ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TESIS PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: DESARROLLO DE UN DECODER DE VIDEO H.264/TS UTILIZANDO LA PLATAFORMA DE MATLAB®

AUTOR: TENORIO MOYA, EDISON FERNANDO

DIRECTOR: DR. OLMEDO, GONZALO

CODIRECTOR: ING. BERNAL, PAÚL

SANGOLQUÍ

2015

CERTIFICADO DE TUTORÍA

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE
DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA
CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICADO

Dr. Gonzalo Olmedo.
Ing. Paúl Bernal.

CERTIFICAN

Que el trabajo titulado "DESARROLLO DE UN DECODER DE VIDEO H.264/TS UTILIZANDO LA PLATAFORMA DE MATLAB®"**,** realizado por Edison Fernando Tenorio Moya, ha sido guiado y revisado periódicamente y cumple normas estatutarias establecidas por la ESPE, en el Reglamento de Estudiantes de la Universidad de las Fuerzas Armadas ESPE.

Debido a que se trata de un trabajo de investigación recomiendan su publicación.

Sangolquí, 29 de Abril de 2015

Dr. Gonzalo Fernando Olmedo Cifuentes
ID: L00007104
DIRECTOR

Ing. Carlos Paúl Bernal Oñate
ID: L00007564
CODIRECTOR

AUTORÍA DE RESPONSABILIDAD

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE
DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA
CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Edison Fernando Tenorio Moya

DECLARO QUE:

El proyecto de grado denominado "DESARROLLO DE UN DECODER DE VIDEO H.264/TS UTILIZANDO LA PLATAFORMA DE MATLAB®", ha sido desarrollado con base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas cuyas fuentes se incorporan en las referencias bibliográficas.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, 29 de Abril de 2015

Edison Fernando Tenorio Moya

AUTORIZACIÓN DE PUBLICACIÓN

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE
DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA
CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Edison Fernando Tenorio Moya

Autorizo a la Universidad de las Fuerzas Armadas – ESPE la publicación, en la biblioteca virtual de la Institución del trabajo "DESARROLLO DE UN DECODER DE VIDEO H.264/TS UTILIZANDO LA PLATAFORMA DE MATLAB®", cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría.

Sangolquí, 29 de Abril de 2015

Edison Fernando Tenorio Moya

DEDICATORIA

A las personas más importantes de mi vida mis padres Fernando y Margarita por enseñarme que las cosas en la vida se las consigue con esfuerzo, perseverancia, trabajo y dedicación, por acompañarme en cada una de las etapas de mi vida sabiendo guiarme, apoyarme, pero sobre todo, por brindarme su amor, a mi hermana Michelle por ser más que una hermana por ofrecerme su apoyo y ánimo incondicional.

A mis amigos y compañeros quienes han formado parte de mi vida todos estos años en los buenos y sobre todo en los momentos que parecían complicarse pero su compañía y apoyo fue esencial, gracias a todos he logrado este objetivo.

Edison Fernando Tenorio Moya

AGRADECIMIENTO

A mis padres Fernando y Margarita a quienes les debo todo lo que soy, por todo su apoyo y comprensión que me permitieron culminar mi carrera profesional, gracias por existir y ser mí motivación personal. A mi hermana Michelle por aguantarme, cuidarme y siempre estar ahí de una u otra forma.

A mis compañeros y amigos de universidad por la suerte que he tenido de encontrar unos tan buenos que me han apoyado y corregido siempre, les agradezco por lo que les deseo lo mejor en su vida profesional.

A mi Director y Co-Director de Tesis Dr. Gonzalo Olmedo e Ing. Paúl Bernal a quienes les agradezco enormemente por su confianza, respaldo y guía en este proyecto de grado. Mi gratitud y amistad eterna a ustedes por toda su gran ayuda.

Finalmente debo agradecer a todas las personas que han sido parte de mi vida, ya que cada una ha aportado algo para ser quien soy ahora.

Edison Fernando Tenorio Moya

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En el presente proyecto se desarrolló el decoder de video H.264/TS utilizando la plataforma de MATLAB®, donde se pudo conocer el tratamiento eficiente que da el códec a cada bloque de una secuencia de imágenes obteniendo así una compresión y calidad adecuada para su empaquetamiento. Para ello se creó una interfaz GUI que llama a funciones como el VideoReader y VideoWriter que se encargan de la compresión, estas a su vez trabajan con un conjunto de scripts internos específicos para el procesamiento de video. En la interfaz se visualiza el resultado tanto de la compresión H.264 como de su empaquetado TS, aquí se ingresa un parámetro que define la calidad del video (Q) este valor puede variar entre 0 y 100, pero al realizar una comparación se concluye que el valor de 50 cumple con una compresión de casi a la mitad del tamaño que se tiene a la entrada del aplicativo, lo cual es una de las principales características del estándar H.264. Adicional gracias a la herramienta FFmpeg desarrollada en Linux se realizó el empaquetamiento donde se añadió a cada paquete de 184 bytes una cabecera de 4 bytes. Mediante este trabajo se ha comprobado que una de las principales ventajas de H.264/AVC es la capacidad de predicción de imágenes y su acoplamiento eficiente con el protocolo de transporte multimedia TS. Referente a lo expuesto se recomienda hacer uso de este decoder para aplicarlo bajo un ambiente de DTV, ya que ofrece características superiores a estándares predecesores.

Palabras Claves:

- **H.264**
- **PREDICCIÓN**
- **TS**
- **EMPAQUETAMIENTO**
- **DECODER**

ABSTRACT

In this project the video decoder H.264/TS was developed using MATLAB platform, where it was known efficient treatment that gives the codec to each block of an image sequence compression and obtaining adequate quality for packaging. To do a GUI that calls functions as VideoReader and VideoWriter that handle compression, these in turn work with a set of specific internal scripts for processing video was created. At the interface the result of both the H.264 compression and its packaging TS is displayed, here a parameter defining the video quality (Q) this value can vary between 0 and 100 is entered, but a comparison is concluded 50 that the value complies with a compression of almost half the size that has application to the input of which is one of the main characteristics of H.264. Further thanks to the FFmpeg Linux tool developed in packaging where added to each 184-byte packet header of 4 bytes is performed. Through this work we have found that one of the main advantages of H.264/AVC is the predictability of images and their efficient coupling with multimedia transport protocol TS. Regarding the above is recommended to use this decoder to apply under an atmosphere of DTV, offering standard features than predecessors.

Key Words:

- **H.264**
- **PREDICTION**
- **TS**
- **PACKAGING**
- **DECODER**

GLOSARIO DE TÉRMINOS

AAC	Advanced Audio Coding
AVC	Advanced Video Coding
AVI	Audio Video Interleave
BIFS	Binary Format for Scenes
CABAC	Context Adaptive Binary Arithmetic Coding
CAVLC	Context Adaptive Variable Length Coding
CBR	Constant Bit Rate
CCIR	International Radio Consultative Committee
CIF	Common Intermediate Format
CR	Compression Ratio
DCT	Discrete Cosine Transform
DM	Delta Modulation
DPCM	Differential Pulse Code Modulation
DVB	Digital Video Broadcasting
DVD	Digital Versatile Disc
ES	Elementary Stream
FIR	Finite Impulse Response
FME	Fractional Motion Estimation
FS	Full Search
GOP	Group of Pictures
GUIDE	Graphical User Interface Development Enviroment
HDTV	High Definition Television
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union (Telecommunication Standardization Sector)
IME	Integer Motion Estimation
ISDB-TB	Integrated Service Digital Broadcasting - Terrestrial Built in
ISDN	Integrated Services for Digital Network

ISM	Interactive Storage Media
JPEG	JoinPhotographic Experts Group
JVT	Joint Video Team
LTE	Long Term Evolution
MAE	Mean Absolute Error
MB	Macro block
MC	Motion Compensation
ME	Motion Estimation
MOS	Mean Opinion Score
MPEG	Moving Picture Experts Group
MSE	Mean Squared Error
MV	Motion Vector
NTSC	National Television System Committee
PAL	Phase Alternating Line
PES	Packetized Elementary Stream
PSI/SI	Program Specific Information / Service Information
QCIF	Quarter Common Intermediate Format
RLE	Run Length Encoding
RTP	Real Time Transport Protocol
SMPTE	Society of Motion Picture an Television Engineers
SAD	Sum of Absolute Diferences
SDTV	Standard Definition Television
TDT	Television Digital Terrestre
TS	Transport Stream
TVD	Television Digital
VBR	Variable Bitrate
VCEG	Video Coding Experts Group
VCR	Video Cassette Recorder
VLC	Variable Length Coding
VQEG	Video quality Experts Group
VRML	Virtual Reality Modelling Language

CAPÍTULO 1

INTRODUCCIÓN

1.1. Antecedentes

En las últimas décadas, la constante evolución a nivel de tecnología han incentivado al ser humano busque nuevas alternativas para ir a la par de estos avances, es así como las técnicas de compresión de vídeo se han convertido en una parte integral de nuestra forma de crear, comunicar y consumir información visual. La comunicación de vídeo digital se puede encontrar hoy en día en muchas aplicaciones, tales como: servicios de radiodifusión a través de canales vía satélite y terrestres, almacenamiento de vídeo digital, cables y servicios de conversación inalámbricos entre otros [1].

Actualmente existe una amplia gama de normas para la codificación de video, las que varían en sus características de acuerdo al tipo de aplicación sean almacenamiento, videoconferencias, indexado, entre otras [2]. A finales de 2001 los grupos de expertos en codificación de video de las dos organizaciones de estandarización de contenido multimedia más importantes del mundo, el *Moving Picture Experts Group* (MPEG) de la *International Organization for Standardization e International Electrotechnical Commission* (ISO/IEC) y el *Video Coding Experts Group* (VCEG) de la *International Organization for Standardization* (ITU-T), enfocaron sus esfuerzos para crear conjuntamente un nuevo estándar de codificación de video, formando para ello el Joint Video Team (JVT).

El estándar que nació gracias a la colaboración entre las dos organizaciones se lo denomina MPEG-4 Parte 10 *Advanced Video Coding* (AVC) por la ISO/IEC y H.264 por la ITU-T, por lo que este es conocido como H.264/AVC [3].

En los últimos años H.264/AVC ha sido el codificador de video más utilizado por la industria audiovisual debido a que su principal característica radica en proporcionar una buena calidad reduciendo notablemente el flujo de bits de salida de

la secuencia de vídeo codificada. Además, pretende conseguir esto sin incrementar la complejidad, evitando así convertir el estándar en algo imposible de implementar [4].

El códec H.264 representa una extensión de los formatos predecesores (MPEG-1, MPEG-2, MPEG-4/Parte 2, H.261, H.263) al mantener la base para la codificación, esto supuso un avance significativo en comparación con los estándares mencionados puesto que es más robusto a errores de transmisión y en términos de eficiencia a la hora de codificar consigue aumentar la compresión hasta un 200%, a la vez que se aumenta de forma sustancial la calidad y definición de imagen [5].

El estándar de video H.264/AVC en relación MPEG-4 básico también incluye el concepto de perfil y/o nivel, esto permite definir conjuntos específicos de capacidades que pueden ser implementadas para cumplir con objetivos particulares [6]; dichos perfiles se los maneja a nivel superior lo que la convierte en un tipo de codificación más compleja la cual requiere de más velocidad de procesamiento y memoria, parámetros que no se consideraban en MPEG-4 Parte 2.

1.2. Justificación e Importancia

En la actualidad el desarrollo de aplicaciones como las relacionadas con sistemas de audio y video en tiempo real, han hecho de los sistemas de codificación y compresión de datos un aspecto esencial de la tecnología digital de comunicaciones y almacenamiento de datos puesto que en casi todos los sistemas emplean formatos comprimidos, sin pérdidas significativas en la calidad; por lo que el video, audio y datos en su forma de almacenamiento y transmisión hoy en día se vuelve un tanto problemático, en dichos términos se hace inevitable comprimir la información [7].

H.264/AVC presenta características fundamentales, como el uso de bloques de tamaño variable, lo que permite mejor adaptación a los movimientos mediante la división de la imagen en macro bloques, además presenta múltiples cuadros de referencia, lo que mejora la capacidad de compresión debido a las técnicas de codificación predictiva que maneja las mismas, que buscan eliminar la redundancia temporal de la secuencia de video; proporcionando mayor flexibilidad destinada a satisfacer el mayor número de aplicaciones posibles, teniendo sin embargo un elevado coste computacional [4].

En sistemas de video *streaming* es necesario tomar en cuenta el tamaño de la cabecera del paquete de video, teniendo en consideración que en este documento se expone la codificación H.264/AVC. Teóricamente MPEG-4 utiliza métodos para el transporte del *streaming*, uno de ellos es el método que utiliza el procedimiento basado en MPEG-2 TS, esto es debido a que los usuarios están habituados a usar el MPEG-2 y sus técnicas de transporte. No está demás decir, que este método es bastante bueno y sobretodo muy usado. El procedimiento de empaquetamiento MPEG2-TS utiliza paquetes TS (TS *packets*), estos paquetes TS son los transportadores del *stream* (*streaming transport*), los cuales constan de dos partes, cabecera y carga útil. La carga útil está constituida por fragmentos del paquete PES, este paquete PES contiene el video original antes de ser particionado, para luego ser transportado por varios paquetes TS, así se obtienen paquetes TS de menor tamaño por los cuales se transportan partes del video, audio e información, sin olvidar que mientras más grande sea el paquete PES van a existir más paquetes TS y por lo cual mayor cantidad de cabeceras TS, incrementando el tamaño de la cabecera del paquete [8].

En este punto se puede evidenciar el problema principal que tiene la utilización del video digital, ya que la cantidad de bits que aparecen al realizar la codificación y su empaquetamiento, es decir, la cantidad de datos con los que se maneja son muy grande y no se debe olvidar que la memoria de dispositivos de almacenamiento, el ancho de banda del canal de transmisión no son infinitos, por lo que, se ve la necesidad de reducir la cantidad de datos necesarios para reproducir video, ya que ahorra espacio de almacenamiento, aumenta velocidad de acceso y es la única manera de lograr video de movimiento en computadoras digitales. Es así como nacen las técnicas de compresión, las mismas que son importantes explorarlas, conocerlas para poder implementarlas en nuestra vida diaria, por ello este Proyecto de Titulación tiene el fin de incrementar el uso de H.264/TS una de las nuevas técnicas de compresión y de empaquetamiento, creando una interfaz que permita que el usuario de una manera simple utilice sus ventajas.

Una vez comprimida la información se nos vuelve más sencillo las tareas de almacenarla o transmitirla. Así pues, la compresión, es un factor económico ya que

reduce el costo de la transmisión o distribución de señales. Dado un medio de transmisión concreto con un determinado coste de amortización, cuanto menor sea el ancho de banda de los canales a transmitir, más canales estarán disponibles y más económico resultará cada canal. Además en determinadas aplicaciones, el factor de compresión va a venir impuesto por la velocidad binaria del canal de transmisión disponible.

1.3. Alcance del proyecto

Realizando una investigación e implementación del presente proyecto con la herramienta de desarrollo MATLAB® la cual cumple con la norma H.264/AVC se llegará a determinar el diseño de un aplicativo que permitirá realizar la codificación de video y su empaquetamiento TS.

Se realizará el análisis de la generación de los flujos de señal MPEG-2 TS, el mismo que consta de dos grandes bloques, denominados “Capa de Compresión” y “Capa de Sistema”. En la “Capa de Compresión” se realiza las operaciones propiamente dichas de codificación en el estándar H.264/AVC, recurriendo a los procedimientos generales de compresión de datos y aprovechando recursos como la redundancia espacial (áreas uniformes) y temporal (imágenes sucesivas), la correlación entre puntos cercanos y la menor sensibilidad del ojo humano a los detalles finos de las imágenes fijas. En la “Capa de Sistema” se realiza las operaciones que conducen a la obtención de los flujos de señal MPEG-2 denominado como empaquetamiento TS el cual consistentes en la organización en “paquetes” de los datos comprimidos para el posterior multiplexado de todas las señales asociadas al programa (vídeo, audio, datos) [9].

Se incluirá además una breve explicación de las características y atributos de la televisión digital, los modelos de la televisión digital y los elementos que intervienen en la difusión de la TV Digital interactiva, basados en el estándar brasileño ISDB-TB adoptado por Ecuador, el cual emplea tecnologías avanzadas de compresión de video H.264 como también la importancia de la utilización de flujos de señales TS.

Referente al aplicativo en software se podrá contar con una interfaz completamente didáctica, práctica y amigable para cualquier tipo de usuario, la cual posee una opción que permite seleccionar el tipo de video a ser comprimido. A continuación se establece el evento de ejecución para la compresión de video y almacena en un directorio el archivo nuevo. Posteriormente el proceso de empaquetamiento TS el cual se realiza el proceso de ejecución permitiendo almacenar el video empaquetado en un directorio nuevo. Una vez establecida la etapa de compresión y de empaquetamiento se podrá realizar la verificación del video mediante el software *VLC Media Player* donde se puede obtener la visualización de forma detallada de las características del video como: ver el códec que usa, su resolución, *frames* por segundo, tipo de video. Finalmente realizado el análisis del decoder de video H.264.TS se evaluará el rendimiento detallando los siguientes parámetros: tamaño del archivo, duración del archivo, entre otros. Todos estos aspectos cabe resaltar son necesarios para utilizarlos en aplicativos comerciales como VillageFlow [10] el cual es la última plataforma de software para generación, operación, procesamiento y control de señales en TDT, abierta y flexible para el desarrollo de sistemas de tratamiento de *Transport Stream*.

1.4. Objetivos

1.4.1. Objetivo General

Diseñar un decoder de video H.264/TS implementado en la plataforma de desarrollo MATLAB®.

1.4.2. Objetivos Específicos

- Estudiar el estándar de codificación de video H.264 equivalente al MPEG-4 establecida por la norma ISO 14496, detalladamente el perfil 10 (*Advanced Video Coding*).
- Investigar MPEG-2 TS uno de los métodos para el transporte de *streaming*, esencialmente la estructura y características de la trama TS.
- Describir el proceso y técnicas de compresión para sistemas de distribución de señales de televisión por TDT.

- Crear una interfaz gráfica para visualizar de forma detallada los formatos de video tanto para la codificación H.264 como para el empaquetado TS a través de un GUI, la cual incorpore el diseño del decoder utilizando el software MATLAB® tratando que la misma sea de fines didácticos, cumpliendo con ser práctico y que su uso sea sencillo, amigable para cualquier tipo de usuario.
- Analizar los resultados obtenidos mediante la comparación de archivos comprimidos con archivos originales; verificando la calidad del video, tamaño del archivo, lo cual demuestre la funcionalidad del decoder de video H.264/TS.

CAPÍTULO 2

FUNDAMENTO TEÓRICO

2.1. Compresión de la información

Hoy en día es impensable la información multimedia en formato sin compresión. Puesto que en la actualidad son más las aplicaciones que requieren de un amplio rango de calidad y funcionamiento de acuerdo a los diversos requerimientos y necesidades de los usuarios. Ventajosamente, durante las últimas décadas un gran número de investigaciones han conducido a varias técnicas y algoritmos de compresión que han hecho factible la transmisión multimedia.

En su mayoría las técnicas de compresión actuales, compiten entre sí o son complementarias. Los métodos de compresión recientes se basan en lograr: calidad en los datos codificados y posteriormente decodificados, tasas de compresión tan alta como se consiga y una apropiada relación costo-beneficio.

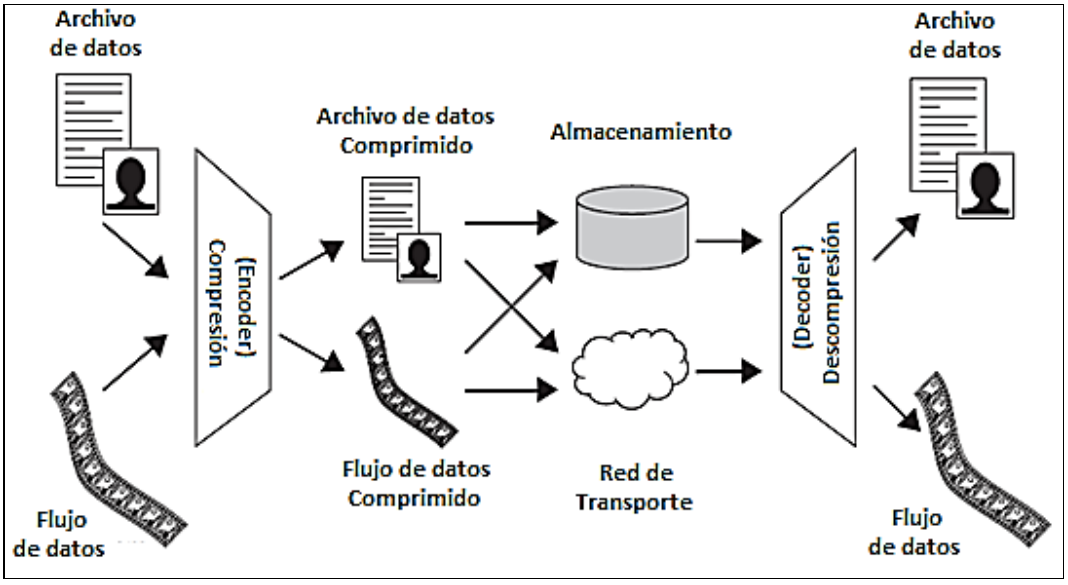

Figura 2.1. Diagrama simplificado de un sistema de compresión [11]

La Figura 2.1 muestra un diagrama de bloques simplificado de un sistema de compresión. La entrada puede ser un archivo de computadora, tal como un documento, una imagen y / o una grabación de vídeo / audio. O la entrada puede ser un continuo flujo de datos, tales como un video digital o señal de audio. De cualquier manera, esta información se introduce en el bloque de compresión. La salida del

bloque de compresión es un segundo archivo de datos o flujo de datos que contiene menos bits que el archivo de entrada. Esta salida puede ser almacenada o transmitida a través de una red. Antes de que el archivo de datos pueda ser utilizado, debe ser restaurado a su tamaño original, a través del bloque de descompresión.

Las preferencias constantes hacia la estandarización de formatos de compresión, han hecho viables una compatibilidad casi universal entre los diversos sistemas de comunicación. Debido a la existencia de diferentes tipos de archivos con los cuales trabajamos cotidianamente y a veces, de manera simultánea; han hecho que los algoritmos de compresión se desarrollen de acuerdo al tipo de archivo y a determinadas relaciones de compresión requeridas. Es así que para el manejo de archivos de texto, audio, imágenes y video existen sus algoritmos correspondientes.

En la Figura 2.2 se aprecia la evolución de los estándares de codificación específicamente los destinados para video.

