitales, como licencias, programas… aquí se necesita la licencia de Matlab que es el programa usado para realizar todas las gráficas, cuyo coste anual estándar actual de la licencia de Matlab es de 860€. Los demás programas se ha usado una versión gratuita de todos ellos.

El coste anual estándar actual de la licencia de Matlab es de 860€ y respecto a la mano de obra, serían 360 horas (lo equivalente al número de créditos del TFG) a razón de 30€ la hora, por lo que el coste total de mano de obra sería de 10800€.

Por último, la mano de obra se necesita a un ingeniero/a trabajando 360 horas (lo equivalente al número de créditos del TFG) a razón de 30€ la hora, con lo cual, el coste total de mano de obra sería de 10800€. En resumen, el coste total estimado será:

Tipo de coste	**Coste (€)**
Medios físicos	800€
Medios digitales/Licencias	860€
Mano de obra	10800€
TOTAL	12460€

Tabla 9. Estimación de costes

7. CONCLUSIONES

En esta última parte del TFG, se describen las conclusiones finales tras el análisis de los resultados, así como las competencias que se han empleado para poder llevarlo a cabo y también las competencias que se han adquirido tras su realización. Por último, también se describen posibles mejoras en el trabajo para implementar en el futuro.

7.1. CONCLUSIONES FINALES

En este TFG se ha presentado un estudio en detalle de la arquitectura del último estándar de codificación H.266, comparándolo con su predecesor, el H.265. Aunque ambos estándares muestran una arquitectura de codificación similar, el nuevo estándar presenta un particionado en bloques con tamaños más grandes, más modos angulares en predicción intra-frame, incorpora un mayor número de filtros adicionales y utiliza más variantes en el bloque de codificación transformacional.

En cuanto a los resultados, se ha realizado un estudio comparativo utilizando videos con distintas resoluciones (HD, FHD y UHD) así como diferentes niveles de *bitrate* (128 KB, 256 KB, 512 KB y 1024 KB), comprimiéndolos mediante la conocida herramienta FFmpeg, con una profundidad de color diferente a la habitual, ya que en este caso se han realizado las compresiones en 10 bits en vez de 8 bits.

Con el fin de comparar los resultados, se ha utilizado una herramienta que permite obtener diversas métricas objetivas, de las que se han utilizado varias de las más representativas (PSNR, VMAF, SSIM).

Los resultados obtenidos han sido coherentes, ya que para todas las métricas y parámetros, el nuevo estándar H.266 ha superado a su estándar predecesor, el H.265, y a uno de los más utilizados en la actualidad, el H.264.

Por otra parte, no se han observado variaciones significativas en cuanto al grado de compresión de los videos, pero sí que la codificación en H.266, requiere de un tiempo de procesado muy superior a los estándares anteriores.

7.2. COMPETENCIAS EMPLEADAS

En este informe se recopila toda la información relevante para la realización del TFG, por lo tanto, muchos de los conocimientos adquiridos durante estos años de estudio han facilitado su ejecución.

Uno de los conocimientos fundamentales empleados, sin los cuales este proyecto no habría sido posible, es el entendimiento del vídeo, así como la familiaridad con los comandos de Linux necesarios para utilizar FFmpeg, y habilidades de programación como Matlab.

Durante los años de carrera, son varias las asignaturas en las que se conoce y profundiza en estos conceptos: Gestión y Optimización de Recursos, Estándares de Comunicación de Audio y Vídeo, Equipos y Sistemas de Audio y Vídeo, Tratamiento Digital de la Imagen, Difusión de Audio y Video…

Por último, otro conocimiento aplicado ha sido el idioma, para comprender textos e interpretarlos puesto que, muchos de ellos estaban en inglés; así como para el uso de Matlab o FFmpeg. Esta competencia empleada se conocía en parte gracias a la asignatura de Idioma Moderno.

En resumen, sin todas y cada una de estas competencias que se han empleado no habría sido posible tener éxito en la realización del trabajo.