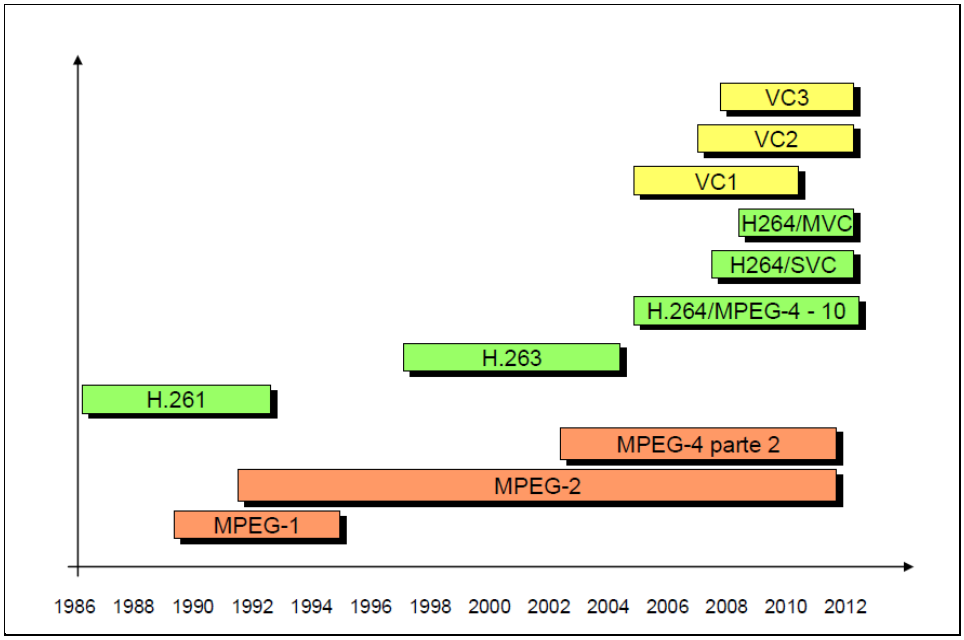

Figura 2.2. Evolución de los estándares de codificación de video [12]

Dentro de la información que transmiten los datos puede ser de tres tipos:

- **Redundante.-** es aquella información repetitiva o predecible.
- **Irrelevante.-** se denomina a aquella información la cual no se puede apreciar y cuando se la elimina no afecta al contenido del mensaje.

- **Básica.-** se la denomina a la información relevante, la misma que no es ni redundante ni irrelevante. La cual debe ser transmitida para que se pueda reconstruir la señal

Los métodos de compresión dependen básicamente del tipo de datos que se van a comprimir, puesto que existe una notable diferencia cuando se comprime una imagen en relación a un archivo de audio debido principalmente a la cantidad de datos con los cuales se manejan. Cuando se habla de compresión, es útil mencionar algunos términos como los siguientes:

- **Redundancia.-** son aquellos datos que son repetitivos o previsibles.
- **Redundancia Espacial.-** esta correlación puede ser percibida en el dominio espacial. Por ejemplo existe redundancia cuando un conjunto de píxeles vecinos poseen valores semejantes, para lo cual los formatos de compresión utilizan la codificación intra cuadro.
- **Redundancia Temporal.-** esta correlación puede ser percibida en el domino temporal. Por ejemplo entre los cuadros que se encuentran próximos entre sí en el tiempo, se puede llegar a conseguir altas tasas de compresión si se hace una exploración eficiente de la redundancia, al cual se le denomina codificación inter cuadros.
- **Redundancia Entrópica.-** está relacionada con las probabilidades de ocurrencia de los símbolos de video codificados, es decir, mientras mayor sea la cantidad de información nueva, disminuirá la probabilidad de aumento de la aparición de un símbolo.
- **Entropía.-** es la información esencial la misma que se define como la diferencia entre la cantidad de datos de un mensaje y su redundancia.
- **Relación de compresión.-** se la conoce también como el factor de compresión o radio de compresión, mide cuanto comprime un algoritmo de compresión. La CR en forma de relación n:1 viene dado por la siguiente expresión:

$$CR = \frac{\text{número de bits sin compresión}}{\text{número de bits con compresión}} \tag{2.1}$$

El número de bits sin compresión representa el tamaño original del archivo (TO) mientras que el número de bits con compresión representa el tamaño comprimido (TC). Por esta razón, expresado como porcentaje la CR viene dado por:

$$CR = \frac{\text{TO} - \text{TC}}{\text{TO}} * 100\ [\%] \tag{2.2}$$

2.2. Compresión de la información con pérdida *LOSSY*

La compresión con pérdida de información está diseñada para permanentemente eliminar la información menos importante [11]. Para ello introduce medidas de distorsión perceptiva y limita umbrales tolerables, puesto que es común que el observador humano no aprecia las diferencias. Al momento de descomprimir la información, no se recupera exactamente la original, ya que en dicho proceso se pierde cierta información la misma que no se considera como fundamental.

La pérdida de datos de la compresión *lossy* se lo realiza principalmente para reducir el tamaño del archivo, pero se ve compensado en su alta relación de compresión. Esto se debe principalmente a que elimina información para lograr la mejor relación de compresión posible mientras mantiene un resultado que es lo más cercano posible a los datos originales.

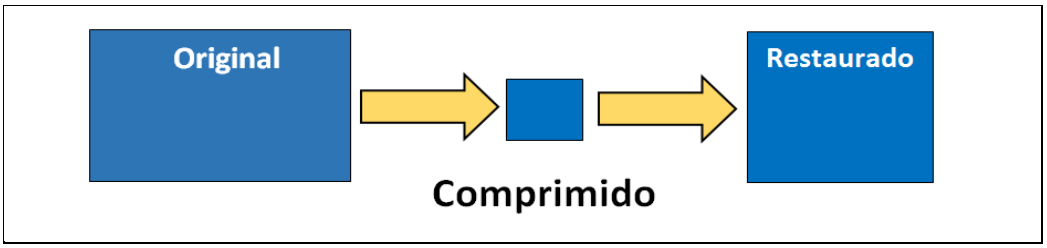

Figura 2.3. Compresión *Lossy* [9]

En la Figura 2.3 se puede apreciar como en la compresión *lossy* existe pérdida de la información la cual genera que la imagen original no pueda ser recuperada en su totalidad.

Para video los formatos: MPEG-1, MPEG-2 y MPEG-4 son los más conocidos.

2.2.1. Codificación Diferencial

Se fundamenta en la reciprocidad existente entre muestras sucesivas de la fuente, basándose en las diferencias entre dos símbolos consecutivos, no se codifica la muestra si no la diferencia entre muestras. Ciertos tipos de datos tienen la propiedad de similitud entre símbolos consecutivos: señal de audio, video, imágenes, entre otros. Esto permite codificar con pocos bits las diferencias; dentro de este tipo de codificadores se tiene:

- DPCM
- DM
- Codificación Predictiva

2.2.2. Codificación por Transformada

Se basa en la codificación por bloques. La señal de entrada se transforma a un domino diferente, se traslada del domino del tiempo al dominio de la transformada y se codifican los coeficientes de la transformación, se evalúa el contenido energético de los coeficientes de la transformada y se trasmiten únicamente aquellos coeficientes codificados de mayor aporte energético. En el receptor, el decodificador calcula la transformada inversa para obtener la señal original reconstruida. La codificación por transforma es utilizada para la codificación de señales de banda ancha de imagen; entre las transformadas de mayor uso se tiene:

- Fourier
- Coseno
- *Wavelet*

2.2.3. Cuantización Vectorial

Establece que en la operación sobre bloques de datos, en lugar de hacerlo sobre datos aislados, dicho bloque de datos se lo puede denominar como un vector (siendo N la dimensión de los vectores). La cuantización vectorial maneja una tabla (*code book*), donde se encuentran códigos conocidos como registros de códigos (RC),los mismos que contiene los posibles vectores a la salida de un cuantificador; los vectores del registro de códigos se los conoce como vector de código y su respectivo

índice en la tabla es la palabra código. Para lograr disminuir el error entre la señal cuantificada y su original se tiene que definir el número de bits del registro de códigos, la dimensión de los vectores de entrada y los valores de los vectores de códigos. El proceso de cuantificar un vector de entrada consiste en determinar su vecino más cercano dentro del registro de códigos y escoger el de menor distancia [12].

2.3. Compresión de la información sin pérdida *LOSSLESS*

La compresión sin pérdida de información nos permite recuperar exactamente los datos originales en su totalidad después de aplicar sobre los datos comprimidos un algoritmo compatible de descompresión.

Para ello utiliza un bloque de compresión sin pérdidas el cual está diseñado para eliminar cualquier redundancia en la señal de la fuente, pero para hacerlo de una manera tal que el bloque de descompresión pueda recrear completamente la señal original [11].

Con este procedimiento se logran índices de compresión muy bajos (cerca de 2:1); en relación a la compresión con pérdida, debido a que dicha compresión se caracteriza por tener una tasa de compresión limitada por la entropía de la señal original, es decir, existe un límite teórico de compresión para los compresores sin pérdida.

En la Figura 2.4 se puede apreciar como en la compresión *Lossless* luego de realizar su proceso se recupera en su totalidad los datos originales.

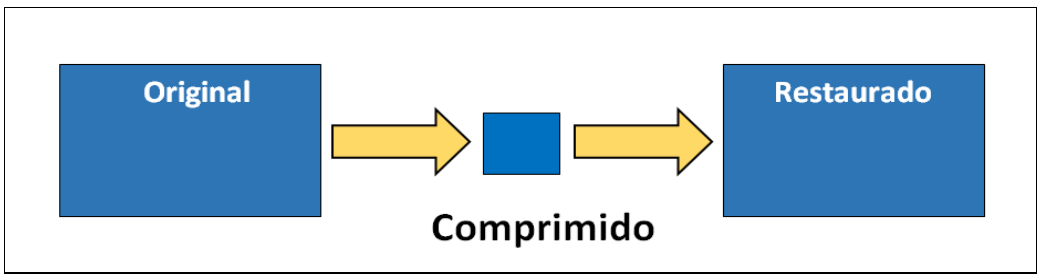

Figura 2.4. Compresión *Lossless* [9]

La compresión sin pérdida de información se usa en aplicaciones donde no es tolerable ninguna diferencia entre los datos originales y comprimidos: textos,

programas, imágenes críticas (radiografías, teledetección). Entre algunos ejemplos se destacan para imágenes: PNG, RLE; referente a video es muy raro puesto que solo suelen utilizarlos para captura.

2.3.1. Codificación Estadística

Utiliza la información de las probabilidades de los mensajes de la fuente para construir una codificación y se fundamenta en establecer códigos cortos a los datos que aparecen más frecuentemente y establecer códigos más largos a los datos más infrecuentes. De esta manera, el número promedio de bits por dato codificado es menor. También se caracterizan por utilizar las propiedades estadísticas de la fuente para mejorar la codificación, dentro de los compresores estadísticos se pueden destacar:

- Shanon-Fano
- Predictivos
- Aritméticos

2.3.2. Codificación de Huffman

Asigna un *variable length coding* (VLC) a cada símbolo basado en la probabilidad de ocurrencia de diferentes símbolos el cual nos ofrece altos radios de compresión. De acuerdo con el esquema original propuesto por Huffman en 1952, es necesario para calcular la probabilidad de ocurrencia de cada símbolo y para construir un conjunto de palabras de código de longitud variable, es decir, es una técnica que establece códigos de bits más cortos a los datos que tienen mayor frecuencia de aparición y códigos más largos a los que aparecen con menos regularidad [13].

2.3.3. Basados en Diccionarios

Se basan en construir un diccionario tomando como referencia la entrada procesada hasta ese momento, dicho diccionario contiene cadenas de mensajes identificados por un índice. Algunas cadenas de símbolos se repiten con frecuencia y en resumen no requiere un conocimiento de la probabilidad de aparición de cada símbolo. Las aplicaciones donde se emplean dichos compresores son en aquellas

donde no sea posible conocer las probabilidades de los símbolos. Dentro de los algoritmos basados en el uso del diccionario se encuentran: RLE, LZW, LZ77 y Lempel Ziv. En la Tabla 2.1 se puede diferenciar en un cuadro comparativo los diferentes métodos de compresión.

Tabla 2.1. Algoritmos de compresión [9]

Algoritmo Características	LZW	Huffman	RLE	Aritmética	Gzip	Basados DCT	Basados Wavelets
Memoria	Baja	Alta	Muy baja	Media	Baja	Media	Baja
Tasa compresión	Media	Media	Baja Media	Baja Media	Media	Media Alta	Alta
Operaciones complejas	No	Construcción de árbol	No	Punto flotante	No	Algunos casos	Algunos casos
Codificación de regiones de interés (ROI)	NA	NA	NA	NA	NA	NA	Si
Manejo errores	NA	NA	NA	NA	Si	NA	Si
Con pérdida de información	NA	NA	NA	NA	NA	Si	Si
Sin pérdida de información	Si	Si	Si	Si	Si	Si	Si

2.4. Técnicas de compresión

Las técnicas de compresión de video se fundamentan en una serie de algoritmos que consisten en reducir la cantidad de información sin que el resultado se vea afectado, es decir en esencia representa la reducción de algo.

El principal objetivo de cualquier esquema de compresión es describir la misma información con un conjunto de datos de menor magnitud; si la compresión es eficiente, el resultado ocupara menor espacio que los símbolos originales. Para ello recurren a procedimientos de compresión de datos, valiéndose además la redundancia espacial de una imagen (áreas uniformes), la correlación entre puntos cercanos y la menor sensibilidad del ojo a los detalles finos de las imágenes fijas (JPEG) y para imágenes animadas (MPEG) se saca provecho también de la redundancia temporal

entre imágenes sucesivas. Algunas de las técnicas de compresión se detallan a continuación.

2.4.1. Compresión Física y Lógica

La compresión física actúa directamente sobre los datos; por ende es cuestión de almacenar los datos repetidos de un patrón de bits a otro. En cambio, la compresión lógica se efectúa por razonamiento lógico al sustituir esta información por información equivalente.

2.4.2. Compresión Simétrica y Asimétrica

Para el caso de la compresión simétrica, maneja el mismo método tanto para comprimir como para descomprimir los datos, cada operación demanda la misma cantidad de trabajo. Generalmente, este tipo de compresión se utiliza en la transmisión de datos. La compresión asimétrica en cambio necesita más trabajo para una de las dos operaciones; o bien para la compresión o descompresión, es frecuente buscar algoritmos para los cuales la compresión es más lenta que la descompresión [14].

Aunque cabe destacar que los algoritmos que ejecutan la compresión de datos con mayor rapidez que la descompresión pueden ser útiles cuando se manejan archivos de datos a los cuales se accede con muy poca frecuencia (por razones de seguridad), puesto que esto crea archivos compactos.

2.4.3. Codificación de Fuente y de Entropía

La codificación de fuente se basan en las propiedades de la fuente de datos a codificar, suelen tolerar perdidas en la codificación (códecs *lossy*) que perceptualmente pasan inadvertidas para el usuario. Son codificadores de propósito específico y de manera general obtienen mayores beneficios que los codificadores basados en la entropía. Suelen ser técnicas de compresión con pérdidas donde se obtienen tasas de compresión elevadas. Como ejemplos se pueden mencionar: codificación diferencial, codificación por transformada.

La codificación de entropía codifica los datos sin necesidad de conocer la naturaleza de éstos, es una codificación de propósito general, es decir, para todo tipo

de datos y son técnicas de compresión sin perdidas. Como ejemplos se tiene compresores estadísticos como Huffman, Aritméticos.

2.2.4. Codificación adaptativa, semiadaptativa y no adaptativa

Los codificadores adaptativos se ajustan a los datos que van a ser comprimidos, no parten de un diccionario ya preparado para un tipo de datos determinado. El más simple es el *Run Length Encoding* (RLE), el cual consiste en sustituir una serie de valores repetidos por una clave con indicador numérico; los compresores de uso general más populares utilizan métodos como este, por eso tardan más en empaquetar los datos que en descomprimirlos [9].

Por otro lado, un codificador semiadaptativo crea un diccionario según los datos que va a comprimir, es decir, define el diccionario mientras analiza el archivo y después lo comprime. Con este procedimiento se logra una mayor compresión pero introduce dos inconvenientes: la pérdida de velocidad al tener que leer el original dos veces, por un lado, y la necesidad de incrustar en el archivo el índice de claves por el otro [9].

Por el contrario, un codificador no adaptativo está basado en diccionarios para un tipo específicos de datos, es decir, implantan códigos con las combinaciones de bits que más se repiten estadísticamente, a estas secuencias se asignan códigos cortos y a otras menos probables claves más largas [9].

2.5. Fundamentos de la codificación de video digital

2.5.1. Jerarquía de los datos de video

Dentro de los estándares MPEG la unidad fundamental es el bloque el cual está formado por un grupo de 8x8 elementos de imagen. La unión de bloques se organiza y forma los macrobloques (MB) que son áreas de 16x16 pixeles.

Una imagen MPEG se encuentra constituida por segmentos, que no es más que una secuencia de macrobloques contiguos en el mismo orden de barrido de la imagen. Así mismo, cierto número de segmentos forman la denominada área activa o cuadro. Finalmente una secuencia de video se compone de uno o más grupos de imágenes (GOP).

En la Figura 2.5 se aprecia la jerarquía de la información de video destinada para MPEG.

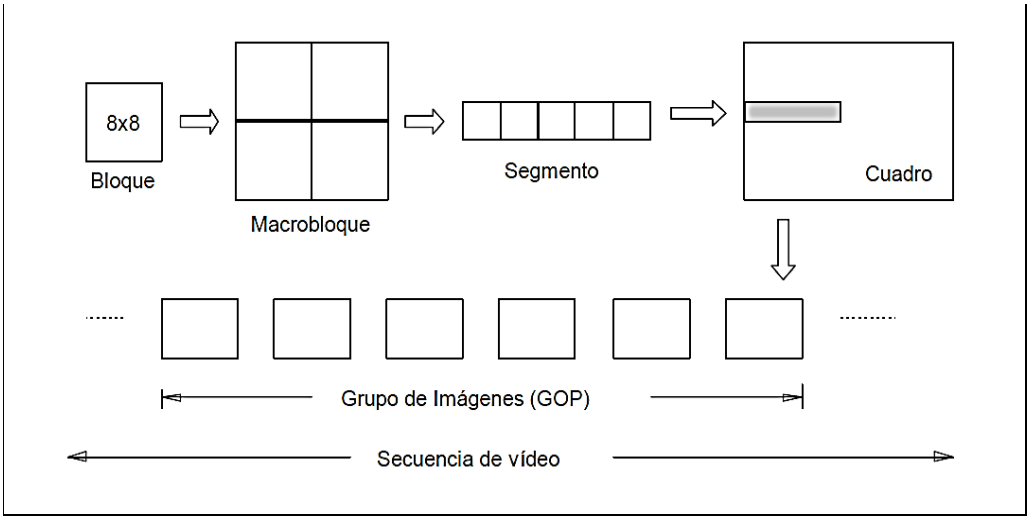

Figura 2.5. Jerarquía de los datos de video [1]

2.5.2. Estructura de las imágenes

El procesamiento de cada imagen no trabaja con la información a nivel de píxel, más bien las imágenes son divididas en áreas cuadradas de tamaño 16x16 para la información de luminancia (Y) y de tamaño variable para la información de crominancia (CB y CR), dependiendo del esquema de muestreo que se haya empleado para la digitalización de las imágenes [15].

El tamaño variable de los bloques se puede apreciar en la Figura 2.6 donde un macrobloque de 16x16 píxeles puede descomponerse en bloques de 8x16, 16x8 y 8x8 píxeles respectivamente. Posteriormente, cada bloque de 8x8 píxeles puede a su vez ser dividido en sub-bloques de 8x4, 4x8 o 4x4 píxeles.

El número de bloques para la luminancia siempre es de 4, mientras que el número de bloques de las crominancias puede ser 8, 4 o 2 en función del esquema de muestreo que se utilice (4:4:4, 4:2:2, 4:2:0).

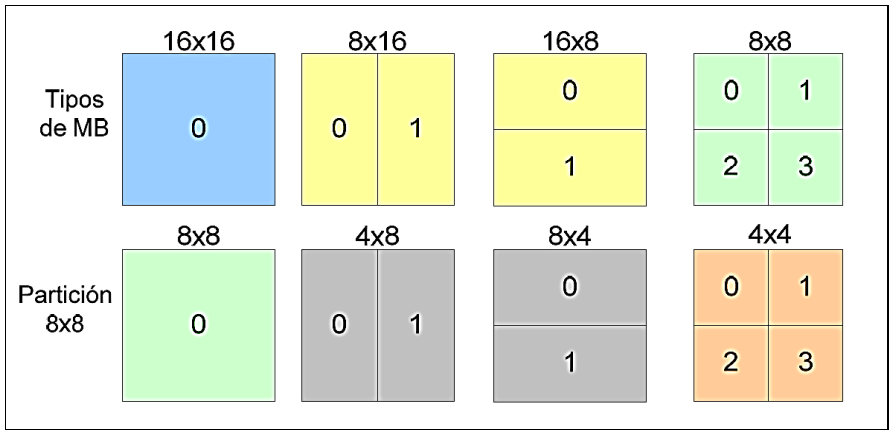

Figura 2.6. Partición de un macrobloque y una submacrobloque [16]

La Figura 2.7 se puede apreciar la estructura de un macrobloque, de acuerdo al esquema de muestreo utilizado. Empezando desde la parte inferior en el primer esquema las muestras de luminancia y crominancia poseen la misma resolución, en el segundo caso las muestras de crominancia posee la misma resolución vertical que las muestras de luminancia pero la mitad de resolución horizontal y en el caso del formato 4:2:0 las muestras de crominancia poseen la mitad de resolución horizontal y vertical de las muestras de luminancia, es decir, por cuatro muestras de Y solo habrán una muestra de Cr y una muestra de Cb, lo cual implica una tasa de compresión del 50% con respecto al formato RGB [16].

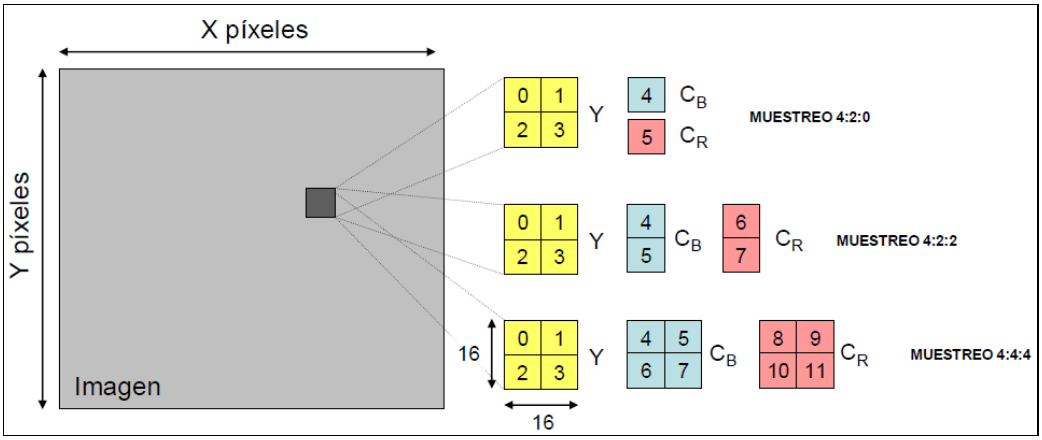

Figura 2.7. Estructura de un MB para diferentes esquemas de muestreo [12]

Si se emplea el esquema de muestreo 4:2:0, un MB estaría compuesto por un bloque de 16x16 pixeles representado por 256 muestras de Luminancia Y (dispuestos en cuatro bloques de 8x8 muestras), 64 muestras de Crominancia Azul Cb (un

bloque de 8x8) y 64 muestras de Crominancia Roja Cr (8x8 muestras), dando un total de seis bloques de 8x8.

En la Figura 2.8 se muestra como está compuesto un macrobloque de 16x16 pixeles en el esquema de muestreo (4:2:0) utilizado por el estándar H.264/AVC.

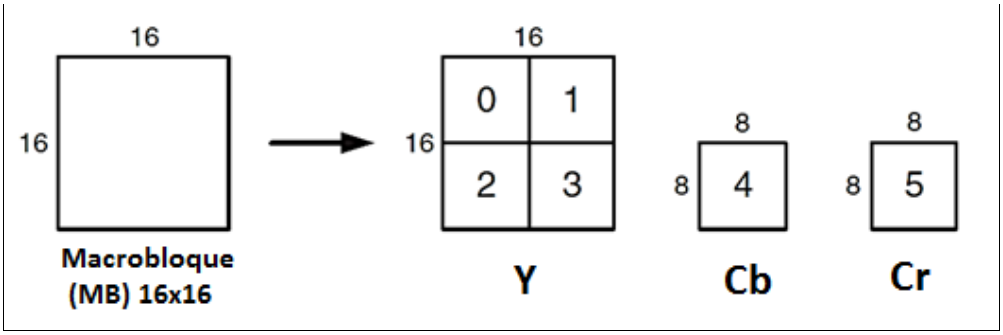

Figura 2.8. Componentes de Y, Cb, y Cr para el esquema de muestreo 4:2:0 [4]

En función del tipo de macrobloques se definen tres tipos de imágenes las cuales son las siguientes.

2.5.2.1 Las imágenes tipo I.

Son imágenes autónomas que se puede codificar de forma independiente sin hacer referencia a otras imágenes, es decir, sus macrobloques están codificados sin predicción. La primera imagen de una secuencia de vídeo es siempre un fotograma I el mismo que sirve como punto de inicio en nuevas visualizaciones o como puntos de resincronización si la transmisión de bits resulta dañada; la inserción de este tipo de imágenes permite evadir los errores asociados al proceso de codificación que se vayan acumulando y provoquen pérdidas significativas en la calidad de la secuencia. La desventaja de este tipo de fotogramas es que consumen muchos más bits, ya que al no tener como referencia a otros fotogramas la información a codificar es mayor. Este tipo de imágenes utiliza la predicción intra.

2.5.2.2 Las imágenes tipo P.

Se definen como imágenes las cuales explotan la redundancia temporal que existe entre varios fotogramas en el tiempo. Hacen referencia a partes de imágenes I o P anteriores para codificar solo la diferencia entre estas y la imagen actual, es decir, sus macrobloques pueden ser de tipo I, si no se ha localizado una predicción

adecuada; o de tipo P si en una imagen de tipo I o P anterior se ha localizado una buena predicción. Al enviar solo la diferencia la carga binaria respecto a los fotogramas I, pero con la desventaja de ser muy sensibles a la transmisión de errores, debido a la compleja dependencia con fotogramas P o I anteriores. Este tipo de imágenes utiliza la predicción inter.

En la Figura 2.9 podemos apreciar una secuencia en la que están imágenes del tipo P y del tipo I así como las relaciones entre ellas.

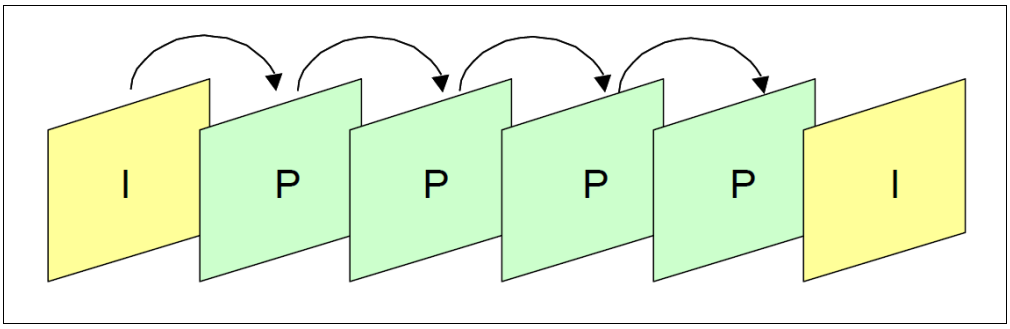

Figura 2.9. Secuencia de imágenes tipo I y P [12]

2.5.2.3 Las imágenes tipo B.

Son similares a las imágenes tipo P, explotando también la redundancia temporal. Hacen referencia tanto a imágenes anteriores como posteriores, es decir, sus macrobloques pueden ser del tipo I, P o B dependiendo de si el codificador ha localizado alguna predicción y si esta se encuentra en las imágenes anterior (referencia *backward*) y/o posterior (referencia *forward*) [15].

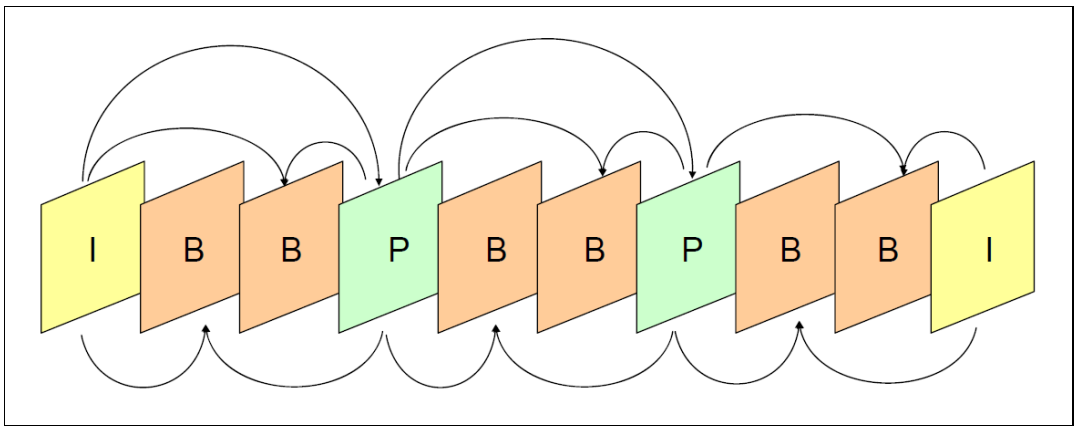

Figura 2.10. Secuencia de imágenes tipo I, P y B [12]

Cada imagen tipo B, independientemente del tipo de macrobloques de que este compuesta, depende de imágenes I y/o P anteriores y/o posteriores tal como se muestra en la Figura 2.10 donde se aprecia como utiliza como referencia un cuadro pasado y otro futuro. Estas imágenes hacen disminuir en gran medida la velocidad de codificación del sistema ya que se requieren menos bits para codificar fotogramas del tipo I o tipo P, esto debido a que la predicción realizada entre los dos fotogramas de referencia puede combinarse provocando que el residuo a codificar sea menor.

Es importante destacar que el orden de codificación no tiene que ser el mismo que el de visualización de un video, si no que éste dependerá del patrón de franjas con que se vaya a codificar debido a que al codificar un macrobloque del tipo B es necesario disponer de macrobloques ya codificados pertenecientes a planos anteriores y posteriores al actual, por tal motivo existe una restricción que condicionará el orden de codificación y harán que pueda ser distinto del de visualización.

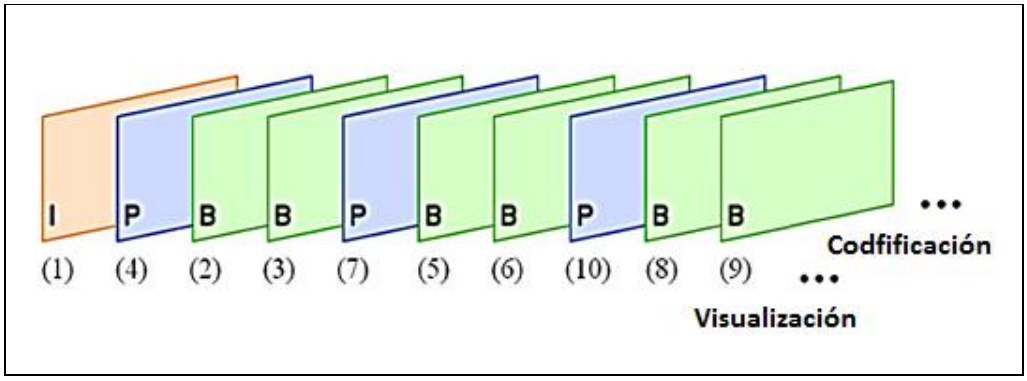

Figura 2.11. Orden de codificación y visualización de un video [18]

En la Figura 2.11 podemos diferenciar el orden de codificación con el de visualización, donde existe alteraciones en el orden de codificación las mismas que continúan durante todo el proceso de codificación; el primer plano a codificar es de tipo I, el único posible en este caso ya que no se dispone de ningún plano anterior codificado, ni por tanto de referencias. A continuación, se codifica un plano P que puede contener macrobloques I y macrobloques P que podrá utilizar como referencia el plano (1) ya codificado. Aquí se puede ver que ya no coincide el orden de codificación con el de visualización ya que este plano, el segundo en ser codificado corresponde con el cuarto en el orden de visualización. Esto se hace para que los dos

planos siguientes, el (2) y el (3), que son de tipo B, dispongan de referencias futuras ya codificadas. Finalmente se codificará el plano (2) de tipo B, que tendrá disponibles como referencias el plano (1) y el plano (4), y después el (3) que dispondrá como referencias pasadas del plano (1) y (2), y como referencia futura el plano (4) [18].

2.6. Conceptos básicos de MPEG

2.6.1. Historia

Desde 1988, el proyecto MPEG fue desarrollado por un grupo de cientos de expertos bajo los auspicios de la ISO y la IEC. Dicho proyecto se encarga de la elaboración de normas internacionales para la compresión de video, descompresión, procesamiento y representación codificada de imágenes en movimiento, audio y su combinación [18]. MPEG incluye parámetros como la predicción de movimiento en una escena y la identificación de objetos. Este grupo realiza sesiones plenarias y reuniones de subgrupos de los requerimientos, sistemas, vídeo, codificación de vídeo, vídeo 3D Audio, Gráficos 3D y comunicación.

La participación está abierta a los expertos, debidamente acreditados por un cuerpo apropiado de Estándares Nacionales. En promedio una reunión con la asistencia de más de 400 expertos de unos 20 países que representan a más de 200 empresas que abarcan todos los ámbitos de la industria con una participación en el audio digital, video y multimedia [6].

2.6.2. Estándares de codificación de video

Durante los últimos 30 años se han elaborado estándares basados en codificadores híbridos, los cuales tienen como principal objetivo reducir el régimen binario de salida manteniendo la calidad de imagen. Es así que se han desarrollado sucesivamente los estándares para video, por parte de la ITU H.261 (1984) y H.263 (1998) mientras que la ISO creo paralelamente estándares como MPEG-1 (1990), MPEG-2 (1995) y MPEG-4 (1999). Posteriormente ambos organismos colectivamente desarrollaron el estándar denominado MPEG-4 Parte 10 (2003) según la ISO y H.264 en términos de la ITU.

Los estándares destinados para la codificación de video recientemente han ido en aumento con la aparición de estándares desarrollados por la S*ociety of Motion Picture & Television Engineers* (SMPTE), dentro de los que se destacan VC-1, VC-2, VC-3 los mismos que han adquirido gran acogido por parte de grandes empresas como Microsoft.

2.6.3. Estándar MPEG - 1

Estandariza tres esquemas distintos para la codificación de ondas de sonido denominados *layers* I, II y III. No estandariza el codificador sino el tipo de información que éste debe producir y cómo un decodificador debe dividir, descomprimir y sintetizar esta información para obtener el sonido codificado. Esta compresión no produce la misma señal después de la decodificación, su objetivo es que el resultado sea muy parecido para el oído humano.

Básicamente analiza la señal en frecuencia y elimina aquellas componentes que se ven enmascaradas por otras (no se pueden oír) y empaqueta esta señal analizada en un *bitstream* MPEG-1 audio. Este *bitstream* tiene un formato fijo que descompone la señal en una secuencia de cuadros (*frames*) que, por norma general, son independientes unos de otros, lo que es adecuado para la difusión de audio.

Características de estándar MPEG-1:

Existen varios requerimientos planteados al momento de la realización de la tecnología LTE. Estos requerimientos clave fueron cristalizados por parte de la 3GPP en el año de 2005 y se los puede resumir de la siguiente manera:

- Frecuencia de muestreo: 32 kHz, 44.1 kHz y 48 kHz.
- Canales de audio: simple, dual, estéreo, y estéreo conjunto.
- Tasas de bits resultantes: 32 kbps.
- Puede transmitir sobre líneas de par trenzado a distancias modestas.
- Es usado para almacenar películas en CD-ROM.
- La parte de video de MPEG-1 propone solamente video progresivo no entrelazado, y puede tener una velocidad de hasta 1.5 Mbps.
- Posee acceso aleatorio.

- Avances y retrocesos rápidos para búsquedas.
- Reproducción en reversa.
- Sincronización audiovisual.
- Menor cantidad de errores.

2.6.4. Estándar MPEG - 2

Este estándar se diseñó de modo compatible al MPEG-1 manteniendo la máxima eficiencia de codificación de video agregando más flexibilidad en el formato del cuadro de entrada, mayor tasa de datos y mejor oposición al error; además se añadieron métodos de predicción para permitir la codificación de vídeo entrelazado y progresivo.

MPEG-2 se destinó para la codificación genérica de imágenes en movimiento y el audio asociado que crea un flujo de vídeo mediante tres tipos de datos de marco (cuadros intra, cuadros posteriores predecibles y cuadros predecibles bi-direccionales) arreglados en un orden específico llamado GOP. La tasa de bit de salida de un codificador MPEG-2 puede ser constante (CBR) o variable (VBR), con un máximo determinado por el reproductor, por ejemplo el máximo posible en un DVD de película es de 10.4 Mbit/s. MPEG-2 se extendió para soportar resoluciones mayores, entre ellas video digital de alta calidad (DVD), TV digital de alta definición (HDTV), medios de almacenamiento interactivo (ISM), retransmisión de video digital *Digital Video Broadcasting* (DVB) y televisión por cable.

Características de estándar MPEG-2:

- La relación de cuadros por segundo está bloqueado a 25 (PAL)/30 (NTSC) al igual que en MPEG-1.
- Soporta varios niveles de resolución: bajo (352 x 240), medio (720 x 480), alto 1440 (1440 x 1152) y alto (1920 x 1080).
- La resolución baja es para las VCRs y por compatibilidad hacia atrás con MPEG-1. La resolución media es la normal para la difusión NTSC.
- Posee una velocidad de transmisión de 2 Mbps hasta 15 Mbps.

- Tiene opciones de selección, ofrece señales de video entrelazado o progresivo, relaciones de aspecto 4:3 o 16:9, capacidad de aceptar diferentes muestreos principalmente de la forma 4:2:2 y 4:2:0.
- Presenta cuantificación mejorada, mejoras en acceso aleatorio y varios programas multiplexados.
- La relación de cuadros por segundo está bloqueado a 25 (PAL)/30 (NTSC) al igual que en MPEG-1.

2.6.5. Estándar MPEG - 4

Desarrollado para aplicaciones audiovisuales que requieren interacción por lo que este estándar es conocido formalmente como "Codificación de Objetos Audiovisuales", el cual está destinada a definir una norma para la codificación audiovisual que satisfaga las nuevas necesidades de comunicación, interacción, servicios de emisión u otros servicios resultantes de la convergencia de distintas tecnologías que normalmente se encontraban separadas; como son los casos de la comunicación, computación y entretenimiento.

Características de estándar MPEG-4:

- Utiliza un lenguaje conocido como BIFS, que permite añadir y borrar objetos, cambiar sus propiedades visuales o acústicas.
- El lenguaje BIFS se relaciona directamente con el VRML muy utilizado para describir objetos de tres dimensiones en Internet y su interacción con los usuarios, incluyendo algoritmos y herramientas para velocidades de 5 y 64 kbps.
- Acepta métodos de barrido progresivo y entrelazado, resoluciones espaciales de luminancia y crominancia 4:0:0, 4:2:0, 4:2:2.
- Posee una velocidad de transmisión de 64 kbps hasta 2 Mbps.
- Acepta métodos de barrido progresivo y entrelazado, resoluciones espaciales de luminancia y crominancia 4:0:0, 4:2:0, 4:2:2.
- Su principal aplicación son los flujos de medios audiovisuales, la distribución en CD, la transmisión bidireccional y la emisión de televisión.

- Presenta las siguientes tasas de bits baja (64 kbps), intermedia (64 - 384 kbps) y alta (384 kbps - 4 Mbps).
- Codifica objetos dentro de una figura arbitraria que forme parte de una escena; en lugar de codificar todo el cuadro completo como se hacía en MPEG-2 y MPEG-1.
- Presenta video de alta calidad, música y voz con un buen ancho de banda, objetos en 3D genéricos y específicos; bajos niveles de error para altas transmisiones de datos comprimidos sobre canales ruidosos

2.6.6. Estándar H.261

Basado en la estructura básica de 64 kbps de ISDN, creado para velocidades de datos múltiplos de 64 kbps, transporta contenidos de video que son manejados con el protocolo RTP y con cualquiera de los protocolos subyacentes de RTP. Permite solucionar el problema en relación a los distintos formatos existentes para televisión NTSC en América y PAL en Europa, ya que se establecen dos nuevos formatos para el monitoreo de imágenes que son aplicables a sistemas de videoconferencia y gráficos de alta resolución.

Características de estándar H.261:

- La velocidad de datos que opera se fijada entre 40 kbps y 2 Mbps.
- Hace posible transmitir imágenes de TV de calidad aceptable con bajos requerimientos de ancho de banda, dichos anchos de banda se han reducido en gran magnitud para logra comunicaciones de bajo coso sobre redes digitales conmutadas.
- Permite solventar el problema de relación a los distintos formatos existentes para televisión (NTSC en América y PAL en Europa), para ello utiliza nuevos formatos de monitoreo de imágenes.
- Los formatos de monitoreo de imágenes que utiliza son: (CIF), el cual especifica una resolución de 288 líneas de 352 pixeles, cada una a una velocidad de 30 imágenes por segundo y (QCIF), el cual especifica una resolución de 144 líneas de 176 pixeles, cada una a una velocidad de 30 imágenes por segundo.

2.7. Estándar H.264/AVC

Este formato de compresión proporciona mayor eficiencia de codificación debido a que la forma de operación de cada bloque de su diagrama lo hace de una manera funcional. Incluye dentro de sus principales funcionalidades son: el tamaño variable de los macrobloques a comprimir, resolución de un cuarto de píxel para la compensación de movimiento, compensación de movimiento con múltiples imágenes de referencia y filtro antibloqueo el cual elimina el efecto de bloque producido por la codificación de video.

Características de estándar H.264:

- Es un codificador de video extremadamente escalable, el cual permite entregar una excelente calidad de video y compresión.
- Mejora la calidad de imagen conservando la misma relación S/N.
- Es robusto a errores, esto lo logra incorporando un ordenamiento tolerante de macrobloques y la transmisión redundante de cuadros para evitar la propagación de errores.
- Flexibilidad para admitir una amplia gama de aplicaciones con diferentes requisitos de frecuencia de bits.
- Capacidades de baja latencia y mejor calidad para latencias mayores.
- Posee codificación de entropía y filtro de desbloqueo perfeccionados.
- Tiene compensación y predicción de movimiento mejorada utilizando alta precisión tomando varias imágenes como referencia.
- Propone nuevas formas de partición de bloques de modo que proporcionan una mayor exactitud en la estimación de movimiento.
- La velocidad de datos que opera se encuentra entre 64 kbps a 150 Mbps.
- Ahorro sustancial de bitrate (hasta de un 50%) en relación a estándares predecesores, manteniendo la misma calidad de imagen.
- Es la solución eficiente al servicio de Televisión Digital, flujo de video por IP y las video comunicaciones.

En los distintos estándares de codificación suelen definir lo que se denominan perfiles y niveles; los perfiles definen un conjunto de características que deben

soportar tanto codificadores como decodificadores si pretenden declararse como compatibles de dicho perfil. Los perfiles detallados en el estándar H.264/AVC se presentan a continuación:

2.7.1. Perfil *Baseline*

Destinado para dispositivos con recursos limitados, proporciona un mínimo coste computacional y tiene máxima robustez frente a errores. Se emplea para aplicaciones de teleconferencia como: teléfonos móviles, *mobile* TV, videotelefonía, videoconferencia y comunicaciones inalámbricas.

2.7.2. Perfil *Main*

Usado para dispositivos generales de consumo, ofrece una máxima tasa de compresión pero menor énfasis en robustez en difusión de TV y almacenamiento de video como: DVD y Blu-ray.

2.7.3. Perfil *Extended*

Mantiene una relación entre los dos perfiles anteriores con atención especial a las aplicaciones de video *streaming*, es decir servicios multimedia en internet.

2.7.4. Perfil *High*

Destinado para codificar video de alta resolución HD con un máximo de 8 bits por muestra y es muy utilizado en el campo del vídeo profesional. En la Tabla 2.2 se muestra los perfiles del estándar H.264 y sus correspondientes herramientas.

Tabla 2.2. Perfiles del estándar H.264/AVC [19]

Perfiles Características	*Baseline*	*Main*	*Extended*	*High*
Imágenes I	X	x	x	x
Imágenes P	X	x	x	x
Imágenes B		x	x	x
Filtro anti-bloqueo	X	x	x	x
Tamaño de bloque variable	X	x	x	x
CAVLC	X	x	x	x
CABAC	X			
Codificación entrelazada		x	x	x
Particionamiento de datos			x	

Los niveles en cabio definen distintos límites en parámetros de codificación como la tasa máxima de bits por segundo, tamaño de la imagen o el máximo número de macrobloques que los decodificadores deben ser capaces de cumplir si quieren declararse como compatibles con dicho nivel.

2.7.5. Diagrama de bloques del decoder de video H.264/AVC

El códec de video H.264 consta de varios procesos basados en bloques, los cuales se pueden apreciar en la Figura 2.12 donde se muestra el diagrama de bloques del codificador.

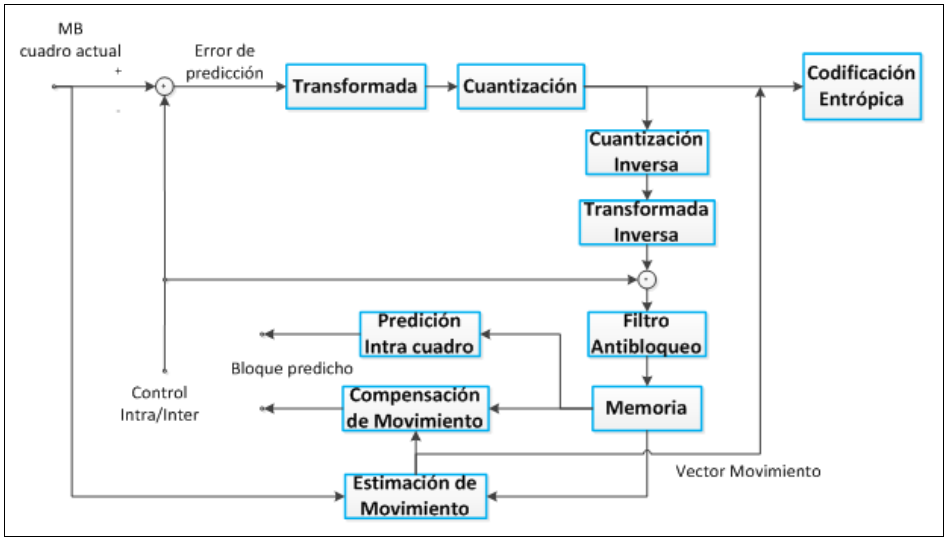

Figura 2.12. Diagrama de bloques del codificador H.264/AVC [21]

El codificador H.264/AVC posee un módulo de predicción intra cuadro el cual se encarga de atenuar la redundancia espacial basándose en la semejanza de los píxeles adyacentes del MB a codificar como se aprecia en la Figura 2.13.