7.3. COMPETENCIAS ADQUIRIDAS

Después de completar el proyecto, se han desarrollado varias competencias como resultado de su ejecución. En primer lugar, se ha obtenido un conocimiento exhaustivo de todos los estándares, especialmente de H.266, que es menos conocido. Esta comprensión se ha logrado gracias a la fase de investigación.

Otra competencia adquirida es la capacidad mejorada para buscar información relevante y sintetizarla de manera efectiva para aplicarla en el desarrollo de un proyecto tan amplio como este.

Además, se han fortalecido y mejorado otras habilidades. Se ha profundizado considerablemente en el conocimiento del mundo audiovisual y del vídeo, así como en el dominio de las herramientas utilizadas en dicho ámbito. Por ejemplo, se ha adquirido experiencia en la resolución de problemas que pueden surgir durante la codificación de vídeos con nuevas herramientas que permiten la codificación en el nuevo estándar.

7.4. TRABAJOS FUTUROS

Respecto a las líneas de mejora o trabajos a futuro de este TFG, destacan dos posibles mejoras importantes:

- Realizar comparaciones del nuevo estándar, H.266, con otros códecs de empresas privadas como por ejemplo Google, y no solo con estándares de la ITU como hemos realizado en este trabajo.
- Aprovechar la instalación del programa que se ha utilizado para realizar las mediciones de las métricas más importantes, *MSU Video Quality Measurement Tool*, y calcular otras métricas de calidad como por ejemplo, VQM, MSAD, MSE…

8. BIBLIOGRAFÍA

[1] Bitrate, "xataka", 2023, [En línea]. Disponible: https://www.xataka.com/basics/que-es-el-bitrate-de-un-video-y-como-saberlo-en-windows-10-y-macos

[2] Resoluciones, "xataka", 2021, [En línea]. Disponible: https://www.xatakahome.com/televisores/hd-full-hd-qhd-uhd-que-significa-cada-resolucion-diferencias

[3] PSNR, "es-academic" [En línea]. Disponible en: https://es-academic.com/dic.nsf/eswiki/889352

[4] Definiciones, "Diccionario audiovisual" [En línea]. Disponible: https://www.telefonicaserviciosaudiovisuales.com/diccionario-audiovisual/

[5] B. García, L. Lopez-Fernández, F. Gortázar, M. Gallego, Practical Evaluation of VMAF Perceptual Video Quality for WebRTC Applications. Disponible en: https://e-archivo.uc3m.es/bitstream/handle/10016/32367/Practical_Electronics_2019.pdf?sequence=1&isAllowed=y

[6] Introducing VMAF percentiles for video quality measurements. en_us, "X engineering", 2023, [En línea]. Disponible: https://blog.twitter. com/engineering/en_us/topics/infrastructure/2020/introducing-vmaf-percentilesfor-video-quality-measurements

[7] J. A Tovar, Transmisión de video streaming de ultra alta definición sobre una red óptica. Disponible en: https://repository.javeriana.edu.co/bitstream/handle/10554/62046/318-attachment-1641934498.pdf?sequence=1#page=46&zoom=100,96,733

[8] Z. Wang, L. Lu, and A. Bovik, "Video quality assessment based on structural distortion measurement,",2004.

[9] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity,", 2004

[10] Definición códec y tipos de códecs, "Observatorio tecnológico", 2022 [En línea]. Disponible:http://recursostic.educacion.es/observatorio/version/v2/gl/software/software-general/388-jorge-blanco

[11] Definición ITU e ISO, "UNED", 2022 [En línea]. Recuperado de: http://ocw.innova.uned.es/mm2/tm/contenidos/pdf/tema10.pdf

[12] Estructura del codificador, "Programador clic", 2022 [En línea]. Disponible: https://programmerclick.com/article/5511676818/

[13] Tipos de codec y definiciones, "Edu.co" [En línea]. Recuperado de: http://aprendeenlinea.udea.edu.co/boa/contenidos.php/11dc9853df8fe7a0e6ea5b13554f1866/361/1/contenido/codecvideo.html