Figura 2.13. Semejanza de pixeles para la predicción intra cuadro

Para ello la imagen se divide en bloques o subimágenes de tamaño reducido donde los cuadros I son comprimidos a través una transformada de coeficientes discretos. Una vez obtenidos estos coeficientes se realiza el proceso de cuantificación donde se filtran los coeficientes principales (coeficientes de baja frecuencia). A partir de aquí, se reordenan los coeficientes, se les aplica una codificación entrópica y se obtiene la imagen con predicción intra cuadro, dicho proceso se muestra en la Figura 2.14.

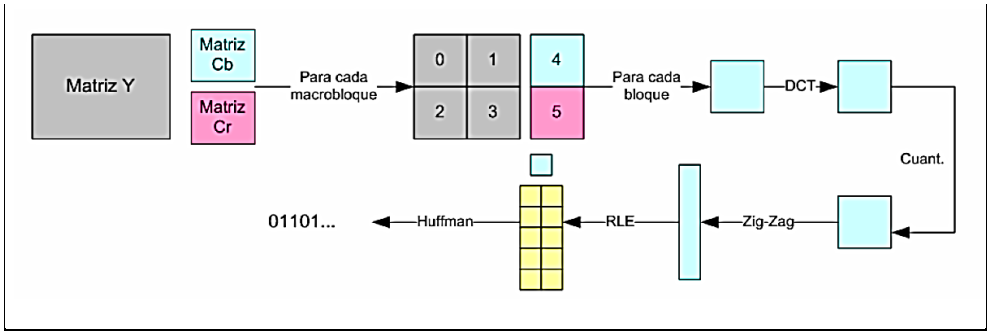

Figura 2.14. Esquema de codificación de las imágenes I [22]

El módulo de predicción inter cuadro, el cual está compuesto por el proceso de estimación de movimiento (ME) y compensación de movimiento (MC), en conjunto se encargan de disminuir la redundancia temporal aprovechando el hecho de que un píxel se repita a lo largo del tiempo, para ello se asignan bloques de diferentes tamaños en función de la cantidad de movimiento que exista entre imágenes.

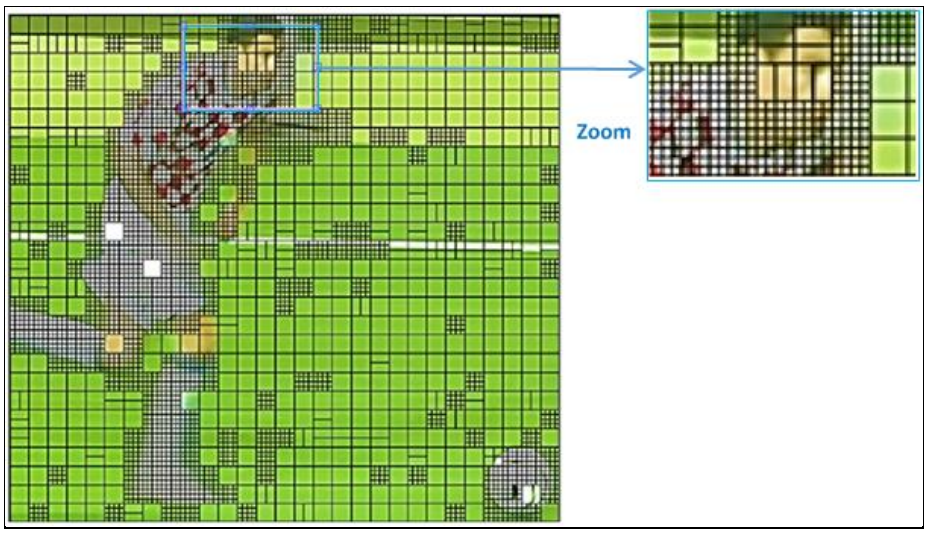

Figura 2.15. Asignación de MBs para la predicción inter cuadro

En la Figura 2.15 se visualiza que las zonas con menor cambio se le asignan macrobloques de mayor tamaño 16x16, mientras que para las zonas con mayor movimiento se descompone en sub-bloques de 16x8, 8x16, 8x8, 8x4, 4x8 o 4x4.Utilizando técnicas predictivas por medio de la comparación de bloques del cuadro actual con respecto a los bloques de un cuadro de referencia, solo se envía la diferencia entre la imagen de actual y la de referencia.

Como consecuencia la información a transmitir es menor que en las imágenes tipo I, es decir, se obtiene la imagen de predicción a partir de vectores de movimiento de imágenes pasadas y/o futuras mediante el proceso de codificación de imágenes P y B mostrados en la Figura 2.16 y Figura 2.17.

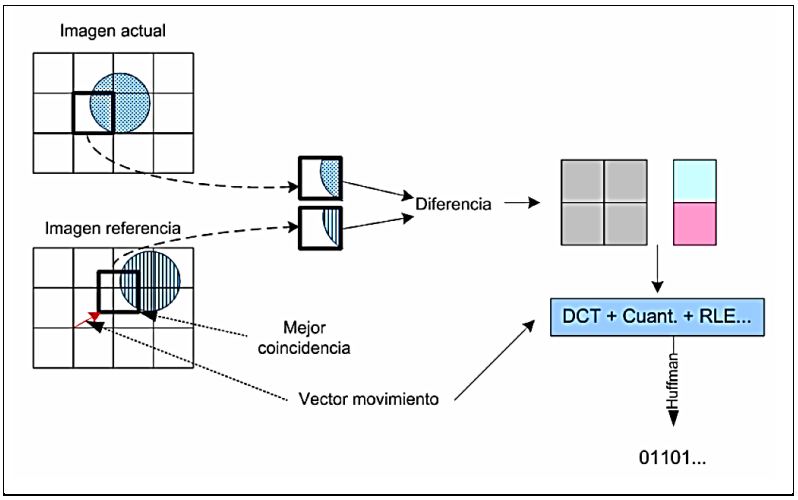

Figura 2.16. Esquema de codificación de las imágenes P [22]

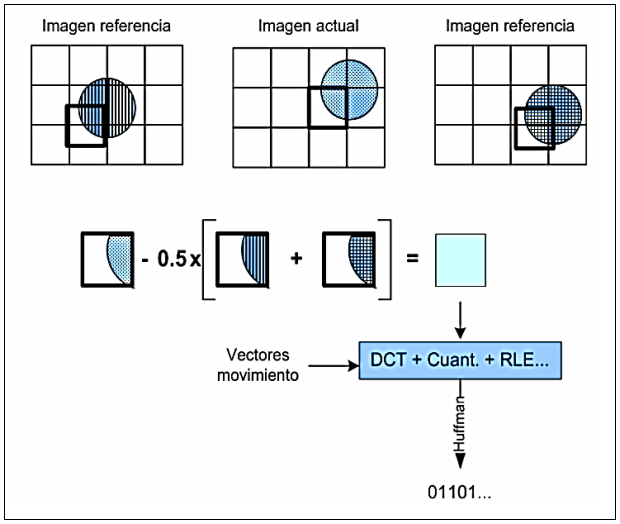

Figura 2.17. Esquema de codificación de las imágenes B [22]

El tipo de predicción que se desee realizar es seleccionado por medio del circuito de control mostrado en la Figura 2.12, para luego ser restado de la imagen original. Luego el residuo o error de los módulos ya mencionados es enviado al bloque de Transformada, cuya función es la de disminuir la redundancia espacial en el dominio de la frecuencia.

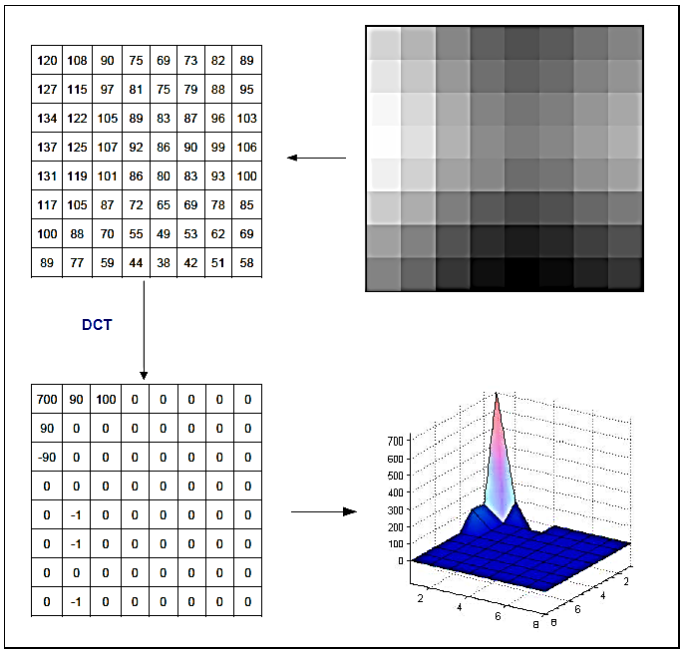

Figura 2.18. Codificación por Transformada DCT [22]

La transformada que se aplican comúnmente es la DCT, en la Figura 2.18 vemos un ejemplo numérico. El bloque de Cuantización se encarga de atenuar la redundancia espacial, donde parte de la información se descarta o se cuantifica con un número de niveles muy reducido.

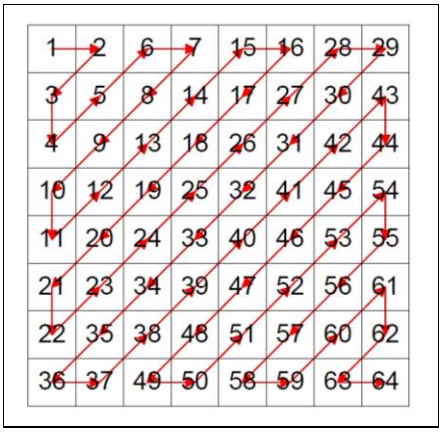

Figura 2.19. Recorrido en Zig-Zag de ordenación de coeficientes [22]

Posteriormente el bloque de Codificación de Entropía tiene por objetivo disminuir la redundancia entrópica para ello reordena los coeficientes y los serializa para que se pueda realizar dicha codificación utilizando el método del Zig-Zag mostrado en la Figura 2.19.

Finalmente se utilizan los bloques de Cuantización y de Transformada inversa para obtener un cuadro reconstruido después de pasar por el filtro de desbloqueo o filtro de loop, el cual es utilizado para atenuar o suavizar los bordes de los bloques de la imagen después de su codificación [20]. Dicho efecto se puede observar en la Figura 2.20 donde los defectos pixelados en la imagen muy comprimida de la izquierda se reducen al aplicar un filtro de eliminación de bloques, como se muestra en la imagen de la derecha.

Figura 2.20. Paso de la imagen comprimida por el Filtro anti bloques.

En la Figura 2.21 se puede visualizar los bloques que forman parte del decodificador H.264/AVC el cual se encarga de procesar la secuencia de bits generada por el codificador.

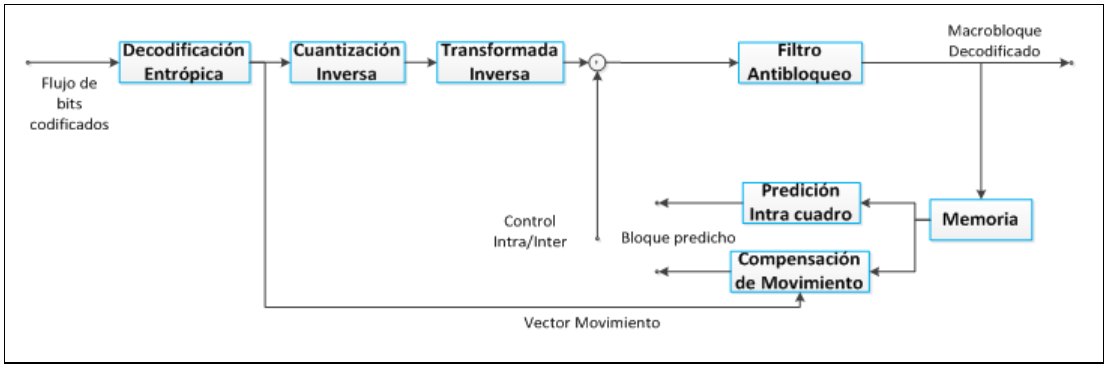

Figura 2.21. Diagrama de bloques del decodificador H.264/AVC [21]

La función que efectúa el decodificador es la de obtener las imágenes reconstruidas con las pérdidas que haya provocado el proceso de codificación. Para cada macrobloque se realiza la descodificación de entropía, la cuantificación inversa y la transformación al dominio espacial. Si se trata de un macrobloque codificado en modo intra, los datos obtenidos son directamente los píxeles que deben almacenarse en la memoria de imágenes descodificadas. Si por el contrario se está procesando un macrobloque de tipo inter, los datos recuperados se corresponden con el vector de movimiento y el error de predicción por lo que el bloque de compensación de movimiento debe obtener la predicción a partir de las imágenes descodificadas anteriormente para ser sumada con el error antes de almacenar el resultado en la memoria.

2.7.6. Técnicas de estimación de movimiento en el estándar H.264

Las principales innovaciones del códec de video H.264 para la compresión de video con respecto al proceso de estimación de movimiento (ME) son: la selección del tamaño de las particiones de un MB, además de la generación de un vector de movimiento (MV) con valores enteros y fraccionarios con una precisión de 1/2 y 1/4 de pixel, así como el uso de múltiples cuadros de referencia previamente decodificados.

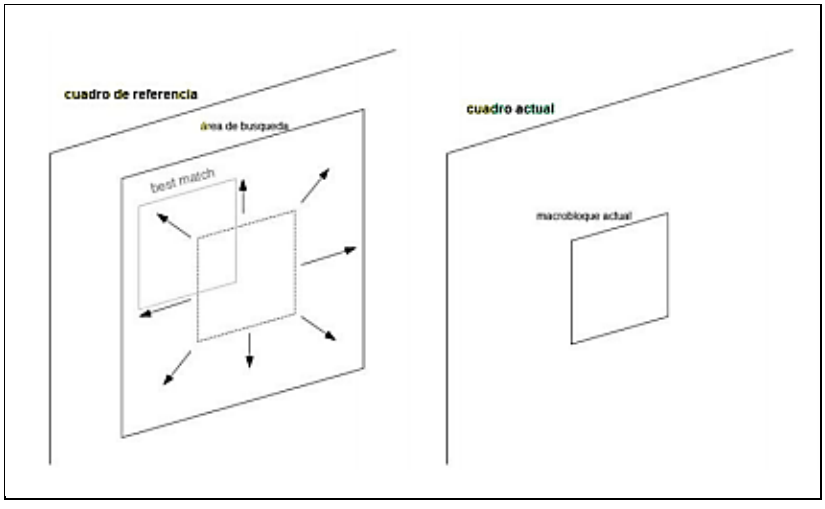

Figura 2.22. Proceso de estimación de movimiento [23]

El proceso ME consiste en la comparación de una región de MxN pixeles del cuadro actual con algunas o todas las posibles regiones de MxN pixeles dentro de un

área de búsqueda del cuadro de referencia. A la región de MxN pixeles se le denomina bloque, además se denomina bloque actual o de referencia dependiendo del tipo de cuadro al que pertenece. El resultado obtenido de este proceso es el bloque de referencia que mejor se aproxima al bloque actual [20]. A este resultado se le denomina *best match* como se puede apreciar en la Figura 2.22.

El criterio utilizado para encontrar el bloque de referencia con el *best match* es la energía residual que se obtiene de restar el bloque actual con un bloque candidato del área de búsqueda. El bloque que obtenga la menor energía residual tendrá el *best match*, lo cual significa que se encontró al bloque con mayor similitud al bloque actual. Además de lo anterior, dentro del proceso, se calcula el desplazamiento de la posición del bloque actual con respecto a la posición del bloque candidato obteniendo un parámetro denominado MV, el cual se usará en el proceso de compensación de movimiento, con el cual se completa todo el proceso de predicción inter cuadro.

Tanto el MV como el bloque candidato con la menor energía residual, son codificados por los procesos de transformadas directas, cuantización y entropía. Luego esta información codificada puede ser recibida por cualquier equipo que contenga un decodificador bajo el mismo formato H.264/AVC. El decodificador utiliza el MV recibido para reconstruir el bloque predicho y formar el cuadro codificado [20].

2.7.6.1 Estimación de Movimiento Entera (IME)

Como se ha explicado hasta el momento el proceso de ME contribuye en la compresión de datos de video, expresando la información como MV en lugar de toda la información correspondiente a los pixeles que forman parte de un MB. Así mismo se explicó que incrementando la precisión del movimiento de un MB se obtienen mejores tasas de compresión sin perder detalles en el movimiento que puede suceder entre los cuadros que forman una cadena de video.

El primer proceso que se realiza en ME es la estimación de movimiento de los pixeles enteros a lo que se denomina Estimación de Movimiento Entera (IME). El siguiente proceso es la Estimación de Movimiento Fraccional (FME) donde se

calcula el desplazamiento con precisión de *Half-Pixel* y/o *Quarter-Pixel* alrededor del MB con el *best match* obtenido del proceso anterior.

2.7.6.2 Algoritmos de búsqueda para Estimación de Movimiento Entera

Para poder encontrar el *best match* en el área de búsqueda se debe efectuar la evaluación de la energía residual para todos los posibles bloques en el área de búsqueda del cuadro de referencia.

Existen tres formas de medir la energía residual: Suma de Diferencias Absolutas (SAD, *Sum of Absolute Diferences*), Error Cuadrático Medio (MSE, *Mean Squared Error*) y Error Absoluto Medio (MAE, *Mean Absolute Error*) expresados por las ecuaciones (2.3), (2.4) y (2.5) respectivamente.

$$\mathrm{MSE} = \frac{1}{\mathrm{N}^2}\sum_{\mathrm{i}=0}^{\mathrm{N}-1}\sum_{\mathrm{j}=0}^{\mathrm{N}-1}\left(\mathrm{C_{ij}} - \mathrm{R_{ij}}\right)^2 \tag{2.3}$$

$$\mathrm{MAE} = \frac{1}{\mathrm{N}^2}\sum_{\mathrm{i}=0}^{\mathrm{N}-1}\sum_{\mathrm{j}=0}^{\mathrm{N}-1}\left|\mathrm{C_{ij}} - \mathrm{R_{ij}}\right| \tag{2.4}$$

$$\mathrm{SAD} = \sum_{\mathrm{i}=0}^{\mathrm{N}-1}\sum_{\mathrm{j}=0}^{\mathrm{N}-1}\left|\mathrm{C_{ij}} - \mathrm{R_{ij}}\right| \tag{2.5}$$

Todas las fórmulas se basan en la resta de los valores de los pixeles del MB actual (C_{ij}) con los del MB de referencia (R_{ij}) con la misma posición relativa. SAD es la medida más ampliamente utilizada de la energía residual por la simplicidad del coste computacional que demanda y a la calidad de los MVs generados.

El algoritmo óptimo de búsqueda *Full Search* (FS) consiste en aplicar un criterio de búsqueda como el SAD a cada posible bloque en el área de búsqueda del cuadro de referencia.

Para poder explicar mejor en qué consiste la IME utilizando el algoritmo *Full Search*, se empleará la Figura 23 en (a) se tiene un cuadro actual con un bloque de NxM píxeles, luego para poder encontrar el *best match* se tendrá que generar un área de búsqueda alrededor de la posición del bloque actual en el cuadro de referencia.

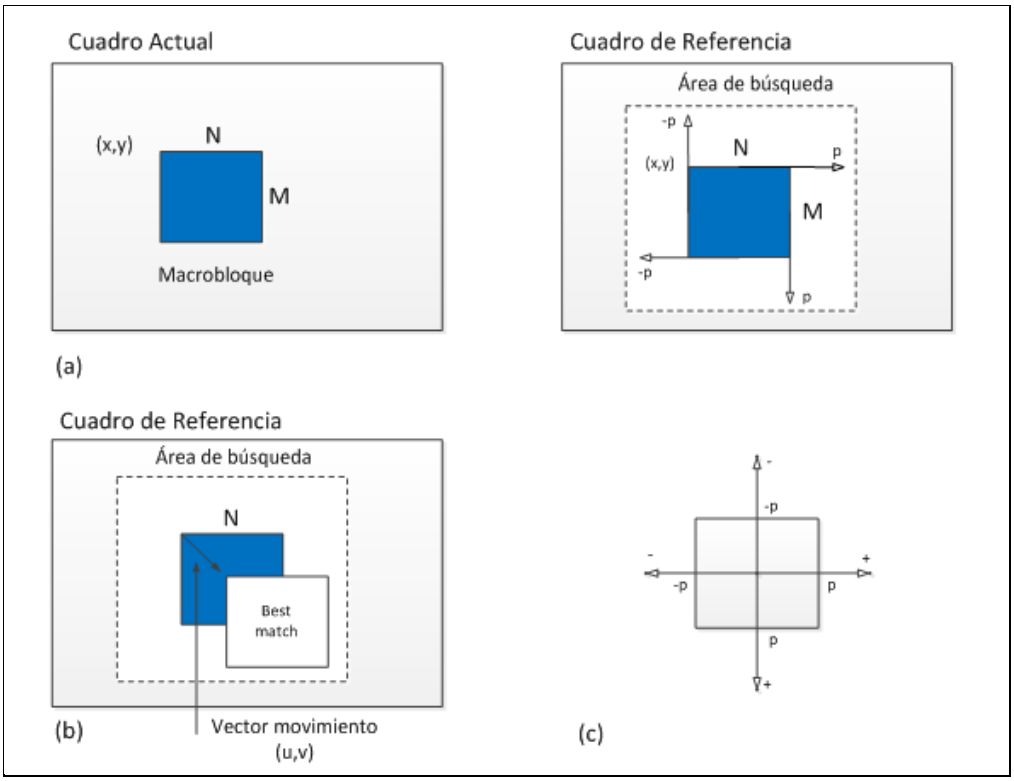

Figura 2.23. Proceso de Estimación de Movimiento Entera [23]

El área de búsqueda se generará a partir del bloque de NxM píxeles pero expandiendo en píxeles por cada borde como se muestra en la parte derecha de (a). Aplicando el algoritmo de búsqueda se puede encontrar el bloque con el *best match cuya posición* (x + u, y + v) respecto a la posición del bloque actual (x, y) permite generar el MV movimiento (u, v). Las coordenadas del MV tienen el siguiente rango: $-p \leq u \leq p$ y $-p \leq u \leq p$.

2.7.6.3 Estimación de Movimiento Fraccional (FME)

Según el estándar H.264/AVC, los movimientos que suceden de un cuadro para otro no están restringidos a posiciones enteras de pixel, se pueden utilizar MVs con valores fraccionarios de 0.5 y 0.25 pixeles.

El desplazamiento del MB o sub-bloque del cuadro de referencia con respecto al del cuadro actual se expresa por medio de un MV, cuya precisión es de 1⁄2 o 1⁄4 de pixel para el componente de Luminancia (Y) y la mitad de precisión de pixel para los componentes de Crominancia (Cr o Cb), es decir una precisión de 1⁄4 o 1⁄8 de pixel respectivamente. Esto se debe a que la resolución de las muestras de Cr y Cb es la

mitad del valor respecto a la resolución de las muestras de Y, como fue explicado en la sección anterior. Sin embargo, las muestras en las posiciones fraccionarias o sub-muestras tanto para Y como Cb y Cr deben de ser calculadas por medio de un algoritmo denominado Interpolación.

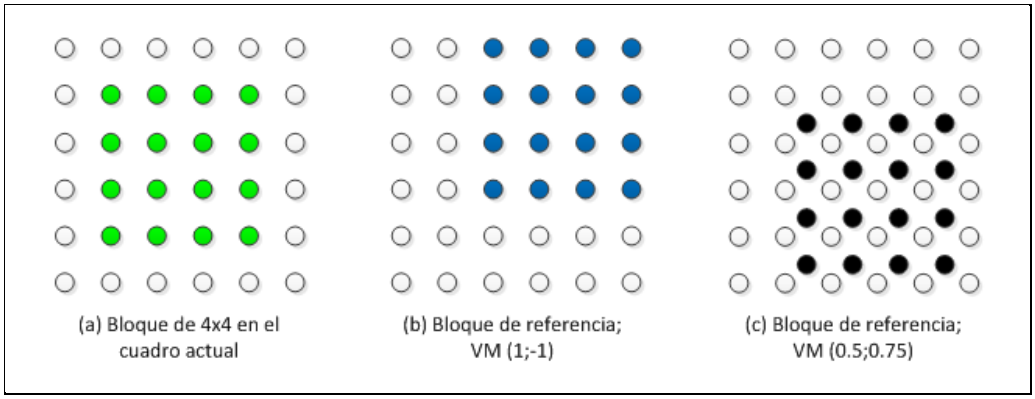

Figura 2.24. Estimación de Movimiento Fraccional [23]

En la Figura 2.24 se muestra un ejemplo de FME para un sub-bloque de 4x4 pixeles. El sub-bloque del cuadro actual de 4x4 pixeles representado por los puntos verdes de la parte (a) sirve como referencia para generar una región de búsqueda en el cuadro de referencia alrededor de la posición del bloque actual. Si las componentes verticales y horizontales del cuadro de referencia son números enteros entonces los pixeles del sub-bloque en el cuadro de referencia serán representados por los puntos azules como se muestra en la parte (b) si el desplazamiento es fraccionario, se debe calcular por interpolación de los pixeles enteros (puntos blancos) los pixeles fraccionarios (puntos negros) los cuales son mostrados en la parte (c).

2.7.6.4 Interpolación para precisión de *Half-Pixel*

Para poder encontrar un bloque con precisión de *Half-Pixel* que posea el *best match* con respecto al macro bloque actual será necesario generar un MB compuesto de *Half-Pixels*. La unidad de interpolación *Half-Pixels* se encargará de generarlos a partir del MB compuesto por pixeles enteros. Para calcular el valor de Luminancia de un *Half-Pixel* adyacente a dos posiciones enteras, se aplica un Filtro FIR a las muestras enteras de Luminancia que forman una línea vertical u horizontal en el MB.

De la Figura 2.25, por ejemplo, para calcular el valor de luminancia del *Half-Pixel* "y", se tiene que calcular un valor intermedio "y1" aplicando el Filtro FIR (en este caso de 6-taps) a la línea de pixeles enteros formados desde la A hasta la F. El valor de luminancia (Y) de los pixeles varía desde 0 a 255. Los pixeles enteros se muestran como pequeños cuadros de color gris con una letra mayúscula de referencia (A, B ,C, D, E, F, K, L, H, N, O, P, U, S, I, J, Q, W, D, N, R y T) y algunos de los *Half-Pixel* como cuadros blancos (aa, bb, cc, dd, ee, ff, gg, hh, g, m, y, x y s).

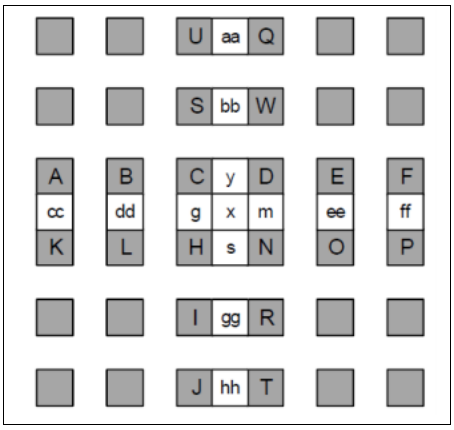

Figura 2.25. Interpolación *Half-Pixel* para un MB de 6x6 píxeles [23]

Es importante notar que hay tres tipos de *Half-Pixels*: los de tipo H, que son aquellos que se calculan a partir de la línea de pixeles horizontales más cercanos a la posición del *Half-Pixel*, los de tipo V que son aquellos que se calculan a partir de la línea de pixeles verticales más cercanos a la posición del *Half-pixel* y por último lo de tipo D que son los *Half-Pixels* alineados diagonalmente con pixeles enteros, para lo cual se toma la línea horizontal de *Half-Pixels* más próxima.

En la Figura 2.26 se observa un área de *Half-pixels* obtenidos a partir del bloque de 4x4 pixeles para lo cual se utilizó tres columnas adicionales de píxeles enteros por cada lado lateral y tres filas en cada borde superior e inferior respectivamente, por ejemplo los *Half-Pixels* h1, h6, h11 y h16 se obtienen a partir de la línea de pixeles enteros horizontal formado por tres pixeles a la derecha (que forman parte del bloque de 4x4) y tres pixeles a la izquierda los cuales forman parte de las tres columnas adicionadas, lo mismo se aplica a los *Half-Pixels* tipo V y D.

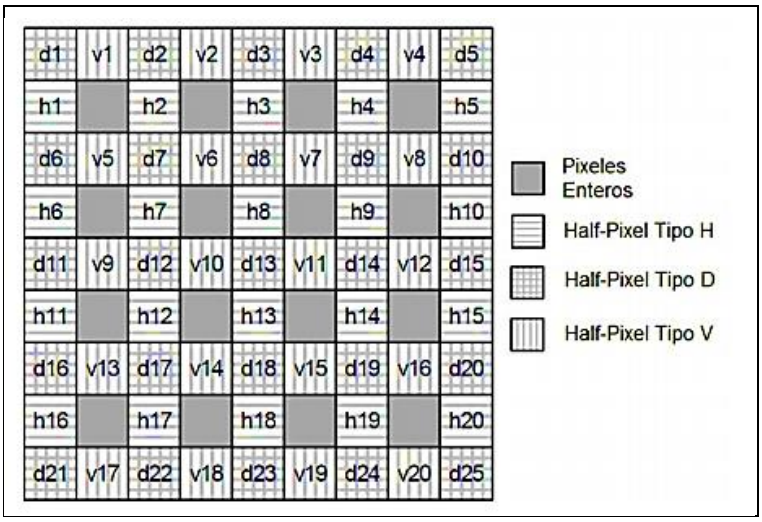

Figura 2.26. *Half-Pixels* interpolados a partir de un bloque de 4x4 píxeles [23]

En la nueva área de búsqueda hay dos posibles coincidencias con bloques del mismo tamaño formado por *Half-Pixels* de tipo V, dos posibles coincidencias con el área formada por los de tipo H y cuatro posibles coincidencias con el área formada por los de tipo D.

2.7.6.5 Interpolación con Precisión de *Quarter*-Pixel

La precisión de *Quarter-Pixel* (0.25 píxel) es una de las nuevas características del formato H.264/AVC que permite aumentar aún más la tasa de compresión de video. Después del proceso de IME y FME con precisión *Half-Pixel* se compara el menor SAD obtenido de ambos procesos. Si el menor SAD obtenido por FME con precisión *Half-Pixel* es menor que el obtenido por IME se obtiene un MV desplazado en ±0.5 píxeles para ambas direcciones (x e y) obtenido del FME, caso contrario sólo se considera el MV obtenido del IME.

Luego a partir del MB de referencia con la posición obtenida de los procesos anteriores, se aplicará un refinamiento para obtener una nueva área de búsqueda formada por *Quarter-Pixels*. Esta área es generada a partir del área del MB formado por píxeles enteros (encontrado por IME) y el área formada por *Half-Pixels* (generada por Interpolación del MB entero). A este refinamiento se le denomina Interpolación *Quarter-Pixel* el cual consiste en aplicar un filtro bilineal a los píxeles y subpíxeles que forman parte del área generada por Interpolación *Half-Pixel*, siendo este filtro una simple media aritmética entre dos medios píxeles.

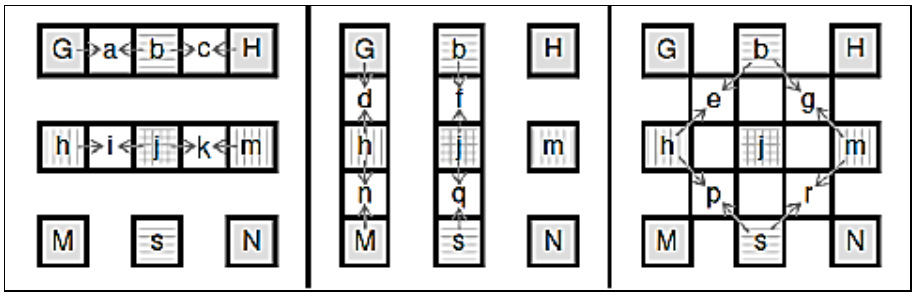

Figura 2.27. Interpolación para precisión de *Quarter-Pixel* [23]

En la Figura 2.27 se muestran los tipos de *Quarter-Pixel* que se obtienen por la Interpolación y que generan una nueva área de búsqueda. Al igual que los *Half-Pixels*, tenemos los de Tipo H, V y D. En el caso de Tipo H, sólo se utiliza los *Half-Pixels* adyacentes en una línea horizontal para calcular el nuevo sub-píxel, en este caso tenemos: a, c, i y k. En el caso de Tipo V es semejante al Tipo H pero en una línea vertical (d, n, f y q) y en el caso de los de tipo D, con los *Half-Pixels* que forman una diagonal (e, g, p y r).

Finalmente en la Figura 2.28 se muestra un área de búsqueda generada por Interpolación *Half-Pixel* y *Quarter-Pixel* alrededor de un bloque de 3x3 píxeles enteros. Donde IXY representan los píxeles enteros (cuadros resaltados), los *half-pixels*: HXY, VXY y DXY (cuadros) y los *quarter-pixels* son: haXY, hbXY, hcXY, hdXY, vaXY, vbXY, vcXY, vdXY, daXY, dbXY, dcXY y ddXY.

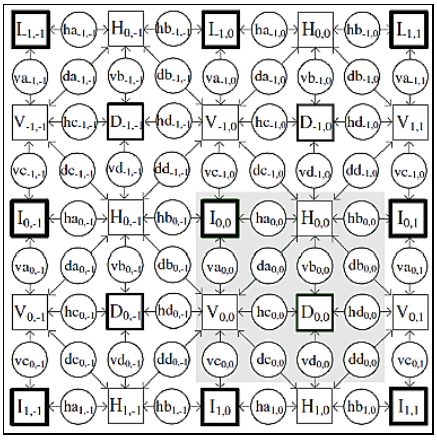

Figura 2.28. Interpolación *Half-Pixels* y *Quarter-Pixel* [23]

El proceso de búsqueda es similar a la efectuada en el proceso de interpolación *Half-Pixel*, ya que se utiliza el mismo criterio de comparación, SAD, para comparar

el bloque actual con los 9 bloques candidatos formados por *Quarter-Pixels* en el proceso de Interpolación. El menor SAD obtenido de la búsqueda con precisión *Quarter-Pixel* se comparará con el menor SAD obtenido en el proceso anterior obteniendo el menor SAD de todo el proceso de ME con un MV de precisión de 0.25 píxeles, el cual será codificados junto el bloque residual obtenido por los demás procesos del codificador H.264/AVC.

2.8. Empaquetamiento para televisión digital

La televisión digital (TVD) es una nueva técnica de radiodifusión de señales que consiste en convertir la imagen, el sonido, y los contenidos interactivos en información digital, es decir, la información no es enviada en forma de ondas que precisan grandes anchos de banda como ocurre con la televisión analógica, sino que la envía como ceros y unos (codificación binaria), aprovechando así los beneficios del procesamiento, multiplexación, codificación y modulación digital de las señales tanto de audio, vídeo y datos, con el único fin de optimizar la transmisión de las señales de televisión [21].

Este proceso de optimización se basa en la capacidad de transmitir múltiples señales en un mismo canal, así como también se consigue una interacción entre los usuarios y los productores de contenidos televisivos, ya que gracias a la forma digital de transmisión se cuenta con vías de retorno [22].

2.8.1. Estructura de un sistema de televisión digital

La estructura elemental de un sistema de televisión digital considerando la parte de transmisión como la parte de recepción de una señal digital consta de varias etapas entre las cuales se encuentra la generación, codificación, compresión, modulación y amplificación de una señal para la parte de transmisión; y sus respectivos procesos inversos para la parte de recepción.

En la Figura 2.29 y Figura 2.30 podemos apreciar la constitución de la parte de transmisión y recepción de un sistema TVD.

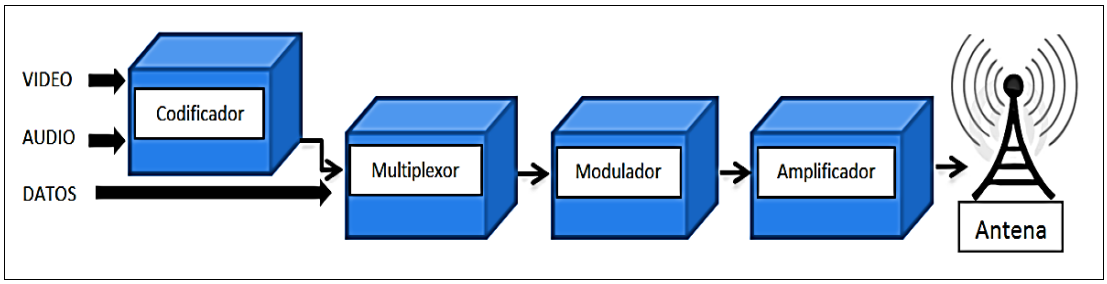

Figura 2.29. Estructura de Transmisión de Televisión Digital [25]

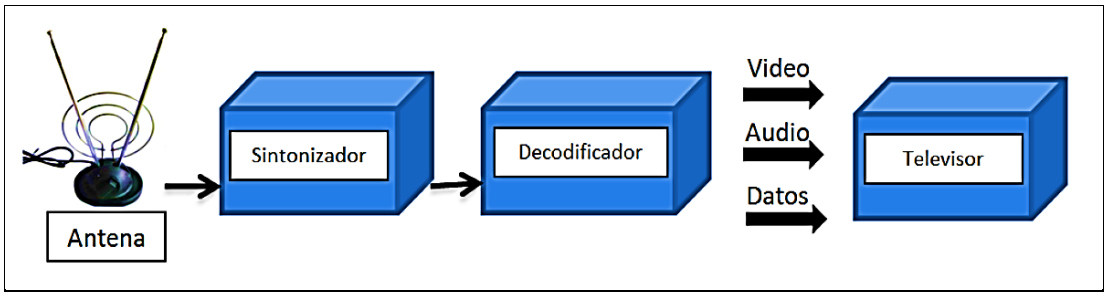

Figura 2.30. Estructura de Recepción de Televisión Digital [25]

En el presente documento se enfocará a lo referente con la trasmisión en la parte de generación de las señales digitales y de una manera más profunda la etapa de codificación de la señal de origen la cual puede provenir de varias fuentes hasta llegar al proceso de paquetización; para posteriormente dejar listo el video para que se pueda realizar la etapa de multiplexación de los diferentes flujos elementales formando un solo flujo digital a su salida.

2.8.2. Televisión Digital Terrestre (TDT)

La televisión digital tiene sus comienzos en el año 1982 cuando se desarrolló el estándar CCIR-601 (actualmente ITU-601). Dentro de las principales características del estándar son: ofrecer flexibilidad, capacidad de expansión y difusión de los servicios de transmisión de multimedia de acuerdo a la red en la que se desenvuelva. Dentro de lo que nos permitirá la TDT, se destaca:

- Incrementar notablemente el número de programas, servicios y señales actualmente disponibles.
- Mejorar la calidad de las imágenes y del sonido de las transmisiones y de las recepciones televisivas.

- Establecer servicios personalizados e interactivos (relación empresa televisiva-audiencia) de radiodifusión y telecomunicaciones.

- Facilitar la convergencia entre el sector audiovisual, las telecomunicaciones y la informática.

Cabe mencionar que en todo el mundo se han desarrollado varios estándares de televisión digital, Ecuador adoptó oficialmente el estándar Japonés-Brasileño (ISDB-Tb), para Televisión Digital Terrestre. El estándar *Integrated Service Digital Broadcasting – Terrestrial Built in* (ISDB-Tb), posibilita tener una transmisión digital en alta definición (HDTV) y en definición estándar (SDTV), transmisión digital simultánea para recepción fija, móvil y portátil, así como también interactividad [22].

El ancho de banda establecido para cada canal es de 6 MHz, la codificación y compresión de audio es por medio de MPEG AAC, la codificación y compresión del video utiliza el códec H.264/MPEG-4. Admite el transportar varios servicios como HDTV, SDTV y servicios de baja velocidad de transferencia pensado para transmitir televisión de baja resolución en teléfonos celulares llamado (*One Seg*) [22].

En la Figura 2.31 se aprecia la estructura del Estándar ISDB-Tb, el mismo que en la parte de transmisión de señales para televisión digital establece cinco factores esenciales:

- Codificación

- Paquetización (PES)

- Tablas PSI/SI

- Formación del *Transport Stream* (TS).

- Modulación.

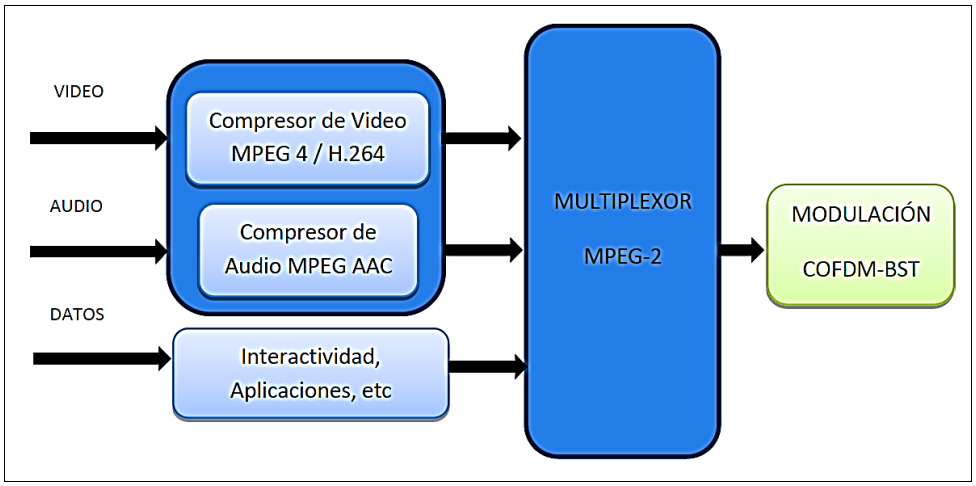

Figura 2.31. Estructura del Estándar ISDB-Tb [25]

2.8.3. Codificación

La codificación se basa en gestionar por separado al video, al audio y los datos, es decir, se crean paquetes de tamaño variable que dependen de la calidad del video y audio, de tal manera que tanto el video como el audio se compriman independientemente formando cada uno de ellos un flujo de datos, este flujo de datos se le conoce como *Elementary Stream* (ES).

En la actualidad existe una gama de mecanismos de codificación cuya finalidad es alcanzar una mayor optimización del espectro radioeléctrico, así como también una mejor calidad en la recepción.

El estándar ISDB-Tb como ya se mencionó para la compresión de video digital utiliza MPEG-4 (versión H.264) principalmente a que presenta mejoras significativas las cuales ya se trataron anteriormente en el presente documento, cabe destacar que el estándar H.264 es uno de los recursos más óptimos para el servicio de DTV y flujo de video por IP.

2.8.4. Paquetización (TS)

En esta parte se obtienen los bits provenientes del codificador de audio o video para estructurarlos en forma de paquetes llamados *Packetized Elementary Stream* (PES).

Los PES son paquetes de tamaño variable con una longitud máxima de 64 kbyte los cuales contienen los ES provenientes de la etapa de codificación, es decir, transportan PES de video, PES de audio y PES de datos [22].

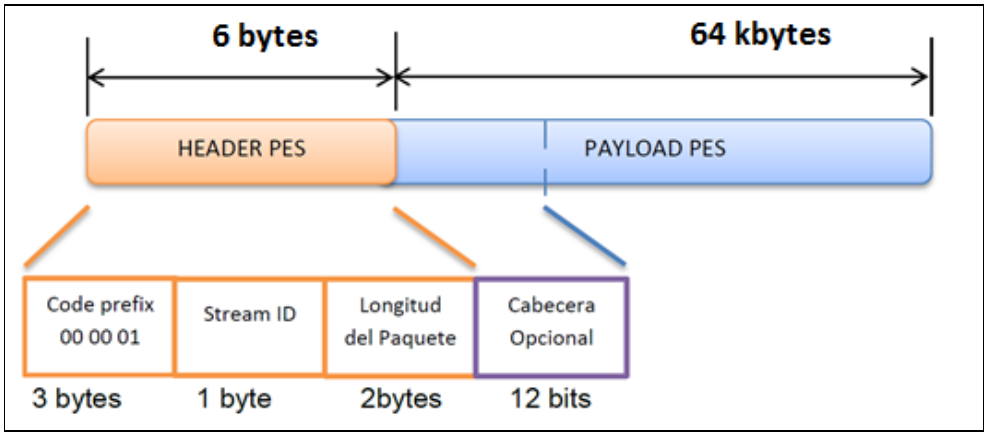

Figura 2.32. Estructura del paquete PES [25]

Los paquetes denominados PES están formados por dos secciones: cabecera y carga útil como lo muestra la Figura 2.32.

2.8.4.1 Cabecera

La cabecera se encarga de llevar información del tipo de datos que se transporta en cada paquete, además contiene información de temporización para que el decodificador sepa cuando decodificar y presentar la trama. La cabecera se encuentra constituida por tres partes como lo muestra en la Figura 2.32.

La primera es un prefijo de código el cual tiene como función la de identificar el comienzo de un paquete, es por ello que siempre tiene el valor de 00 00 01, la longitud del prefijo es de 3 bytes. La segunda parte nos sirve para identificar el tipo de contenido (audio, video o datos) lleva el paquete, a esta parte se la llama Stream ID. La tercera parte se encarga de establecer la dimensión del paquete, es por ello que se lo conoce como Longitud del Paquete.

2.8.4.2 Carga útil

En esta sección se encuentra la información de video, audio o datos que van a ser transmitidos dentro del paquete PES. Esta parte cuenta con una cabecera opcional en la cual posee varias *flags*.

2.8.5. Estructura y Generación del *Transport Stream* (TS)

El TS se encuentra definido como el protocolo de comunicación para audio, video y datos, el cual genera un flujo de datos que se adaptan para comunicar o almacenar uno o más programas de televisión junto con datos adicionales y su principal función es la de permitir la multiplexación síncrona de video, audio y datos [23].

2.8.5.1 Generación del *Transport Stream*

El *Transport Stream* es un flujo de transporte definido en el estándar MPEG-2, dicho flujo se encarga de multiplexar los contenidos de audio, video y datos, sincronizándolos de tal manera que se consiga un solo flujo de transporte. Este flujo de transporte se genera a partir de los procesos anteriormente señalados, como la codificación, paquetización (PES) y multiplexación como se muestra en la Figura 2.33.