[14] MPEG-4, "Wondershare" [En línea]. Recuperado por: https://recoverit.wondershare.es/video-repair/mpeg-vs-h264.html

[15] WMV, "Techlandia", 2012, [En línea]. Disponible: https://techlandia.com/avi-contra-mpg-contra-wmv-contra-mov-info_189229/

[16] VP9, "Javier Ortiz", 2019, [En línea]. Disponible: https://javierortiz.mx/glosario/codec-de-video/codec-vp9/

[17] AV1, “IONOS”, 2021, [En línea]. Disponible: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/codec-av1/
[18] A. Diyanov, “Evalución del nuevo coficador H.266 para sistemas de streaming”, 2021.
[19] A. Filippov & V. Rufitskiy. Recent advances in intra prediction for the emerging H.266/VVC video coding standard, 0525–0530, 2019
[20] Características H.266, “ITU”, 2022, [En línea]. Available: https://www.itu.int/en/mediacentre/Pages/pr13-2020-New-Versatile-Video-coding-standard-video-compression.aspx
[21] H. Schwarz , M. Coban, M. Karczewicz, T. Chuang, F. Bossen , A. Alshin, J Lainema, C.R. Helmrich and T Wiegand ,Quantization and entropy coding in the versatile video coding (VVC) standard., DOI: 10.1109/tcsvt.2021.3072202, 2021.
[22] B. Bross, Y. Wang, Y. Ye, S.Liu,J. Chen, G.J. Sullivan, and J.R. Ohm, "Overview of the versatile video coding (VVC) standard and its applications", DOI: 10.1109/tcsvt.2021.3101953, 2021.
[23] F. Pakdaman, M.A. Adelimanesh1, M. Gabbouj2, M.R. Hashemi, "Complexity analysis of next-generation VVC encoding and decoding", DOI: 10.1109/icip40778.2020.9190983, 2020.
[24] B. Bross ,J. Chen , J.R Ohm,G.J. Sullivan and Y.K. Wang, "Developments in international video coding standardization after AVC, with an overview of versatile video coding (VVC)", 10.1109/jproc.2020.3043399, 2021.
[25] B. Zhu, S. Liu, Y. Liu, Y. Luo, J. Ye, H. Xu, Y. Huang, H. Jiao, X. Xu, X. Zhang, C. Gu,"A software decoder implementation for H.266/VVC video coding standard",2020.
[26] CABAC, “hmong”, [En línea]. Disponible:https://hmong.es/wiki/Context-adaptive_binary_arithmetic_coding#:~:text=La%20codificaci%C3%B3n%20aritm%C3%A9tica%20binaria%20adaptativa,de%20alta%20eficiencia%20(HEVC)
[27] VCC, "hhi,fraunhofer", [En línea]. Disponible: https://www.hhi.fraunhofer.de/en/departments/vca/technologies-and-solutions/h266-vvc/vvc-overview.html
[28] J. Brandenburg, A. Wieckowski, T. Hinz, I. Zupancic, B. Bross (s/f).,VVenC Fraunhofer Versatile Video Encoder., Disponible en: https://www.hhi.fraunhofer.de/fileadmin/Departments/VCA/MC/VVC/vvenc-v1.3.1-v1.pdf
[29] H.265/H.266, "winxdvd", 2020, [En línea]. Disponible:https://www.winxdvd.com/video-transcoder/h266-vvc-vs-h265-hevc.htm

ANEXOS

A.1. INSTALACIÓN FFMPEG

Para su instalación se recomienda seguir los siguientes pasos:

- Ir a la web oficial de FFmpeg (https://ffmpeg.org/download.html) y descargar la última versión disponible, eligiendo también la arquitectura del procesador e iniciar la descarga.
- Una vez descargado el archivo se debe extraer todo su contenido.
- Se recomiendo renombrar el archivo extraído para evitar confusiones, se le puede poner por ejemplo el nombre FFmpeg.
- Ahora se mueve esta carpeta recién creada a la unidad de instalación de Windows, es decir, copiamos la carpeta y abrimos la unidad C del equipo y lo pegaremos ahí.
- A continuación, en nuestro dispositivo accedemos a propiedades del sistema, lo podemos encontrar dentro del explorador de archivos. Aquí elegimos la opción este PC y propiedades.
- Dentro de propiedades, encontramos opciones avanzadas del sistema y variables de entorno.