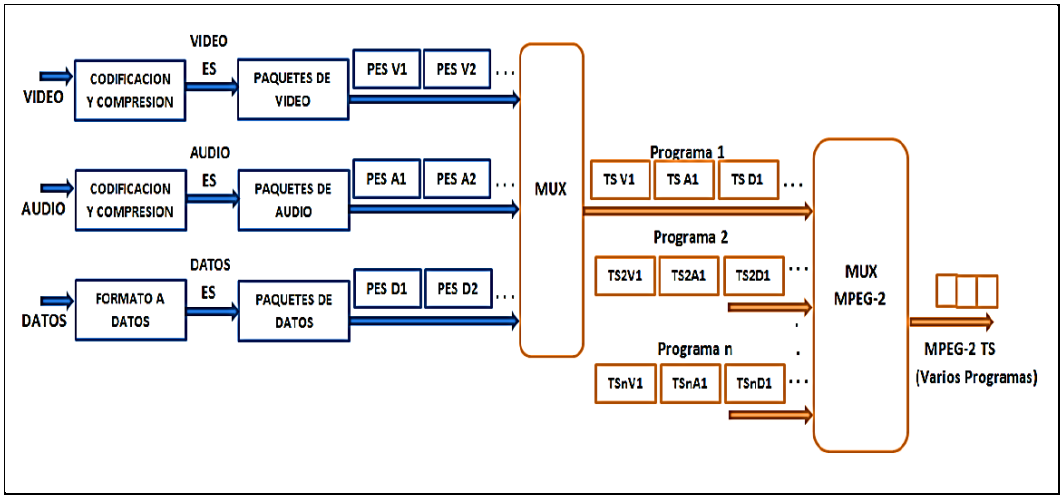

Figura 2.33. Generación de los Paquetes TS y Multiplexación [25]

Para la generación del TS lo que se hace en primer lugar es una etapa de codificación gestionando por separado tanto el audio, video y los datos para formar flujos de datos llamados ES, es decir, se da formato a los datos y se comprimen las señales de audio y video. Luego se emplea la técnica de paquetización, es decir, los *Elementary Streams* son almacenados en paquetes de tamaño variable llamados *Packetized Elementary Stream* (PES).

Finalmente se realiza un primer nivel de Multiplexación que combina el audio, el video, y datos en un solo flujo de paquetes de longitud constante de 188 bytes [22].

2.8.5.2 Estructura del *Transport Stream*

El TS posee una longitud constante de 188 bytes la cual se encuentra constituida por dos partes esenciales: la primera tiene una longitud de 184 bytes denominada *payload* o carga útil (contiene el audio, video y datos) y la segunda posee una longitud de 4 bytes y se le conoce como *header* o cabecera (contiene información de transmisión y sincronización de los paquetes).

Dentro de su estructurar consta de varias capas como lo indica la Figura 2.34, de tal manera que la capa inferior está formada por flujos de datos elementales (ES) de video, audio y datos.

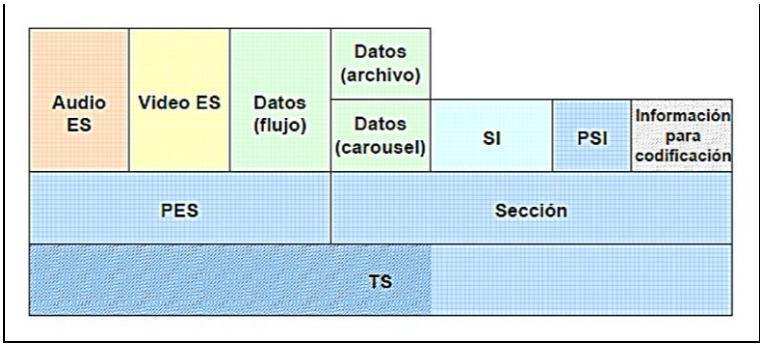

Figura 2.34. Estructura en capas del *Transport Stream* [25]

Una capa superior es la de paquetización donde todos los flujos elementales se almacenan en paquetes (PES). Otra capa contiene tanto las Tablas PSI/SI como los carruseles de datos y objetos. Finalmente la capa principal es la de TS ya que ésta contiene tanto los paquetes PES como el bloque de Sección, es decir, posee el video, audio, datos, y toda la información necesaria para su transmisión y recepción [22].

CAPÍTULO 3

DISEÑO E IMPLEMENTACION EN MATLAB® DEL DECODER DE VIDEO H.264/TS

3.1. Introducción

En este capítulo se hablará del diseño y la implementación del códec H.264 así como de su empaquetamiento TS, para llevar a cabo este proceso se utilizará MATLAB®, ya que es considerada una herramienta útil para tratamiento de imágenes y procesamiento digital de señales.

Al correr el aplicativo se pretende obtener una reducción significativa del tamaño del video procesado manteniendo una calidad de imagen superior a la que otros algoritmos proveen, es así que al obtener el archivo la diferencia en relación al original será mínima, dicha diferencia en este proyecto dependerá del factor de calidad (Q) que será ingresado en el programa.

Finalmente para que el resultado obtenido del códec H.264 esté listo para su transporte, en este caso un ambiente de TV Digital, se hará uso de un protocolo de comunicación para audio, video y datos como es el *Transport Stream* (TS).

3.2. Criterio de diseño

Puesto que un video es una secuencia de imágenes las cuales son representadas en forma de matrices, MATLAB® se vuelve en una de las principales herramientas elegidas para el desarrollo de este proyecto ya que se desenvuelve en un entorno matemático ideal para el procesamiento digital de imágenes.

Para la implementación del decoder de video H.264/TS en MATLAB® se consideró realizarlo basado en una interfaz gráfica GUI la cual posee un archivo .m y un archivo .fig que relacionados entre sí, permiten alcanzar el objetivo de este trabajo.

El programa se encuentra distribuido en tres partes esenciales: cargar, comprimir y empaquetar las mismas que aparecen secuencialmente en la interfaz mientras se va procesando el video original.

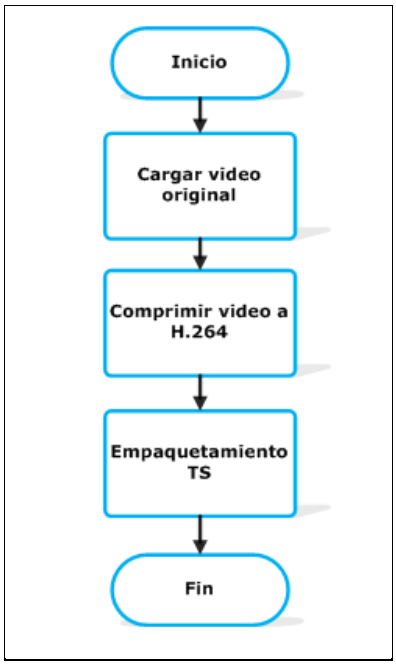

Figura 3.1. Diagrama de bloques del decoder de video H.264/TS

El programa implementado permite cargar un video en formatos definidos por las funciones de MATLAB® y convertirlo a un archivo de video comprimido MPEG-4 con codificación H.264, con una calidad de imagen variable que puede ser escogida por el usuario desde un rango de 0 a 100. De igual manera se utiliza el video comprimido para obtener adicionalmente un archivo video con un empaquetamiento TS.

3.3. Implementación del decoder de video H.264/TS

Como se mencionó anteriormente este trabajo consiste en una aplicación realizada en MATLAB® donde se usó funciones específicas para el tratamiento de video, las mismas que fueron enlazadas hacia el entorno GUI que posee esta herramienta, a través de estas funciones se hace posible procesos como la predicción que consiste en la interpolación y extrapolación de imágenes, la transformada que separa los componentes de baja y alta frecuencia donde luego cada entrada es redondeada con la cuantización mientras la codificación entrópica reduce la información asignando códigos a información que se repite de cada cuadro, los pasos explicados dan como resultado una compresión H.264. Al igual que en la codificación para llevar a cabo el empaquetamiento se usa una función desarrollada

en Linux la cual es llamada desde MATLAB® para tener como resultado el *Transport Stream* (TS).

3.3.1. Proceso de carga de video original

En la implementación se eligió un video que posee un formato MPEG que es tomado por MATLAB® y llevado a una memoria temporal, esta etapa es realizado por la función VideoReader.

3.3.1.1 VideoReader

Es una función que actúa como un lector multimedia, el cual lee datos de video desde un archivo en el espacio de trabajo de MATLAB ® y los almacena temporalmente en un objeto que tiene toda la información sobre dicho video [24].

Si el video no se encuentra en el directorio actual de trabajo es necesario escribir la ruta completa del archivo. Los formatos que puede leer esta función varían según el sistema operativo sobre el cual este soportado MATLAB®, estos formatos se detallan en la siguiente tabla.

Tabla 3.1. Formatos que soporta la función VideoReader [27]

Plataforma	Tipo de formato	Extensión
Todas las plataformas	AVI (incluyendo indexados, escala de grises) Motion JPEG 2000	(.avi) (.mj2)
Todos los Windows	MPEG-1 Windows Media® Video	(.mpg) (.wmv, .asf, .asx)
Windows 7 o posterior	MPEG-4 incluyendo videos codificados H.264 Apple QuickTime Movie	(.mp4, .m4v) (.mov)
Macintosh	Formatos soportado por QuickTime Player incluyendo: MPEG-1 MPEG-4 incluyendo videos codificados H.264 Apple QuickTime Movie	 (.mpg) (.mp4, .m4v) (.mov)
Linux®	Cualquier formato soportado por plug-ins instalados para GStreamer 0.10 o superior, incluyendo Ogg Theora	(.ogg)

Al ingresar la variable (Muestra.mpg) al VideoReader la función empieza a leer cada fotograma del video y crea una estructura de video con los fotogramas leídos, es decir, crea un objeto para lo cual se emplea la siguiente sintaxis:

obj = VideoReader (nombre de archivo)

Dónde:

nombre de archivo: especifica el archivo de video a leer el cual debe ser una cadena de caracteres entre comillas simples.

En este diseño para usar la función VideoReader se almacenó el video original (Muestra.mpg) en el directorio de trabajo de MATLAB®, el *path* de dicho video es localizado con la función ***uigetfile*** en la cual se especifica los formatos aceptados en este programa. En la variable ***direccion*** es almacenado el *path* y nombre del video con la función ***fullfile***, esta información es usada por el VideoReader para leer el archivo del video original y almacenarlo en un espacio temporal. La configuración de la sección explicada se observa en la Figura 3.2.

```
%% CARGAR EL VIDEO
%Permite seleccionar el video a cargar
[FileName, PathName]=uigetfile({'*.mpg';'*.avi';'*.mj2';'*.mp4';'m4v';
    '*.wmv';'asf';'*.asx';'*.mov'},'Seleccionar video');
%Manejador que almacena la ruta del video seleccionado
handles.direccion = fullfile(PathName,FileName);
%Actualizar la estructura del handles global
guidata(hObject, handles)
%Lee y almmacena dentro del objeto videoEntrada
videoEntrada = VideoReader(handles.direccion);
```

Figura 3.2. Proceso de cargar del video original

Para obtener las propiedades como por ejemplo la duración y ubicación del video almacenado se ejecuta el comando ***get***, su resultado se muestra en la Figura 3.3.

```
>> get(videoEntrada)
  General Settings:
    Duration = 4.7020
    Name = Muestra.mpg
    Path = D:\Mis Documentos\MATLAB\Tesis_H.264_TS
    Tag =
    Type = VideoReader
    UserData = []

  Video Settings:
    BitsPerPixel = 24
    FrameRate = 29.9700
    Height = 240
    NumberOfFrames = 141
    VideoFormat = RGB24
    Width = 320
```

Figura 3.3. Ejecución del comando *get*

Todas las propiedades que entrega la función VideoReader son sólo de lectura y se detallan en la Tabla 3.2.

Tabla 3.2. Propiedades de la función VideoReader [27]

Propiedades	Descripción	
Duration	Longitud total del archivo dada en segundos.	
Name	Nombre del archivo asociado con el objeto.	
Path	Contiene la ruta completa del archivo que se lee.	
Type	Nombre de la clase del objeto: 'VideoReader'.	
BitsPerPixel	Bits por píxel de los datos de video.	
FrameRate	Velocidad de fotogramas del video dado en fotogramas por segundo.	
Height	Altura del cuadro de video en píxeles.	
Width	Ancho del cuadro de video en píxeles.	
NumberOfFrames	Número total de fotogramas de la secuencia de video.	
VideoFormat	Indica la representación del formato de video.	
	Formato de video	**Valor**
	AVI o archivos MPEG-4	'RGB24'
	AVI con video indexado	"Indexado"
	AVI con video escala de grises	"Escala de grises"

Si se requiere de una propiedad específica, por ejemplo la duración, se utiliza la sentencia que se muestra en la Figura 3.4.

```
>> duracion_video = videoEntrada.duration

duracion_video =

    4.7020
```

Figura 3.4. Detalle de una propiedad específica del video

3.3.2. Proceso de compresión H.264 del video

Los códecs se basan en aprovechar modelos de predicción temporal y espacial, con lo que finalmente codifican toda la información utilizando códigos entrópicos, reduciendo el número de bits necesarios para codificar la información. Una vez que se obtiene la información del video comprimido, se necesita darle formato y ponerla en un contenedor para poder ser transportada y almacenada.

En este caso se tiene como contenedor al formato MP4 y como códec al H.264 definido por el estándar MPEG-4 Parte 10 AVC. Una vez adquirido el video de entrada, este es manipulado por la función VideoWriter el mismo que lleva a cabo la mayor parte de la compresión H.264.

3.3.2.1 VideoWriter

Es una función utilizada para la escritura de videos, es decir, toma un archivo de video que debe tener un tamaño máximo de 2GB el cual es almacenado en memoria temporal y de acuerdo a la compresión que en MATLAB® es denominado perfil construye el nuevo video [25].

Los perfiles que soporta para presentar el formato final junto con el códec utilizado se detallan en la Tabla 3.3.

Tabla 3.3. Perfiles de la función VideoWriter [28]

Perfil	Tipo de archivo
Archival	Video comprimido con el códec JPEG 2000
Motion JPEG AVI	Un archivo AVI con compresión Motion JPEG
Motion JPEG 2000	Video comprimido con el códec JPEG 2000
MPEG-4	Un archivo MPEG-4 con compresión H.264
Uncompressed AVI	Un archivo AVI con datos de video sin comprimir
Indexed AVI	Un archivo AVI con datos de video indexado
Grayscale AVI	Un AVI con datos de video en escala de grises
NOTA: La función VideoWriter si no se le asigna un perfil específico tiene por defecto el perfil denominado 'Motion JPEG AVI'.	

La sintaxis del VideoWriter se estructura de la siguiente manera:

$$objw = VideoWriter\ (\ nombre\ ,\ perfil\ )$$

Dónde:

nombre: es el archivo leído y debe ser una cadena de caracteres entre comillas simples.

perfil: hace referencia al formato y tipo de compresión, de igual manera este debe estar entre comillas simples.

En el caso de no especificar el perfil la función crea un archivo AVI comprimido Motion JPEG con la extensión (.avi) la misma que es usada por defecto.

Para el proceso de escritura VideoWriter se ayuda de funciones adicionales llamadas métodos descritos en la Tabla 3.4, los cuales en conjunto trabajan y hacen posible la obtención de un archivo que en este caso es un video comprimido con el códec de video H.264.

Tabla 3.4. Métodos para usar la función VideoWriter [28]

Métodos	Descripción
open	Abre el archivo para escribir datos del video.
writeVideo	Escribe los datos de video para presentar.
Close	Cierra el archivo después de escribir datos del video.
getProfiles	Lista de perfiles y formatos de archivos compatibles con VideoWriter.

En la Tabla 3.5 se observan las propiedades que entrega la función VideoWriter donde la mayoría son sólo de lectura pero *Quality* por ejemplo es un parámetro manipulable dentro del programa diseñado.

Tabla 3.5. Propiedades de la función VideoWriter [28]

Propiedades	Descripción	
Filename	Nombre del archivo asociado con el objeto	
Path	Contiene la ruta completa del archivo	
FileFormat	Tipo de archivo para escribir: 'avi', 'mp4', o 'mj2'	
Duration	Duración del archivo denotado en segundos	
ColorChannels	Número de canales de color en cada fotograma del video	
	Tipo de archivo	**Canales de color**
	AVI, MPEG-4 (con datos RGB24)	3
	AVI (indexados, escala de grises)	1
	Motion JPEG 2000	1 (datos en B/N) 3 (datos de color)
Height	Altura de cada fotograma de video en píxeles	
Width	Ancho de cada fotograma de video en píxeles	
FrameCount	Número de fotogramas grabados en el archivo de video	
FrameRate	Número de fotogramas por segundo del video. Por defecto: 30	
VideoBitsPerPixel	Número de bits por píxel en cada fotograma del video de salida	
VideoFormat	Representación MATLAB del formato de video	
	Formato de video	**Valor**
	AVI, MPEG-4 (datos RGB24)	'RGB24'
	AVI (video indexado)	"Indexado"
	AVI (video escala de grises)	“Escala de grises"
VideoCompressionMethod	Indica el tipo de compresión del video: "H.264", "Motion JPEG" o "Motion JPEG 2000"	
Quality	Valor entero de 0 a 100. Los números más altos resultan en una mayor calidad de video y archivos de mayor tamaño. Los números más bajos resultan en una menor calidad de video y archivos de menor tamaño	
Colormap	Matriz numérica que contiene información de color sobre el archivo de video	
CompressionRatio	Número mayor que 1 que especifica la relación objetivo entre el número de bytes en la imagen de entrada y el número de bytes en la imagen comprimida	
LosslessCompression	Valor booleano (lógico verdadero o falso)	

En la Figura 3.5 se presenta un ejemplo del VideoWriter donde se puede ver que la función se complementa con los métodos especificados en la Tabla 3.4.

```
>> obj=VideoReader('Muestra.mpg')
numFrames = obj.NumberOfFrames
objw=VideoWriter('Muestra_H264','MPEG-4')
open(objw)
for i=1:numFrames
    videoFrame = read(obj,i);
    writeVideo(objw,videoFrame);
end
close(objw)
```

Figura 3.5. Estructura general de la función VideoWriter

Haciendo uso de la función VideoWriter se ha producido un flujo de bits H.264 comprimido, dicha función se ayuda de otras funciones internas almacenadas en el *toolbox* de MATLAB® en la sección *audiovideo* que hacen posible el proceso del códec de video H.264 que se observa en la Figura 3.6.

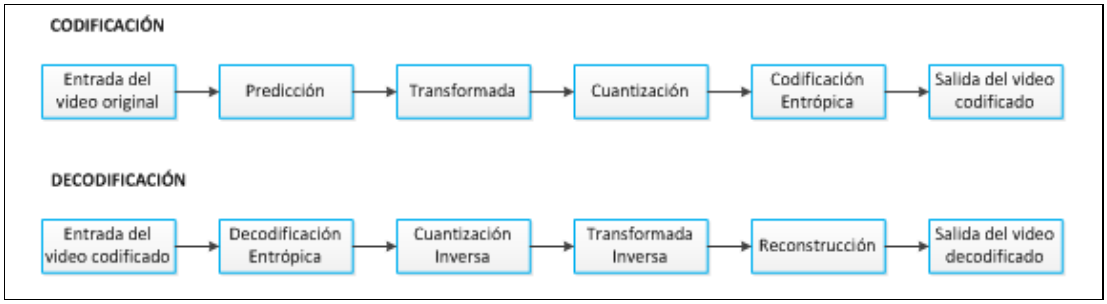

Figura 3.6. Diagrama de bloques del proceso del códec H.264

El proceso inicia con la predicción intra que toma el primer cuadro de la secuencia de video, dado que los valores de la imagen varían lentamente entre píxeles consecutivos se toma bloques de 8x8 y se aplica la transformada discreta del coseno (DCT) donde se obtiene una matriz cuyo primer coeficiente tiene el valor más alto y concentra la mayor parte de la energía importante para su visualización, dicho valor es el codificado mientras que los valores de amplitud cero no serán tomados en cuenta.

Cada uno de los 64 coeficientes obtenidos de la DCT se cuantifica uniformemente, es decir, se realiza una división de cada coeficiente DCT por su

coeficiente de cuantificación y su resultado es redondeado al entero más cercano con el propósito de descartar la información que no es visualmente relevante.

Sobre la matriz cuantizada se aplica la codificación de entropía donde cada valor de mayor frecuencia se le asigna un código de longitud variable, es decir, a los coeficientes que aparecen frecuentemente se les asigna códigos más cortos y a los menos frecuentes códigos más largos, con esto se reduce la información procesada para esta etapa se utilizó los códigos de Huffman.

Luego de procesar el primer cuadro el cual es usado como referencia para el proceso de predicción inter que divide los macrobloques del cuadro actual en trozos más pequeños, con el objetivo de mejorar la estimación de movimiento especialmente en aquellas zonas donde hay mucho movimiento, para ello se compara el bloque del cuadro actual con alguno o todos los posibles bloques de los cuadros que se tiene como referencia, donde se busca un área o bloque de la imagen de referencia (pasada o futura, la única condición es que haya sido codificada y transmitida previamente) que sea óptima para realizar la predicción del bloque actual este proceso se conoce como estimación de movimiento. La zona elegida se convierte en el predictor para el bloque actual y lo que se hace es restar al bloque actual el predictor. Así, tenemos un bloque residual este proceso es la compensación de movimiento. El bloque residual o también llamado vector movimiento al igual que el cuadro tipo intra son transformados, cuantizados, codificados y finalmente transmitidos.

Para cumplir con cada uno de los pasos explicados la función VideoWriter combina una serie de 47 scripts donde solo un grupo de ellos siguen el códec H.264, estos archivos son: *union.m*, *unique.m*, *ismember.m* los mismos que están almacenados en el siguiente path:

C:\Program Files\MATLAB\R2013b\toolbox\matlab\ops\@cell

3.3.2.1 union.m

Este script obtiene los valores combinados de dos o más matrices sin repeticiones ordenados ascendentemente y adicional arroja la posición de cada

componente, cuyo propósito del resultado se utiliza para comparar los bloques del cuadro actual con los bloques de referencia.

3.3.2.2 unique.m

Trabaja con una sola matriz donde devuelve los mismos valores pero sin repeticiones y ordenados, además realiza operaciones sobre la matriz.

3.3.2.3 ismember.m

Asigna códigos a los elementos comunes de dos matrices obteniendo como resultado una sola matriz codificada.

Los scripts restantes realizan las validaciones del perfil seleccionado (MPEG-4 con codificación H.264), a través de este proceso se prepara el contenedor para el video de salida donde se definen aspectos como las dimensiones de cada cuadro de la secuencia de video, bits por pixeles, extensión, tipo de formato, entre otros. Este resultado que se obtiene es una secuencia de imágenes, ya que durante la compresión se suprime el audio.

En la Figura 3.7 se muestra como la función VideoWriter es implementada en el programa, para lo cual primero es leído el ***videoEntrada*** y luego es procesado por una serie de scripts llamados por el VideoWriter.

```
%% COMPRESION H.264 DEL VIDEO ORIGINAL
%Lee y almmacena dentro del objeto videoEntrada
videoEntrada = VideoReader(handles.direccion);
%Se obtiene la ruta temp del video
handles.nomb_salida_h264 = ['C:\Windows\Temp\mivideo'
    num2str(randi(100,1))];
%Actualizar la estructura del handles global
guidata(hObject, handles)
%Creamos el objeto para guardar el video h.264
videoSalida = VideoWriter(handles.nomb_salida_h264,'MPEG-4');
```

Figura 3.7. Ejecución del VideoWriter

El parámetro de entrada utilizado es el factor de calidad (Q) que es un valor entero de 0 a 100, parámetro definido por MATLAB®, el mismo que ingresa como

una propiedad Quality del video de salida (***videoSalida***) que hace referencia a la calidad con la que se visualizará.

Seguidamente se utiliza el método open para abrir la estructura del objeto ***videoSalida*** e ir escribiendo *frame* a *frame* tomando características del video original (***videoEntrada***) pero con un contenido que difiere en un número menor de bits, esto se lo hace a través de un lazo cuyo límite es el número de total de *frames* del ***videoEntrada***, esta distribución se lo puede observar en la Figura 3.8.

```
%Obtenemos y almacenamos el # de frames del video de entrada
numFrames = videoEntrada.NumberOfFrames;
%Seteamos calidad
videoSalida.Quality = handles.factorq;
%Abrimos estructura objeto para poder escribir
open(videoSalida);
%Bucle donde escribe los frames del video de salida h.264
for i=1:numFrames
    videoFrame = read(videoEntrada,i);
    writeVideo(videoSalida,videoFrame);
end
%Cierra el objeto videoSalida
close(videoSalida);
```

Figura 3.8. Ejecución de los métodos del VideoWriter

Figura 3.9. Primer cuadro del video H.264

Al terminar de escribir los *frames* del video de salida se cierra la estructura del objeto ***videoSalida*** y el primer cuadro es mostrado en la interfaz GUI de la Figura 3.9, el cual es un indicador de que el video esta comprimido y listo para seguir con el proceso de empaquetamiento TS.

3.3.3. Proceso de empaquetamiento TS

Una vez que tenemos el video almacenado en un formato conocido (MP4), la aplicación desarrollada analiza y fragmenta el contenido para obtener el video TS, a través del uso del programa FFmpeg que es una herramienta desarrollada en Linux, la cual puede ser compilada en la mayoría de sistemas operativos incluyendo Windows y con el comando ***system*** es incorporada al entorno de programación de MATLAB®.

3.3.3.1 FFmpeg

El nombre FFmpeg proviene del grupo de estándares de video MPEG, junto con "FF" (*Fast Fordward*). Se define como una función destinada para el manejo multimedia de datos la cual puede grabar, convertir y hacer *streaming* de audio y video [26].

Para ello trae incluido una biblioteca de códecs que se muestran en la Tabla 3.6, todos los videos que usen estos códecs pueden ser leídos como un parámetro de entrada de esta función.

Tabla 3.6. Códecs de video soportados por FFmpeg [29]

Grupo	Nombre del formato
ISO/IEC/UIT-T	MPEG-1 parte 2, H.261, H.262/MPEG-2 parte 2, MPEG-4 parte 2, H.264/MPEG-4 AVC, HEVC/H.265, Motion JPEG
SMPTE	VC-1 (aka WMV3), VC-2 (aka Dirac), VC-1 (aka AVID DNxHD) y DPX
Microsoft	Microsoft RLE, Microsoft Video 1, Cinepak, Indeo (v2, v3 y v5), Microsoft MPEG-4 (v1, v2 y v3), Windows Media Video (WMV1, WMV2, WMV3/VC-1)
RealNetworks	RealVideo 1,2,3 y 4
QuickTime	Cinepack, ProRes, Sorenson 3 Codec, Animation Codec, QuickTime Graphics, Apple Video
Adobe Flash Player (SWF)	Screen video, Screen video 2, Sorenson 3 Codec, VP6 y Sorenso códec
Xiph.Org	Theora
On2/Google	Duck TrueMotion 1, Duck TrueMotion 2, VP3, VP5, VP6, VP7, VP8, VP9
RAD Game Tools	Smacker video y Bink video
RenderWare	TXD

El manejo se realiza por líneas de comando, una sintaxis simple del uso del FFmpeg para convertir un video de un formato a otro es la siguiente:

ffmpeg -i video_a_convertir.avi video_convertido.mpg

Dónde:

-i: hace referencia al video de entrada.

video_original.avi: nombre del video de entrada.

video_convertido.mpg: nombre del video de salida.

La función FFmpeg consta de varios atributos que facilitan la manipulación multimedia, cada uno de estos están explicados en la Tabla 3.7.

Tabla 3.7. Atributos de la función FFmpeg [29]

Atributo	Descripción
Ffmpeg	Herramienta de líneas de comandos para convertir audio y video de un formato a otro.
Ffserver	Servidor de *streaming* multimedia para transmisiones en vivo, soporta HTTP y RTSP.
Ffplay	Reproductor multimedia basado en SDL y en las bibliotecas FFmpeg.
Ffprobe	Herramienta de línea de comandos para mostrar información multimedia (texto, CSV , XML)
libswresample	Biblioteca que contiene rutinas de remuestreo de audio.
Libavresample	Contiene rutinas de remuestreo de audio desde el proyecto Libav, similar a libswresample de ffmpeg.
Libavcodec	Biblioteca que contiene todos los códecs de audio y de video.
Libavformat	Biblioteca que contiene los multiplexadores y demultiplexadores para los archivos contenedores multimedia.
Libavutil	Biblioteca de apoyo que contiene todas las rutinas comunes en las diferentes partes de FFmpeg.
Libpostproc	Biblioteca que contiene funciones de postproceso de video.
Libswscale	Biblioteca de escalado de video.
Libavfilter	Permite que el audio y video puedan ser modificados.

Dentro de la elaboración del programa se tienen parámetros de uso específico para obtener el flujo de bits TS, es decir, toma el video con códec H.264 y lo procesa de acuerdo a la selección puntualizada en la Tabla 3.8. Para acoplar el FFmpeg al programa implementado en MATLAB® se recurre al comando ***system*** que con una sintaxis especifica obtiene el video empaquetado con la extensión *.ts.

Tabla 3.8. Parámetros para empaquetamiento TS

Parámetro	Descripción
-i	Archivo fuente de video
input_file.mp4	Video original .mp4 con codificación H.264
-vcodec copy	Indica el códec de video a utilizar, al poner copy conserva el mismo que el video fuente
-acodec copy	Indica el códec de audio, en este caso en la etapa de codificación solo maneja video
-vbsf h264_mp4toannexb	Atributo para convertir un archivo MPEG-4 con codificación H.264 para flujos de transporte TS
output_file.ts	Video generado .ts

El video codificado ingresa a la función donde se especifica que conservará el mismo formato del video origen (MPEG-4 con codificación H.264) y formará pequeños paquetes de 188 bytes de los cuales 4 bytes son de inclusión obligatoria que ayudará a la construcción ordena de paquetes después de su transmisión.

De igual forma que en la etapa de compresión se hace uso del ***imshow*** para mostrar el primer *frame* TS en la interfaz GUI. La combinación de parámetros del FFmpeg para obtener el TS se los muestra en la Figura 3.10.

```
%% EMPAQUETAMIENTO .TS
%Empaquetamiento .TS ffmpeg
system(['ffmpeg\bin\ffmpeg.exe -i ' handles.fNamemp4
    ' -vcodec copy -acodec copy -vbsf h264_mp4toannexb '
    handles.nomb_salida_ts '.ts']);
%Actualizar la estructura del handles global
guidata(hObject, handles)
%Muestra primer frame en el axes 3
axes(handles.axes3)
imshow(handles.videoFrame);
```

Figura 3.10. Combinación de la función *system* con FFmpeg

En relación a lo explicado en la Figura 45 se muestra el diagrama de bloques que sigue el aplicativo decoder de video H.264/TS donde se encuentran especificados secuencialmente los pasos que sigue el video desde el momento que se carga, pasando por la compresión H.264 y finalmente se procesa para su empaquetamiento.

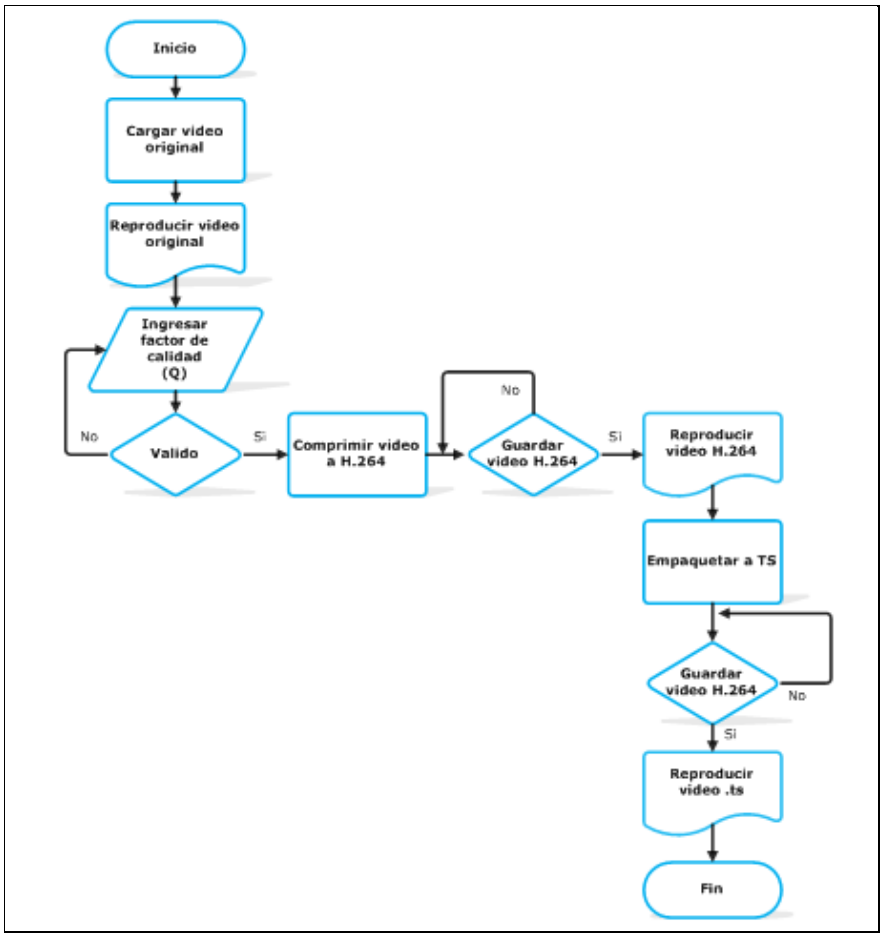

Figura 3.11. Diagrama de flujo del decoder de video H.264/TS

CAPÍTULO 4

FUNCIONAMIENTO Y RESULTADOS

4.1. Introducción

En esta sección se presentara los resultados obtenidos tras la ejecución del programa decoder de video H264/TS, tomando como referencia un video al cual se le realizara el proceso de compresión H.264 y empaquetamiento TS variando el factor de calidad (Q) y que mediante el uso de métodos objetivos y subjetivos que evalúan la calidad de los videos comprimidos se determina el valor de Q más óptimo.

4.2. Funcionamiento del aplicativo

Al correr el programa del decoder de video H.264/TS se muestra la interfaz GUI de MATLAB® como se puede apreciar en la Figura 4.1.

Figura 4.1. Interfaz GUI del Decoder de Video H.264/TS

Al inicializarse la interfaz se procede a cargar el video original el mismo que puede tener los formatos mencionados en la Tabla 3.1 definidos por las funciones de MATLAB®.

En este caso se utilizó el video ***Muestra.mpg*** cuyas características se pueden observar en la Tabla 4.1.

Tabla 4.1. Características del video Muestra.mpg

TAMAÑO DEL ARCHIVO	645 kb
Característica	**Descripción**
AUDIO	
Códec	MPEG Audio layer 1/2/3 (mpga)
Canales	Estéreo
Tasa de muestra	44100 Hz
Tasa de bits	64 kb/s
VIDEO	
Códec	MPEG-1/2 Video (mpegv)
Resolución	320x240
Tasa de fotogramas	29.970029
Formato decodificado	Planar 4:2:0 YUV

Al terminar el proceso para cargar el video la interfaz muestra el path donde se encuentra el video original, en este caso nos indica la siguiente:

D:\Users\Usuarios\Documents\MATLAB\Tesis_H.264_TS\Muestra.mpg

Después para seguir con el proceso de compresión de video H.264 se debe ingresar el parámetro de calidad Q el cual puede ser un valor entre 0 y 100. A continuación presionando el botón *Comprimir video a H.264* comprime el video el mismo que se lo debe almacenar en un nuevo directorio con la opción *Guardar video H.264*, así mismo se puede visualizar el resultado al seleccionar *Reproducir video H.264*.

En la Figura 4.2 se presenta las opciones para comprimir el video.

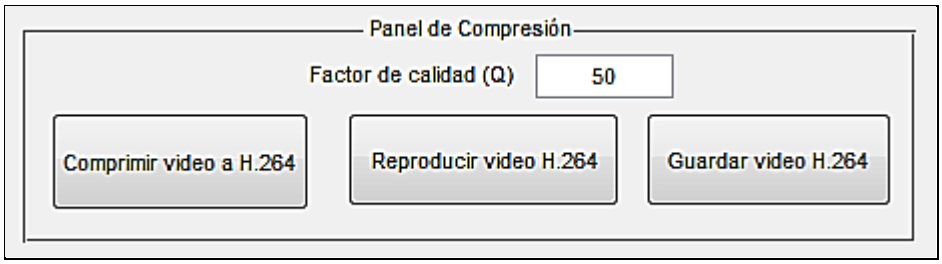

Figura 4.2. Opciones de la aplicación para la compresión H.264

Como último paso se tiene el empaquetamiento que toma el video comprimido con la opción *Empaquetamiento *.TS*, de igual manera la salida de este proceso debe ser almacenado (*Guardar video *.TS*) para poder reproducirlo y apreciar el resultado del nuevo video con respecto al original.

La Figura 4.3 se muestra las opciones para el empaquetamiento del video.

Figura 4.3. Opciones de la aplicación para el empaquetamiento TS

4.3. Pruebas de funcionamiento del Decoder de video H.264/TS

Para probar que el algoritmo implementado funciona de acuerdo a las definiciones planteadas en el marco teórico se ha utilizado el software *VLC Media Player* el cual permite apreciar las características que poseen tanto el video original, comprimido y empaquetado.

La Figura 4.4 muestra la las características del video original el cual ahora está haciendo uso del códec H.264, de la misma forma en la Figura 4.5 se observa cómo el video que ahora tiene como códec H.264 es empaquetado con el protocolo de transporte TS.

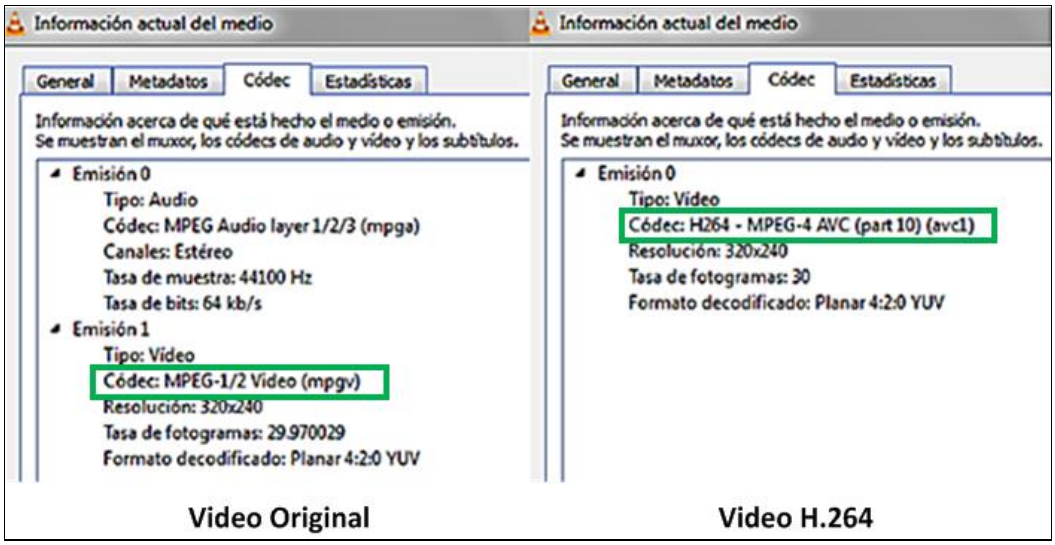

Figura 4.4. Resultado obtenido al ejecutar la compresión H.264

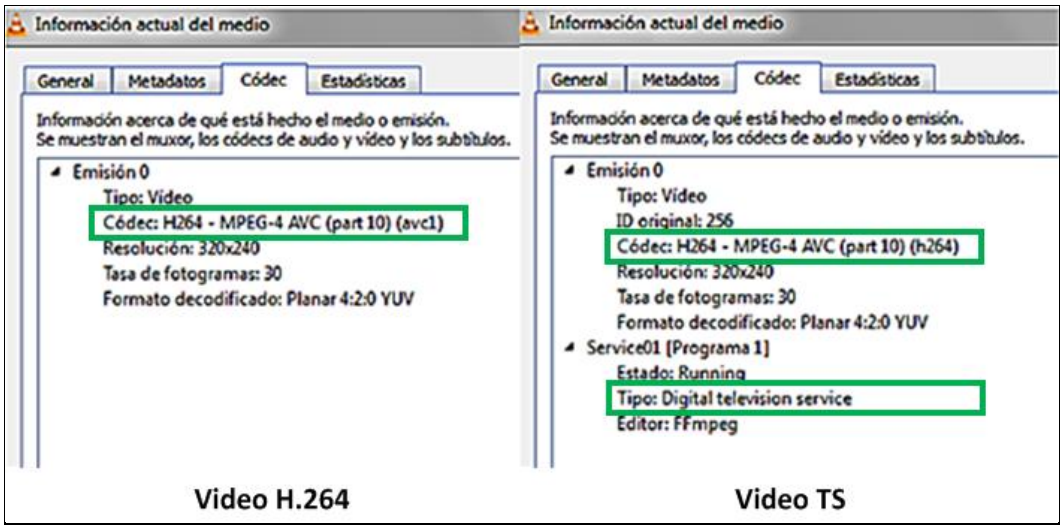

Figura 4.5. Resultado del video empaquetado TS

Otro software que se utilizó para probar el empaquetamiento de video es el analizador de TS propiedad de la ESPE mostrados en la Figura 4.6 y 4.7 donde se visualiza que cumple los parámetros para el *Transport Stream*.

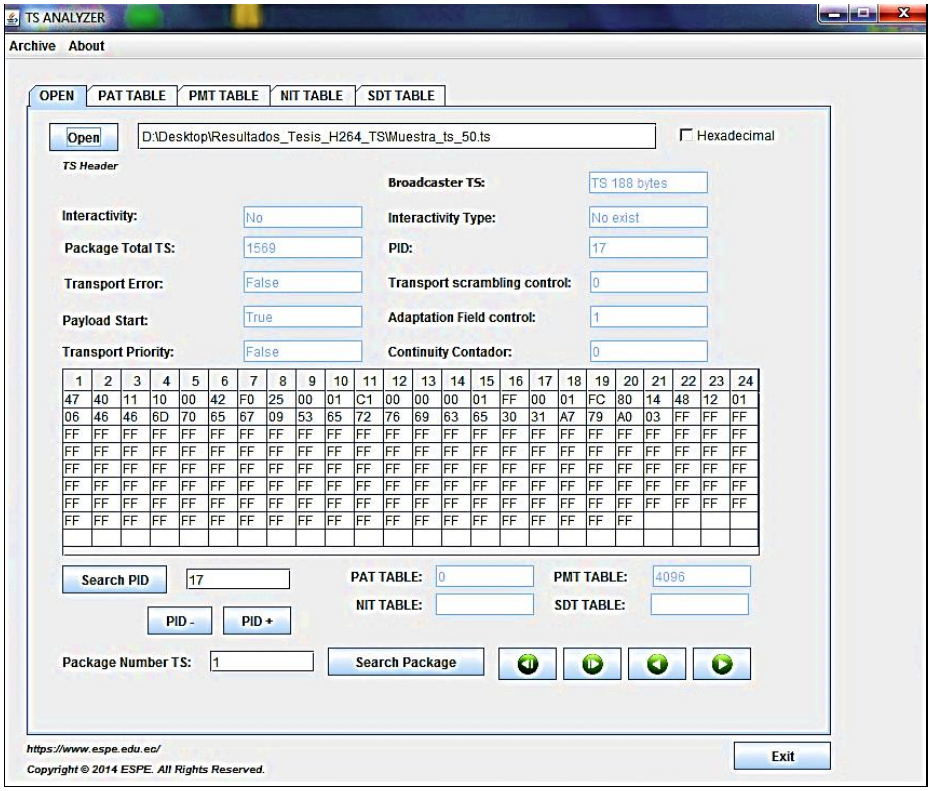

Figura 4.6. Resultados con el TS *Analyzer*

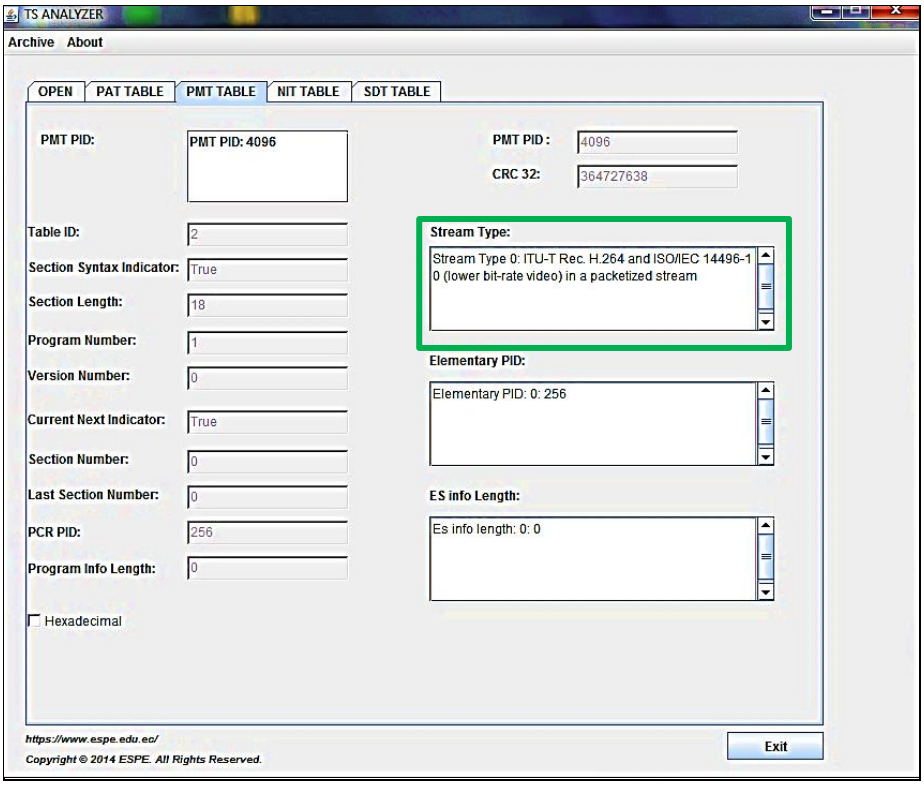

Figura 4.7. Validación del empaquetamiento

4.4. Análisis de resultados obtenidos con el códec H.264

En una compresión de video digital cualquiera que sea el método aplicado, existen varios tipos de distorsiones o degradaciones en la secuencia de imágenes, esto nos permite medir la calidad del video que es un parámetro fundamental dentro del análisis de resultados de este documento.

De acuerdo al estándar del *Video Quality Experts Group* (VQEG) para la evaluación de la calidad de video se tomaron técnicas objetivas y subjetivas.

4.4.1. Método objetivo

Permite que se puedan medir con fiabilidad la calidad percibida en base al cálculo del MSE y PSNR obteniendo así el análisis del comportamiento del video.