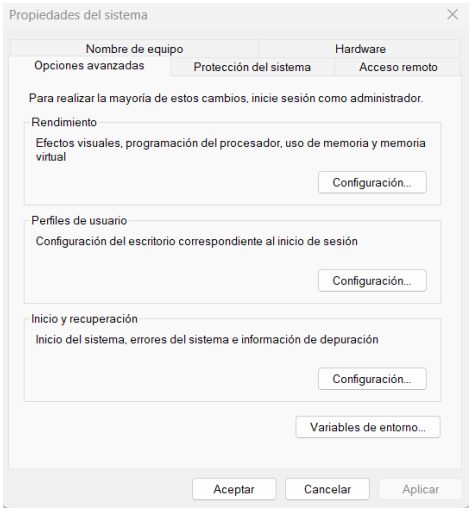


Figura 23. Instalación FFmpeg – Propiedades del sistema

- Una vez dentro de las variables de entorno, debemos editarlas, en variables de usuario donde añadimos una nueva ruta que en este caso será C:\ffmpeg\bin, lo haremos pulsando editar.

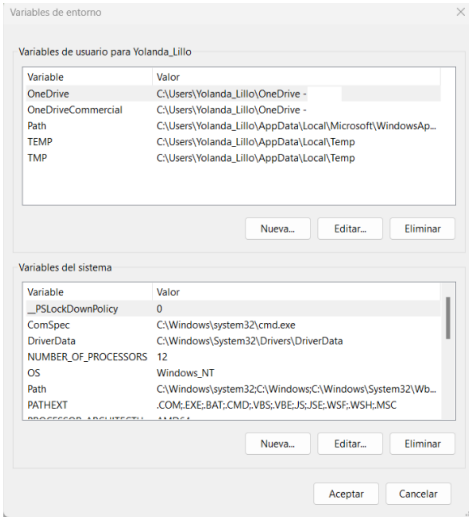


Figura 24. Instalación FFmpeg – Variables de entorno

- Por último, se guardan estos cambios.
- Ahora ya está instalado FFmpeg y configurado las variables de entorno apropiadas. Para asegurarnos de la correcta instalación y funcionamiento, abre el símbolo del sistema y ejecuta este comando para ver la versión que tenemos instalada: *ffmpeg -version.*

Aunque también se puede obtener directamente el ejecutable y usarlo desde un terminal.

A.2. INSTALACIÓN VVCEASY

VVCEasy es una herramienta de FFmpeg codificar VVC, es simple y fácil además, que nos sirve para codificar y decodificar. Lo que nos ofrece son comandos sencillos de ffpemg con VVC Tools, VLC o266player, VVDEC Web Player, Bitmovin VVDec Player, libvvdec y más.

Para su instalación, se debe ir a su página de github, que se añade a continuación:

- https://github.com/MartinEesmaa/VVCEasy

Dentro de esta página, debemos dirigirnos hasta “Release” y descargar la última versión acorde con nuestro sistema operativo.

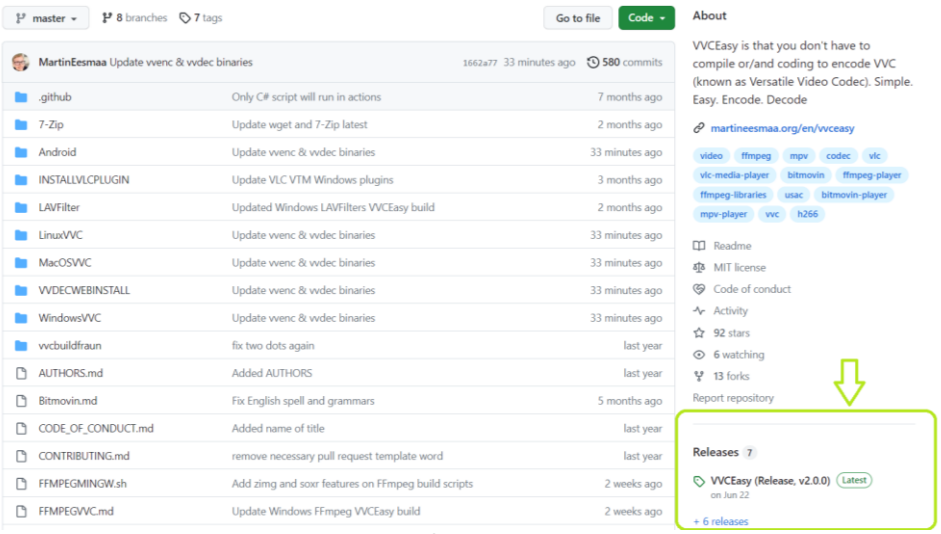


Figura 25. Página github VVCEasy

Una vez descargado, se usa el ejecutable, *ffmpeg_vcc.exe*, desde la consola de comandos para realizar nuestras pruebas, también usaremos *ffplay_vvc.exe*, para ver los videos resultantes.

A.3. INSTALACIÓN *MSU Video Quality Measurement Tool*

Es un programa muy útil y sencillo para obtener métricas con diferentes vídeos, para su instalación, primero se debe acceder a la siguiente url:

- https://compression.ru/video/quality_measure/video_measurement_tool.html

Una vez aquí, se hace click en obtener la versión gratuita, se selecciona Windows como sistema operativo y se añade nombre, email y se debe pulsar descargar. De este modo, se obtiene la última versión más actualizada, es decir, en este caso la versión 14.1. Tras descargar el ejecutable solo se deben seguir los pasos indicados, pulsar siguiente hasta instalar.

En la siguiente imagen se puede ver la interfaz del programa.

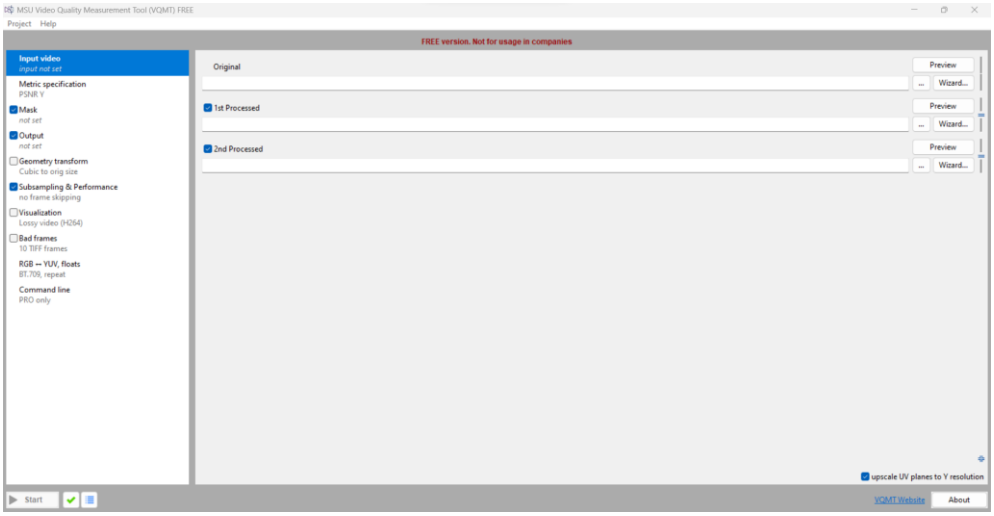


Figura 26. Interfaz programa

Una vez está instalado el programa, se puede empezar a calcular las métricas que necesitemos.