Como se mencionó en secciones anteriores en el aplicativo desarrollado se manipula el valor Q que define la calidad del video comprimido y es propio de la herramienta MATLAB® para el tratamiento multimedia. Por ello se tomó cuatro valores de Q (0, 50, 75, 100) con los cuales se calcularon los valores de error

cuadrático medio MSE y la relación de señal a ruido de pico PSNR que están definidas por las ecuaciones (4.1) y (4.2), estas métricas requieren de la referencia completa del video original para ser calculadas.

$$\mathrm{MSE} = \frac{1}{\mathrm{MN}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(C_{ij} - R_{ij}\right)^2 \tag{4.1}$$

$$\mathrm{PSNR} = 10 * \log_{10}\left(\frac{L^2}{MSE}\right) \tag{4.2}$$

Donde M, N representan el alto y ancho en pixeles de la secuencia de imágenes, C, R son la imagen original y la distorsionada respectivamente y L toma el valor de 255 ya que toma 8 bits por píxel.

Para determinar los datos de MSE se utilizó la herramienta MSU *Video Quality Measurement Tool Free* que es un programa para la evaluación objetiva de la calidad de video, donde se ingresa el video de referencia y el video comprimido los que son comparados por áreas específicas.

En la Tabla 4.2 y la Figura 4.9 se muestra los resultados de la compresión H.264 con los distintos valores de Q, el MSE y el cálculo del PSNR.

Los valores típicos de PSNR para imágenes con pérdida y compresiones de video son entre 30 y 50 dB, donde a más alto, mejor. Cuando las dos imágenes o videos son idénticos, el MSE será cero. Para este valor, el PSNR está indefinido [27].

Tabla 4.2. Tamaño del video comprimido en relación al video original

Nombre del archivo	Factor Q	Tamaño del video [kb]	MSE	PSNR [dB]
Muestra.mpeg	-	645	-	-
Muestra_H264_0.mp4	0	35	144,33	26,54
Muestra_H264_50.mp4	50	256	46,08	31,50
Muestra_H264_75.mp4	75	393	111,43	27,66
Muestra_H264_100.mp4	100	1.165	34,67	32,73

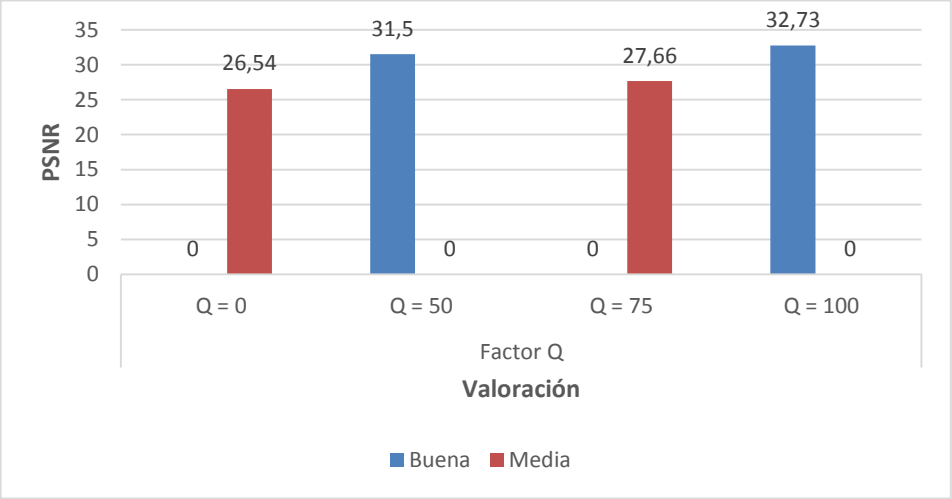

Figura 4.8. Resultados de MSE y PSNR

De acuerdo al método objetivo analizado los valores aceptables de PSNR para la calidad de video son 31,50 y 32,73 correspondiente a un Q de 50 y 100 respectivamente.

Debido a que para un Q = 100 se tiene un tamaño de casi el doble del video original, se concluye que para un factor Q = 50 proporciona una buena calidad de imagen y una reducción de tamaño considerable la misma que es necesaria para que cumpla con los estándares de Televisión Digital.

4.4.2. Método subjetivo

La manera más confiable de medir la calidad de una imagen o un video es la evaluación subjetiva, realizada por un conjunto de personas que opinan acerca de su percepción. La opinión media, obtenida en forma directa mediante el *Mean Opinion Score* (MOS) son métricas generalmente aceptadas como medida de la calidad. Diversos métodos subjetivos de evaluación de video son reconocidos, y están estandarizados en las recomendaciones ITU-R BT.500-11 desarrolladas para aplicaciones de televisión y la ITU-T P.910 para aplicaciones multimedia.

En todos los métodos propuestos, los evaluadores son individuos que juzgan la calidad en base a su propia percepción y experiencia previa. El MOS es una prueba subjetiva basada en una progresión de 1 a 5, donde 1 representa la peor calidad y 5 la mejor. En la Tabla 4.3 se observa cómo se valora la calidad según la medida MOS.

Tabla 4.3. Valoración de MOS

Escala	Calidad	Deterioro
5	Excelente	Imperceptible
4	Buena	Perceptible pero tolerable
3	Media	Ligeramente molesto
2	Pobre	Molesto
1	Mala	Muy molesto

Este tipo de pruebas son óptimas para el usuario pero se debe sustentar estos resultados con los valores objetivos hallados (PSNR) y realizar una conversión al MOS como se muestra en la Tabla 4.4, es así que para el PSNR de 31,50 (Q=50) y 32,73 (Q=100) sus valores MOS están en la escala de cuatro, es decir, son Buenos.

Tabla 4.4. Conversión PSNR - MOS

PSNR [dB]	MOS
< 20	1
21 – 25	2
26 – 30	3
30 – 40	4
> 40	5

En base a lo explicado anteriormente se presentaron cuatro videos H.264 con factor Q de 0, 50, 75 y 100 a diez personas que no poseen mayor conocimiento de la teoría de compresión de video, donde se evaluó la calidad del video en base a la precepción del usuario esto es: Excelente, Buena, Media, Pobre y Mala.

Los resultados de estas pruebas se presentan en la Tabla 4.5.

Tabla 4.5. Resultados pruebas MOS

	Factor Q			
Valoración	Q = 0	Q = 50	Q = 75	Q = 100
Excelente	0	1	2	2
Buena	0	9	6	8
Media	1	0	2	0
Pobre	7	0	0	0
Mala	2	0	0	0

De los datos obtenidos para un Q igual 0 al apreciar el video siete personas indicaron que la calidad era pobre, es decir, comentaron que existen distorsiones en varias partes de la secuencia de imágenes que se muestra en la Figura 4.9.

Figura 4.9. Video H.264 con Q igual a 0

Con un valor de Q igual a 50 nueve persona afirmaron que la calidad es buena y las imágenes se muestran con mejor nitidez, esto se lo puede validar en la Figura 4.10.

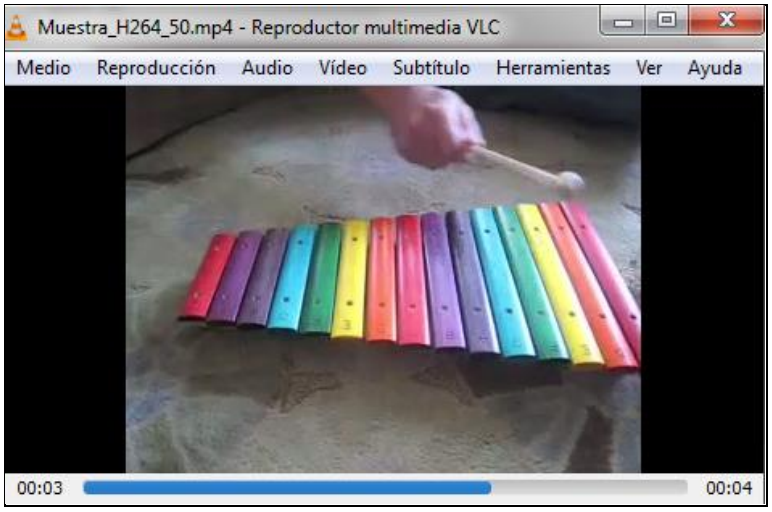

Figura 4.10. Video H.264 con Q igual a 50

Finalmente para los valores de Q igual a 75, 100 entre seis y ocho personas respectivamente señalaron que los videos eran buenos, en las Figuras 4.11 y 4.12 se presentan una muestra de estos videos.

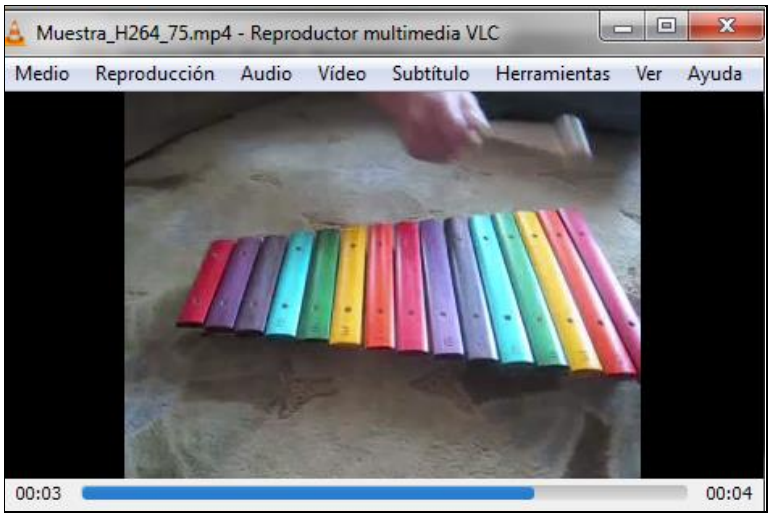

Figura 4.11. Video H.264 con Q igual a 75

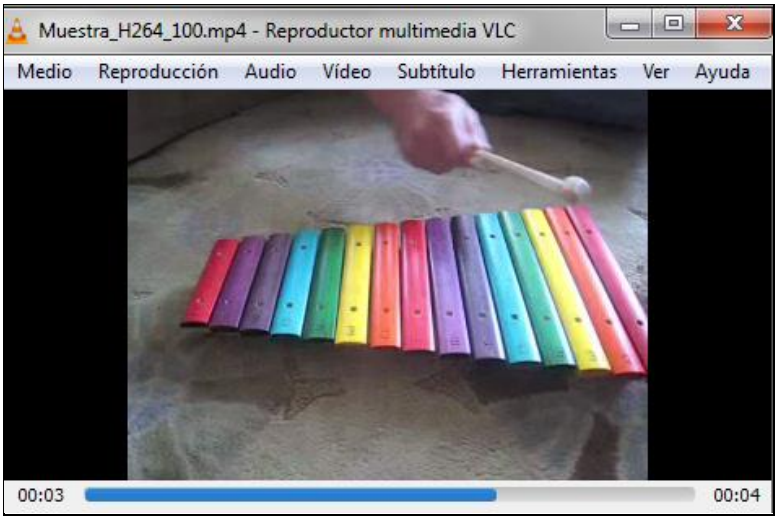

Figura 4.12. Video H.264 con Q igual a 100

Luego de analizar las evaluaciones de los cuatro videos comprimidos expuestos a un total de diez personas en la Figura 4.13 se puede observar claramente que el video es bueno para Q igual a 50 y 100, esto quiere decir que estos valores proporcionan una buena calidad de imagen a diferencia de Q igual a 0 y 75.

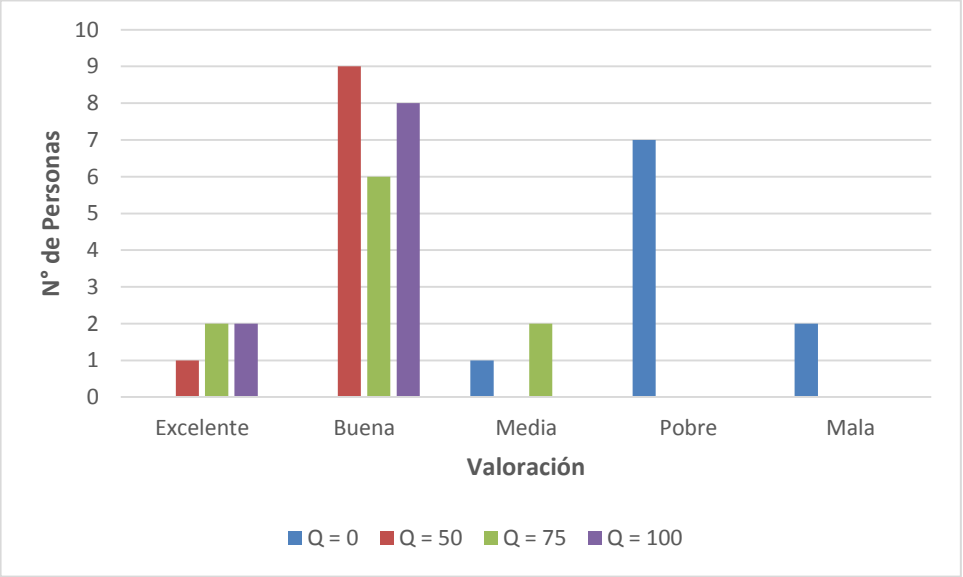

Figura 4.13. Resultados MOS

4.4.3. Análisis final de resultados

De acuerdo a los resultados arrojados por los métodos de evaluación de calidad tanto objetivos como subjetivos, se tiene que para los valores de Q igual a 50 y 100 la calidad del video es Buena ya que presentan PSNRs mayores de 31,50 y 32,73 dB respectivamente, esto quiere decir que tienen menores pérdidas las cuales son poco notorias para la percepción del usuario final.

En vista de que para los dos valores de Q la calidad de video es buena, se tiene que para Q igual a 100 el tamaño es mucho mayor en relación al video original, por lo que se concluye que el factor de calidad ideal es Q igual a 50 ya que proporciona una buena calidad de imagen y una reducción de tamaño de casi la mitad con respeto a su entrada.

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

Luego de un estudio minucioso del estándar H.264/AVC y empaquetamiento TS se presenta el diseño e implementación de un aplicativo con las características y funcionalidades que se plantearon al comienzo del proyecto con el cual se obtiene una compresión y calidad favorable para la posterior transmisión de un video procesado por esta aplicación.

Se ha comprobado que una de las principales ventajas de H.264 es la capacidad de predicción de imágenes mejorada ya que analiza secciones de bloques mucho más pequeños en relación a otros estándares anteriores lo cual permite obtener videos de muy buena calidad y menor bitrate afirmando así el por qué ha recibido buena aceptación entre el mundo del video digital.

Al usar el *Transport Stream* se verifica que se acopla de una manera óptima con el códec de video H.264 ya que el resultado de este no varía significativamente con respecto al video que sale del decodificador, por lo tanto se demuestra lo definido por el estándar de TV Digital ISDB-Tb.

Uno de los procesos que marcan los beneficios del algoritmo implementado es la estimación y compensación de movimiento los mismos que son ejecutados a través de operaciones como la transformada, cuantización y codificación entrópica dando como resultado una precisión de imagen propicia y como consecuencia se requiere de mayor procesamiento del hardware usado.

Como resultado del análisis de compresión de video basados en métodos objetivos y subjetivos se concluye que el factor de calidad ideal es Q igual a 50 ya que proporciona una buena calidad de imagen y una reducción de tamaño de casi la mitad con respeto a su entrada la misma que es necesaria para la división de tramas que realiza el *Transport Stream* TS.

En este trabajo se presentaron conceptos e informaciones necesarias tanto para compresión de video como para su empaquetamiento, volviéndose una referencia importante para facilitar futuros desarrollos basados en la programación del estándar.

5.2. Recomendaciones

Para el adecuado funcionamiento y utilización de la aplicación decoder de video H.264/TS es fundamental realizar una correcta compilación de FFmpeg en el proceso de instalación, ya que esto podría causar errores al momento de invocar a esta biblioteca. Con la finalidad de evitar posibles inconvenientes es recomendable seguir a detalle cada uno de los pasos descritos en la Guía de compilación FFmpeg, misma que se encuentra en la página oficial de la herramienta.

REFERENCIAS BIBLIOGRÁFICAS

[1] C. Pérez Vega y J. M. Zamanillo Sainz de la Maza, Fundamentos de Televisión Analógica y Digital, Santander: Servicios de Publicaciones de la Universidad de Cantabria, 2003.

[2] G. Flores Chapa, «Conteo de objetos en flujos de video H.264,» Noviembre 2011.

[En línea]. Available: http://mcyti.izt.uam.mx/archivos/Tesis/Generacion2009/ICR_GustavoFlores.pdf.

[3] ISO/IEC, «Information technology - Coding of audio-visual objects - Part 10: Advanced Video Coding,» Diciembre 2005.

[En línea]. Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=56538.

[4] I. Richardson, The H.264 Advanced Video Compression Standard, Second ed., Chichester: John Wiley and Sons, Ltd., 2010.

[5] T. Wiegand, G. J. Sullivan, G. Bjntegaard y A. Luthra, Overview of the H.264/AVC Video Coding Standard, vol. 13, IEEE Transaction on Circuits Systems for Video Technology, 2003, pp. 560-576.

[6] MPEG, «Advanced Video Coding,» Enero 2014.

[En línea]. Available: http://mpeg.chiariglione.org/standards/mpeg-4/advanced-video-coding.

[7] M. Rabbani y P. Jones, Técnicas de la compresión de la Imagen Digital, Bellingman: Prensa Óptica de la Ingeniería de SPIE, 1991.

[8] A. MacAulay, B. Felts y Y. Fisher, «White Paper – IP Streaming of MPEG-4: Native RTP vs MPEG-2 Transport Stream,» Octubre 2005.

[En línea]. Available: http://www.envivio.com/files/white-papers/RTPvsTS-v4.pdf.

[9] M. S. Olivé, «Algoritmo de compresión de imágenes de alta resolución sin pérdidas,» Septiembre 2008.

[En línea]. Available: http://tesis.bnct.ipn.mx/bitstream/handle/123456789/5745/ALGORITMODECOMPR.pdf.

[10] INTPLUS, «Compresion Multiple H.264/MPEG4/MJPEG,» Diciembre 2014.

[En línea]. Available: http://www.superinventos.com/S120515.htm.

[11] W. Simpson, Video Over IP, Burlington: Focal Press, 2008.

[12] F. Pescador Del Oso, «Contribución a las mitologías de optimización del tiempo de ejecución de algoritmos de decodificación de video sobre DSPs,» Julio 2011.

[En línea]. Available: http://oa.upm.es/8712/1/FERNANDO_PESCADOR_DEL_OSO.pdf.

[13] B. Duque y C. Tipantuña, «Estudio, análisis e implementación en software de simulación MATLAB,» Diciembre 2010.

[En línea]. Available: http://bibdigital.epn.edu.ec/handle/15000/2619.

[14] D. Huffman, «A method for the construction of minimum redundancy codes,» *Publicación de Proceedings of the IRE*, pp. 1098-1099, 1952.

[15] Varios, «Algoritmos de compresión y efecto de la compresión en el tamaño de imágenes, audio y video.,» Enero 2013.

[En línea]. Available: http://www.sistmult.wordpress.com/page/2/.

[16] M. d. C. Gómez Cruz, «Estudio del streaming de audio y vídeo sobre redes heterogéneas,» Noviembre 2010.

[En línea]. Available: http://e-archivo.uc3m.es/handle/10016/8706.

[17] L. V. Agostini, «Desenvolvimento de Arquiteturas de Alto Desempenho Dedicadas à Compressão de Vídeo Segundo o Padrão H.264/AVC,» Agosto 2007.

[En línea]. Available: http://www.lume.ufrgs.br/bitstream/handle/10183/12425/000627573.pdf?sequence=1.

[18] D. A. Jines Espín, «Compresión de archivos de video según el estándar H.264 para el mejoramiento de la velocidad de transmisión en tiempo real del sistema de videovigilancia de global compu.,» Julio 2012.

[En línea]. Available: http://repo.uta.edu.ec/handle/123456789/2372.

[19] D. Salomon, Data Compression. Chapter 6, Video Compression, London: Springer-Verlang, 2007.

[20] R. Vega Alonso, Implementacion en Java/C++ del estándar de codificación H.264, Madrid, 2009.

[21] Multimedia Signal Processing Group, «Video coding with H.264/AVC: Tools, Performance and Complexity,» Marzo 2015.

[En línea]. Available: http://www.img.lx.it.pt/~fp/artigos/H264_final.pdf.

[22] E. Aguilar Fernández, «Decodificador de video MPEG-2 en Matlab y análisis del bitstream,» Junio 2008.

[En línea]. Available: http://bibing.us.es/proyectos/abreproy/11618/fichero/MEMORIA%252F0+-+Portada.pdf.

[23] E. C. Villegas Castillo, «Diseño de una arquitectura para la interpolación de quarter-pixel para estimación de movimiento según el formato H.264/AVC empleado en el estándar SBTVD de televisión digital terrestre.,» Agosto 2011.

[En línea]. Available: http://tesis.pucp.edu.pe/repositorio/handle/123456789/869.

[24] A. A. Morales Figueroa, «Diseño de la red para interactividad en Televisión Digital Terrestre e IPTV en el campus ESPE Sangolquí,» Junio 2010.

[En línea]. Available: http://repositorio.espe.edu.ec/bitstream/21000/2645/1/T-ESPE-029824.pdf.

[25] A. G. Gutiérrez Tapia y M. Á. Cochancela Alvear, «Diseño de un laboratorio de televisión digital para la transmisión de señales con multiprogramación, contenidos interactivos y guía electrónica de programación (EPG),» Abril 2013.

[En línea]. Available: http://dspace.ucuenca.edu.ec/handle/123456789/372.

[26] M. A. Illescas Robalino y D. F. Villamarín Zapata, «Implementación de un transmisor de pruebas de TV digital terrestre ISDB-Tb, para la emisión de aplicaciones interactivas,» Junio 2011.

[En línea]. Available: http://repositorio.espe.edu.ec/handle/21000/4289.

[27] MathWorks, «VideoReader class,» Enero 2015.

[En línea]. Available: http://es.mathworks.com/help/matlab/ref/videoreader-class.html?refresh=true.

[28] MathWorks, «VideoWriter class,» Enero 2015.

[En línea]. Available: http://es.mathworks.com/help/matlab/ref/videowriter-class.html.

[29] FFmpeg, 30 Enero 2015. [En línea]. Available: https://www.ffmpeg.org.

[30] C. D. Segura Villalobos, «Diseño e implementación de una metodología para la medición de QoS/QoE en servicios OTT montados sobre una plataforma LTE/IMS,» Agosto 2012.

[En línea]. Available: http://www.academia.edu/9815022/DISE%C3%91O_E_IMPLEMENTACI%C3%93N_DE_UNA_METODOLOG%C3%8DA_PARA_LA_MEDICI%C3%93N_DE_QoS_QoE_EN_SERVICIOS_OTT_MONTADOS_SOBRE_UNA_PLATAFORMA_LTE_IMS_MEMORIA_PARA_OPTAR_AL_T%C3%8DTULO_DE_INGENIERO_CIVIL_ELECTRICISTA_CRISTI%C3%8.

ACTA DE ENTREGA

El proyecto de grado titulado "DESARROLLO DE UN DECODER DE VIDEO H.264/TS UTILIZANDO LA PLATAFORMA DE MATLAB®" fue entregado en el Departamento de Eléctrica y Electrónica y reposa en la Universidad de las Fuerzas Armadas – ESPE desde:

Sangolquí, a 29 de Abril de 2015

ELABORADO POR:

Edison Fernando Tenorio Moya

C.I.: 172186553-1

AUTORIDAD:

Ing. Paúl Bernal MSc.

DIRECTOR DE LA CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES