

# **ESCUELA POLITÉCNICA NACIONAL**

## **ESCUELA DE POSGRADO EN INGENIERÍA Y CIENCIAS**

### **IMPLEMENTACIÓN DE UN SISTEMA PILOTO DE TELE- PRESENCIA (VIDEO CONFERENCIA) PARA BANDA ANCHA.**

**TESIS PREVIA A LA OBTENCIÓN DEL GRADO DE MAGÍSTER (Msc) EN  
INGENIERÍA ELÉCTRICA, MENCIÓN CONECTIVIDAD Y REDES DE  
TELECOMUNICACIONES**

**GUSTAVO ADOLFO SALVADOR SALVADOR**

**DIRECTOR: Msc. Ramiro Morejón**

**Quito, Mayo de 2007**

## DECLARACIÓN

Yo, Gustavo Adolfo Salvador declaro que la presente Tesis de Grado es de mi autoría, y delego todos los derechos de autor de la misma a la Escuela Politécnica Nacional.,

---

Gustavo A. Salvador S.

## CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Gustavo Adolfo Salvador Salvador, bajo mi supervisión.

---

Msc. Ramiro Morejón

DIRECTOR DE PROYECTO

## DEDICATORIA

La presente Tesis de Grado la dedico a mis padres, por su constante apoyo, y a quienes me han brindado siempre su afecto, apoyo y su comprensión, mi muy amada esposa Kathy y la pequeña Luz que Dios puso en nuestras vidas, nuestra hijita Eli.

## CONTENIDO

CAPÍTULO 1 .....	1
1.1 INTRODUCCION .....	1
CAPÍTULO 2 .....	7
2.1 REQUERIMIENTOS DEL SISTEMA .....	7
2.1.1 El codec de video .....	7
2.1.2 El codec de audio .....	40
2.1.3 La captura del video .....	41
2.1.4 La captura del audio .....	43
2.1.5 El canal de datos.....	43
CAPITULO 3 .....	46
3.1 DISEÑO DEL SISTEMA.....	46
3.1.1 LAS HERRAMIENTAS DE DESARROLLO .....	46
3.1.2 EL ESQUEMA GENERAL DE LA APLICACIÓN DE VIDEOCONFERENCIA.....	48
3.1.3 LA INICIALIZACIÓN DEL SISTEMA .....	48
3.1.3 LA ESTRUCTURA GENERAL DEL SISTEMA.....	50
3.1.4 EL CODEC DE VIDEO .....	51
3.1.5 EL CODEC DE AUDIO.....	56
3.1.6 LA CAPTURA DEL VIDEO.....	57
3.1.7 LA CAPTURA DEL AUDIO .....	58
3.1.8 EL CANAL DE DATOS .....	58
3.1.10 LA APLICACIÓN DE PRESENTACIÓN REMOTA.....	60
CAPITULO 4 .....	70
4.1 EL SISTEMA DE VIDEO CONFERENCIA.....	70
4.2 EL SISTEMA DE PRESENTACIÓN REMOTA .....	76
4.2.1 EL LADO SERVIDOR .....	76
4.2.2 EL LADO CLIENTE.....	84
CAPITULO 5 .....	86
5.1 PRUEBAS Y RESULTADOS.....	86
5.1.1 CONEXIÓN ENTRE REDES LAN EMPLEANDO RUTEADORES Y ADMINISTRANDO LA VELOCIDAD DEL CANAL .....	86
5.1.2 CONEXIÓN ENTRE EQUIPOS EN UNA RED WAN .....	87
5.1.3 CONEXIÓN EN UNA RED LAN A TRAVÉS DE EQUIPOS INALÁMBRICOS .....	87
5.1.4 CONCLUSIONES Y RECOMENDACIONES.....	88
5.1.5 LO QUE SE PUEDE MEJORAR DEL SISTEMA .....	89
REFERENCIAS BIBLIOGRÁFICAS.....	90
ANEXOS.....	91

## FIGURAS

Figura 1.1. Prototipo de 1968 de video conferencia.	1
Figura 1.2. PicturePhone de AT & T	2
Figura 1.3. Sistema de video conferencia Isabel 8	4
Figura 2.1 Diagrama de crominancia xy CIE 1931 en el que se muestra la gama del espacio de color definido por Adobe y la localización de los colores primarios. El punto blanco D65 se muestra en el centro.	8
Figura 2.2 Diagrama de crominancia xy CIE 1931 en el que se muestra la gama del espacio de color definido por sRGB. El punto blanco D65 se muestra en las coordenadas [0.3127, 0.3290].	8
Figura 2.3 A) Matriz representando los componentes de color rojo, verde y azul de un bloque de píxeles (celdas de imagen). B) La misma matriz con un sub-muestreo 4:4:4, esto es, sin pérdida. C) La misma matriz pero con un sub-muestreo 4:2:2, en el que se puede observar que ha ocurrido cierta compresión.	10
Figura 2.4 A) La imagen que se desea muestrear B) El muestreo de la luminancia. C) El muestreo de la crominancia diferencial azul. D) El muestreo de la crominancia diferencial roja.	11
Figura 2.5 Diagrama resumido del codec de video especificado en la recomendación ITU H.261	13
Figura 2.6 Posición de las muestras de luminancia y crominancia en 4:2:2	14
Figura 2.7 Respuesta del ojo con respuesta a las frecuencias temporal (movimiento) y espacial (resolución).	15
Figura 2.8 La persistencia de la visión humana. A) Un objeto pasa rápidamente causando borramiento por su movimiento. B) El mismo objeto es seguido por el movimiento del ojo evitando el borramiento.	16
Figura 2.9 Se produce el efecto de aliasing pero no es percibido debido al moviendo del ojo al tratar de seguir al objeto en la pantalla.	17
Figura 2.10 Interpolación proyectada paralela al movimiento.	18
Figura 2.11 A) El algoritmo espera encontrar un bloque similar en la siguiente imagen en la misma posición. B) Como no ha sido posible encontrar el bloque, este se mueve por el área de búsqueda.	20
Figura 2.12 Correlación Espacial Jerárquica.	20
Figura 2.13 Comparación del volumen de cálculos.	21
Figura 2.14 Sistema de fase lineal.	22
Figura 2.15 Correlacionador de fase básico.	23
Figura 2.16 Transformada inversa muestra desplazamientos. A) No hubo movimiento. B) Movimiento a la derecha. C) Movimiento a derecha e izquierda	24
Figura 2.17 Superficie de correlación.	25
Figura 2.18 Un cuadro subdividido en bloques.	26
Figura 2.19 Diagrama en bloques de un estimador de movimiento por desplazamiento de fases.	28
Figura 2.20 Superficie de correlación vs. identificación de movimiento.	29
Figura 2.21 Asignación de vectores.	30
Figura 2.22 El codificador de fuente según ITU H.261.	31
Figura 2.23 Bloque de 8x8 luminancia / crominancia de una imagen.	32

Figura 2.24 Bloque de luminancia / crominancia restado 128.	33
Figura 2.25 Bloque con los índices de la DCT.	33
Figura 2.26 Matriz de cuantificación.	34
Figura 2.27 Bloque DCT luego de aplicar la matriz de cuantificación.	34
Figura 2.28 Codificación DPCM	35
Figura 2.29. Resumen de la codificación Intra de imágenes.	36
Figura 2.30 Orden de los elementos de la imagen en “zig-zag”	36
Figura 2.31 Los cuadros tipo P	37
Figura 2.32 Los cuadros tipo B	38
Figura 2.33 Resumen de la codificación Inter de imágenes.	38
Figura 2.34 Diagrama de bloques del codificador.	40
Figura 2.35 Diagrama de bloques del decodificador.	40
Figura 3.1 El sistema de video conferencia en bloques	48
Figura 3.2 Esquema general de inicio de procesos	49
Figura 3.3 Estructura general del sistema	50
Figura 3.4 Diseño general del sistema de video	52
Figura 3.5 Clases generales del sistema de video	53
Figura 3.6 Diagrama de bloques del codificador	54
Figura 3.7 Algoritmo para obtener la DCT	56
Figura 3.8 Las componentes armónicas de una señal cuadrada	57
Figura 3.9 La estructura del canal de datos	59
Figura 3.10 La aplicación de presentación remota en bloques	60
Figura 3.11. Los procesos de inicio del lado servidor	61
Figura 3.12. Las clases de la aplicación del lado servidor	62
Figura 3.13. El procedimiento de creación de una nueva presentación	63
Figura 3.14. La operación de creación de una nueva presentación	63
Figura 3.15. El establecimiento de la conexión.	64
Figura 3.16. El control de transmisión.	65
Figura 3.17. La operación con las diapositivas.	66
Figura 3.18. Detalle de la operación con las diapositivas.	66
Figura 3.19. Las clases de la aplicación cliente.	67
Figura 3.20. Establecimiento de la conexión en el lado cliente.	67
Figura 3.21. La operación de Transmisión – Recepción.	68
Figura 3.22. La operación con las diapositivas en el lado cliente.	68
Figura 3.23. El procesamiento de las señales de control.	69
Figura 4.1 La ventana de dialogo principal.	71
Figura 4.2. El menú desplegable.	71
Figura 4.3. La ventana de dialogo de conexión.	73
Figura 4.4. La confirmación de conexión.	73
Figura 4.5. La ventana de dialogo de calibración de los niveles de audio.	74
Figura 4.6. La ventana Acerca De	75
Figura 4.7. El formulario principal	76
Figura 4.8. La ventana de dialogo guardar como.	79
Figura 4.9. La ventana de dialogo buscar carpeta.	80
Figura 4.10. El formulario de progreso.	81
Figura 4.11. La ventana de dialogo abrir una presentación existente.	82

Figura 4.12. La ventana de dialogo acerca de.	83
Figura 4.13. La ventana de dialogo de solicitud de conexión.	84
Figura 4.14. La ventana principal de la aplicación cliente.	84

## **TABLAS**

Tabla 2.1 Coeficientes de Huffman	35
Tabla 2.2 Número de píxeles por línea y número de líneas de cada formato de imagen H.263 normalizado.	39
Tabla 4.1. Descripción del menú de la ventana de dialogo principal.	72
Tabla 4.2. El menú desplegable del formulario principal.	78

## RESUMEN

La presente Tesis de Grado está compuesta de cinco capítulos:

En el primer capítulo tenemos una introducción a los protocolos que emplearemos en el desarrollo del proyecto.

En el segundo capítulo se analizan los requerimientos que tiene el proyecto para su operación, en lo referente al funcionamiento y las matemáticas necesarias para la implementación del codec de video, el codec de audio, el protocolo de transporte y finalmente la captura del audio y del video.

En el tercer capítulo se realiza el diseño de la aplicación, es decir se define como se construirá la aplicación a partir de diagramas de bloques hasta la definición de las clases que constituirán la aplicación.

En el cuarto capítulo se muestra de forma general la construcción de las aplicaciones (se diseñó e implementó adicionalmente una aplicación de envío – recepción de diapositivas + pizarra virtual).

En el quinto capítulo se analizan las pruebas que se realizaron a las aplicaciones y los resultados obtenidos de las mismas.

## **PRESENTACIÓN**

El presente proyecto de Tesis de Grado pretende implementar un sistema de video conferencia, con menos requerimientos de hardware, más económico, bajo la plataforma más comúnmente empleada en el país y más fácil de usar que las aplicaciones disponibles en el mercado, de modo que ponga la video conferencia al alcance de la mayoría de personas. También pretende ser una introducción al desarrollo de esta tecnología en el país, puesto que es muy susceptible de ser mejorada pese a las dificultades técnicas que esto implica.

ANEXO D: FORMATO PARA ORDEN DE ENCUADERNACIÓN

UNIDAD ACADÉMICA: \_\_\_\_\_

ORDEN DE ENCUADERNADO

---

---

Se emite la presente orden de encuadernación con fecha: \_\_\_\_\_

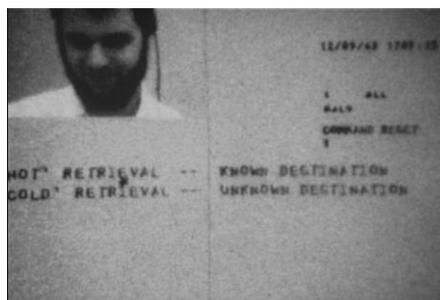
Para constancia firman los miembros del Tribunal Examinador:

# CAPÍTULO 1

## 1.1 INTRODUCCION

### 1.1.1 LA EVOLUCIÓN DE LOS PROTOCOLOS DE TRANSMISION CODIFICADA BAJO IP

La idea de una video conferencia, es tan antigua como la televisión, en un primer lugar se la realizaba de forma análoga mediante dos circuitos cerrados de televisión. Luego como ha sido usual los viajes espaciales incentivaron el mejoramiento de la tecnología, en este caso la NASA empleó dos enlaces de radio frecuencia (UHF o VHF), uno en cada dirección para enlazar la nave con el centro de control.



**Figura 1.1. Prototipo de 1968 de video conferencia.**

Luego vinieron los enlaces satelitales, empleando camiones especiales, los cuales contenían toda una infraestructura de radio frecuencia para establecer enlaces móviles. Toda esta tecnología resultaba demasiado onerosa para aplicaciones más mundanas como telemedicina, educación a distancia o encuentros de negocios, lo que impulso el uso de las redes telefónicas para transmitir video. Los primeros intentos fracasaron debido a la baja calidad de las imágenes y a no tener disponible una técnica de compresión de video eficiente, tal fue el caso del

Picturephone de A&T, que empleaba un ancho de banda de 1 MHz y una velocidad de 6 Mbps.<sup>1</sup>



**Figura 1.2. PicturePhone de AT & T <sup>1</sup>**

Fue en 1980 que las redes de telefonía digital se hicieron posibles con el advenimiento de ISDN, sin embargo, la video conferencia continuaba siendo cara y los equipos que se requerían eran propietarios. Fue hasta 1990 con la llegada de la videoconferencia empleando IP (protocolo H.320), que comenzó a hacer más asequible esta tecnología.

En 1991 PictureTel e IBM demostraron de forma práctica por primera vez la posibilidad de realizar una video conferencia empleando computadores personales.<sup>2</sup> Posteriormente en Diciembre de 1991 Apple, lanzo un conjunto de codecs de video con su QuickTime®. Luego en 1992, en la universidad de Cornell, Tim Torcey, desarrollo CU-SeeMe, un sistema de video conferencia cuya plataforma era Apple Macintosh®, seguidamente apareció en el mismo año Video For Windows de Microsoft <sup>3</sup>(Apple posteriormente demandó tanto a Microsoft como a Intel por haber plagiado parte de su código al “accesar” a su contratista, Canyon, con quien rescindió el contrato. El asunto se arregló posteriormente con una inversión de 150 millones de dólares de parte de Microsoft en Apple, así como el acuerdo de desarrollar Office e Internet Explorer para Macintosh<sup>4</sup>). Posteriormente salieron los protocolos H.261 en Marzo de 1993, seguido del protocolo H.262 en Febrero del 2000, a continuación en Enero de 2005 salió el

<sup>1</sup> [http://en.wikipedia.org/wiki/Video\\_conferencing](http://en.wikipedia.org/wiki/Video_conferencing)

<sup>2</sup> <http://video.ifas.ufl.edu/VC-History.htm>

<sup>3</sup> <http://en.wikipedia.org/wiki/Quicktime>

<sup>4</sup> <http://www.roughlydrafted.com/RD/RDM.Tech.Q1.07/592FE887-5CA1-4F30-BD62-407362B533B9.html>

protocolo H.263. Todos ellos relacionados con desarrollo de aplicaciones de videoconferencia.

La cual se hizo muy popular primero con aplicaciones como NetMeeting®, lanzada en Agosto de 1996, por Microsoft, la que hasta la actualidad emplea el protocolo H.261 para la implementación de la video conferencia, esta aplicación fue en un inicio licenciada y actualmente liberada con la salida de Windows XP®. Así mismo Video For Windows evolucionó hasta convertirse en la actualidad en Direct Show® por una parte y Windows Media® por otra parte (Microsoft fue también acusado de plagiar código de los codecs MP3 a Alcatel – Lucent, y deberá pagar 1.5 billones de dólares y tiene pendiente un juicio por plagiar código de compresión del sistema de reconocimiento de voz, también de Alcatel – Lucent<sup>5</sup>). Otra aplicación muy conocida que emplea video conferencia punto a punto es Skype (Que ha sido llevado a los tribunales con una demanda de \$4.1 billones de dólares por haber plagiado código de la conexión punto a punto (que es lo que le ha hecho tan famoso), de StreamCast Networks, así también Net2Phone ha demandado a Skype por haber infringido la patente 6,108,704 de los Estados Unidos de América, también relacionada con la conexión punto a punto<sup>6</sup>).

Por otra parte han surgido sistemas de video conferencia cuyos equipos son sumamente caros, ya que se trata de hardware propietario implementado protocolos como el H.323, como por ejemplo los equipos de Polycom® o Sony®. Así como también ha surgido código abierto, que proporciona simplemente voz sobre IP como OpenH323<sup>7</sup> o un sistema de video conferencia completo y sumamente complejo, útil más en aplicaciones educativas a nivel superior, en las cuales se tiene a disponibilidad una mayor infraestructura de hardware, como es el caso de Isabel<sup>8</sup>.

---

<sup>5</sup> <http://www.networkworld.com/news/2007/030207-one-alcatel-lucent-claim-against-microsoft.html>

<sup>6</sup> <http://en.wikipedia.org/wiki/Skype>

<sup>7</sup> <http://www.openh323.org/>

<sup>8</sup> <http://isabel.dit.upm.es/>



**Figura 1.3. Sistema de video conferencia Isabel<sup>8</sup>**

Actualmente la evolución de los protocolos ha llegado hasta el H.264, que fue publicado en Septiembre del 2005; iChat® de Apple hace uso de esta tecnología, aplicación que se encuentra incluida en su sistema operativo OS X®.

### **1.1.2 ¿CUÁLES SON LOS PRINCIPALES PROTOCOLOS DE CODIFICACIÓN DE LINEA?, Y ¿CÓMO FUNCIONAN?**

Realmente como tal la única recomendación que define un protocolo o procedimiento para la codificación de información de audio y video es ITU H.262, la cual depende de H.261 para lo referente a la codificación del video, la cual a su vez hace referencia a la recomendación BT601-5 en lo referente a la definición del espacio de color de las imágenes, y a T.81 en lo referente a la codificación de imágenes fijas. H.263 define simplemente adicionales respecto al manejo de la información tales como el control de errores, basándose en la recomendación H.262 para todo lo demás. Sin embargo H.263 junto con H.264 son conocidas como las recomendaciones para video conferencia.

En lo referente a como este protocolo funciona, el detalle esta en la lectura de la recomendación, pero básicamente nos habla del manejo de la imagen en cuadros, como se puede comprimir esta información, así como también nos habla de la compresión de la información de cada cuadro, además de definir la trama de información que deberá ser transmitida para que en el decodificador se pueda recuperar la información.

Finalmente tenemos a la recomendación G.711 que trata sobre el manejo de la compresión de una señal de audio.

### **1.1.3 ¿EN QUÉ CONSISTE LA APLICACIÓN QUE SE LES DARÁ A ESTOS PROTOCOLOS EN LA PRESENTE TESIS DE GRADO?**

La presente Tesis de Grado, utilizará la recomendación BT601-5, en lo referente a la transformación de los espacios de color desde RGB, que es el formato original de una imagen de computador a  $Y C_R C_B$ , también conocido como YUV, que es un sistema de espacios de color más útil para tratar la información de la imagen. Por otra parte también se emplea H.262, también conocido como MPEG-2 y sus recomendaciones referidas como T.81, en lo referente al manejo de macro bloques, a la compresión y descompresión de la información de un cuadro, y en parte a la forma que se crean los vectores de movimiento para la estimación de movimiento y la predicción de cuadros.

Por otra parte emplea G.711, referida en H.262 y H.263 como el protocolo para la compresión de audio.

Finalmente empleamos parte de H.263 para la definición de la trama de transporte de datos, ya que nosotros pretendemos dar cierta seguridad (evitar que la señal sea capturada por personas no autorizadas) a la transmisión sin la necesidad de emplear protocolos de transmisión codificada como SSL, los cuales requerirían una validación de una casa verificadora como VeriSign, lo cual haría que el software comenzara a tener un costo más o menos elevado, debido al valor de la licencia que estas casas emiten.

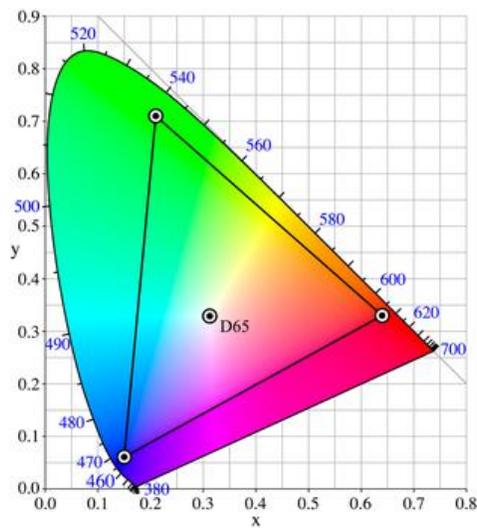
## CAPÍTULO 2

### 2.1 REQUERIMIENTOS DEL SISTEMA

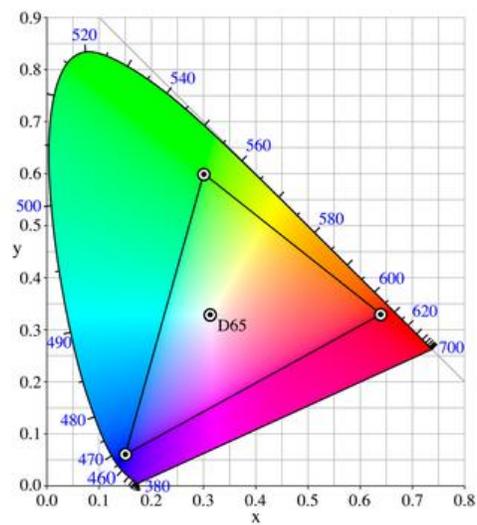
#### 2.1.1 El codec de video

Para poder definir el codec de video que empleará la presente Tesis de Grado es necesario primero comprender que implica la recomendación H.263 y sus recomendaciones relativas comenzando por la recomendación ITU BT.601-5, anteriormente CCIR-601.

Un *modelo de color* es una forma matemática de representar los colores como tuplas de 2 o 3 elementos, como es el caso del modelo RGB (Red, Green, Blue), que permite cuantificar los valores de rojo, verde y azul que tiene un color, este es el formato original en el cual se maneja el color en los computadores. En este caso, no se crea un mapa de color con estos valores, si se deseara hacerlo, se tendría que acomodar cada componente de color en un eje cartesiano, siendo posible ubicar cada posible color dentro de este *espacio de color*. Como se podría pensar, es posible asignar cada color a un eje diferente cada vez que se cree el espacio de color, es por esto que se requiere de estándares, y por ello ha surgido el CIEXYZ, pero además de este espacio de color existen otros talvez más comercialmente conocidos como el de Adobe, y el de Microsoft – Hewlett Packard (sRGB), este último es especialmente importante porque es el que se emplea en los monitores y pantallas de equipos portátiles, como cámaras fotográficas (transformada desde su forma RGB original).



**Figura 2.1 Diagrama de crominancia xy CIE 1931 en el que se muestra la gama del espacio de color definido por Adobe y la localización de los colores primarios. El punto blanco D65 se muestra en el centro.<sup>9</sup>**



**Figura 2.2 Diagrama de crominancia xy CIE 1931 en el que se muestra la gama del espacio de color definido por sRGB. El punto blanco D65 se muestra en las coordenadas [0.3127, 0.3290].<sup>10</sup>**

<sup>9</sup> [http://en.wikipedia.org/wiki/CIE\\_1931\\_color\\_space](http://en.wikipedia.org/wiki/CIE_1931_color_space)

<sup>10</sup> [http://en.wikipedia.org/wiki/SRGB\\_color\\_space](http://en.wikipedia.org/wiki/SRGB_color_space)

Como se puede observar hay una diferencia en la gama de colores entre ambos espacios de color, especialmente en lo que se refiere al azul, en el caso de sRGB, la gama de azules es más pequeña, por lo que cuando se requiere expresar los colores para impresión la gama de Adobe es la que comúnmente se prefiere.

Es también interesante mencionar algunos modelos de color existentes:

CMYK: Por cian, magenta, amarillo y negro en inglés, es un modelo de color substractivo muy empleado en todo lo relacionado a la impresión.

RGB: Por rojo, verde, y azul en inglés, es un modelo de color aditivo.

HSV: Por color, saturación y valor (brillo) en inglés, más empleado por los artistas.

HSL: Por color, saturación y luminancia en inglés, similar al anterior.

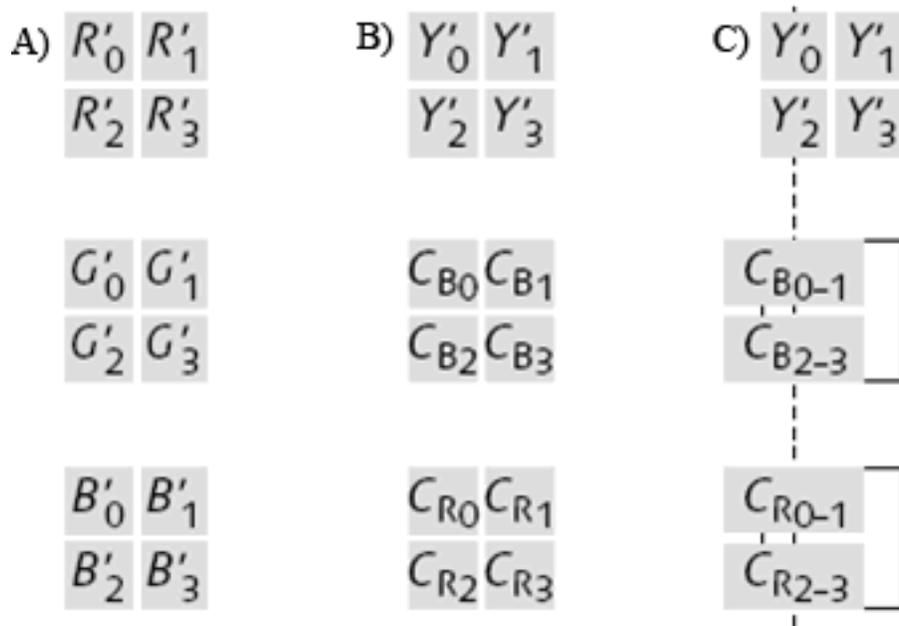
Una vez que se ha comprendido lo que es un espacio de color, se requiere mencionar que el ojo humano es más sensible a la luminancia que a la crominancia de una imagen, esto permite crear una forma de representar el color, como lo define la recomendación ITU BT601-5, esta define un espacio de color en la que la luminancia y la diferencia de los colores básicos azul y rojo, definen un color, esta definición permite realizar cierta compresión en lo que se refiere a la compresión de una imagen, sin que exista una diferencia notoria con la imagen original, al menos para el ojo humano.

Para comprender mejor como funciona esta compresión es necesario aclarar en primer lugar lo que es el *sub-muestreo de crominancia*. A esta se la define como la práctica de implementar más resolución (la cantidad representada) para la luminancia que para el color (la cantidad representada).

Como se mencionó anteriormente, se puede engañar al ojo humano al dar más valor a la luminancia que al color, reduciendo el ancho de banda que se requiere para transmitir la señal, por esto se muestrea de forma diferente la imagen para obtener la información digital de la luminancia que para obtener la información digital de la diferencia de color tanto azul como rojo, normalmente se muestrea la información de la luminancia a una frecuencia mayor que las otras dos, obteniendo con ello mayor información de la luminancia y perdiendo algo de información del color. Resultando en una imagen que contiene menos información que la original, pero para el ojo humano esta pérdida es difícilmente perceptible.

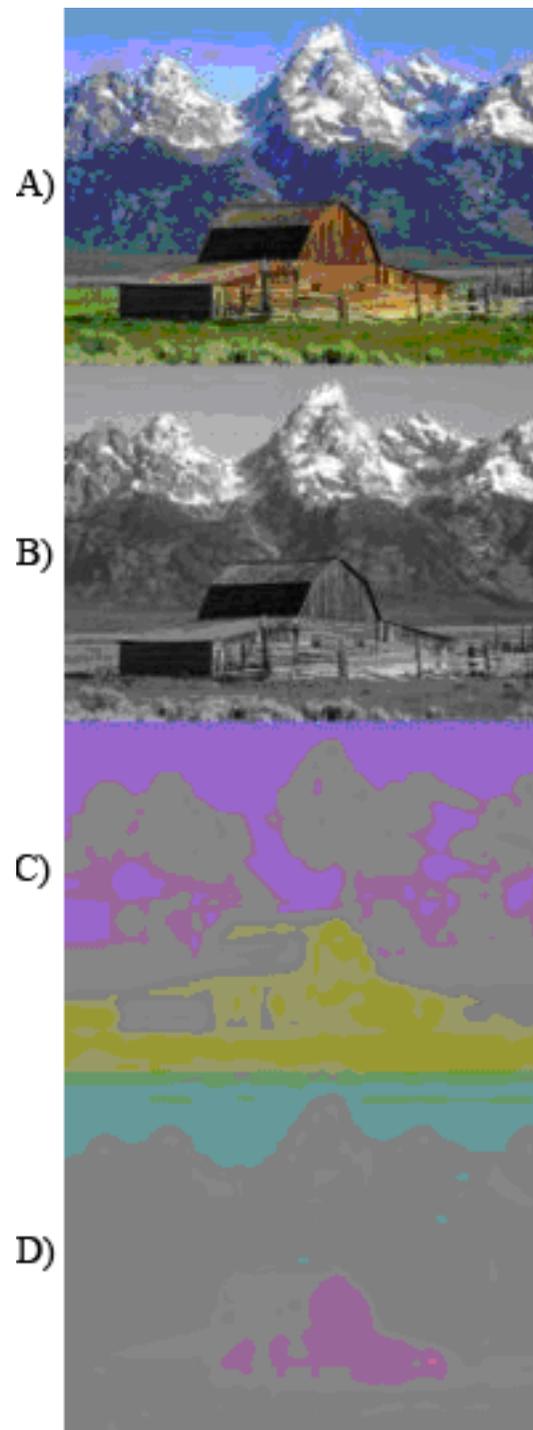
El sub-muestreo normalmente se lo expresa mediante tres cifras separadas por “:”, como por ejemplo 4:2:2. La primera cifra indica la frecuencia de muestreo, que suele ser múltiplo de  $3 \frac{3}{8}$  de Mhz, en el ejemplo es  $4 \times (3 \frac{3}{8})$ , la segunda cifra se refiere a la frecuencia que es muestreada la diferencia de color rojo, con respecto a la frecuencia de la luminancia (horizontal), en este caso nos dice que se muestrea a la mitad de la frecuencia original. Finalmente la tercera cifra nos indica algo similar a lo anterior pero referente al muestro vertical.

Volviendo al espacio de color que se mencionó anteriormente, se lo denomina  $Y C_R C_B$ , dónde Y representa la luminancia,  $C_R$ ,  $C_B$  representan las diferencias de color tanto roja como azul.



**Figura 2.3 A) Matriz representando los componentes de color rojo, verde y azul de un bloque de píxeles (celdas de imagen). B) La misma matriz con un sub-muestreo 4:4:4, esto es, sin pérdida. C) La misma matriz pero con un sub-muestreo 4:2:2, en el que se puede observar que ha ocurrido cierta compresión.<sup>11</sup>**

<sup>11</sup> Chroma Subsampling Notation, Charles Poyton, White Paper.



**Figura 2.4 A) La imagen que se desea muestrear B) El muestreo de la luminancia. C) El muestreo de la crominancia diferencial azul. D) El muestreo de la crominancia diferencial roja.<sup>12</sup>**

<sup>12</sup> <http://en.wikipedia.org/wiki/YCbCr>

Falta mencionar un último detalle respecto a una imagen, esta es su *relación de aspecto*, la cual se la define como  $x : y$ . Esto da la proporción horizontal con respecto de la vertical. Las relaciones de aspecto más comunes son de 4:3 y para equipos de alta definición 16:9. Esto quiere decir que si se tiene una pantalla de 27" de ancho esta tendrá 15" de alto, esto es, alargada horizontalmente (rectangular).

Una vez que se tiene el conocimiento de los detalles que se refieren a como se muestrea una imagen, es necesaria la pregunta, ¿Cómo se puede transformar la información de una imagen, que se encuentra representada en RGB a  $Y C_R C_B$ ?. Pues bien esto es parte de la recomendación ITU BT.601-5, que fue originalmente pensada para el muestreo de señales analógicas para los sistemas de televisión PAL - SECAM / NTSC (sistema de video europeo y sistema de video americano respectivamente). Esto es, se define para una señal analógica:

$$E'_Y = 0.299 \cdot E'_R + 0.587 \cdot E'_G + 0.114 \cdot E'_B$$

$$\left( \begin{matrix} E'_R - E'_Y \\ E'_B - E'_Y \end{matrix} \right) = \begin{pmatrix} 0.701 \cdot E'_R - 0.587 \cdot E'_G - 0.114 \cdot E'_B \\ -0.299 \cdot E'_R - 0.587 \cdot E'_G + 0.886 \cdot E'_B \end{pmatrix} \quad [13]$$

Estas mismas ecuaciones se pueden evaluar cuando se trata de una señal obtenida de una fuente digital directamente, y tienen la forma:

$$Y = \frac{77}{256} E'_{RD} + \frac{150}{256} E'_{GD} + \frac{29}{256} E'_{BD}$$

$$C_R = \frac{131}{256} E'_{RD} - \frac{110}{256} E'_{GD} - \frac{21}{256} E'_{BD} + 128$$

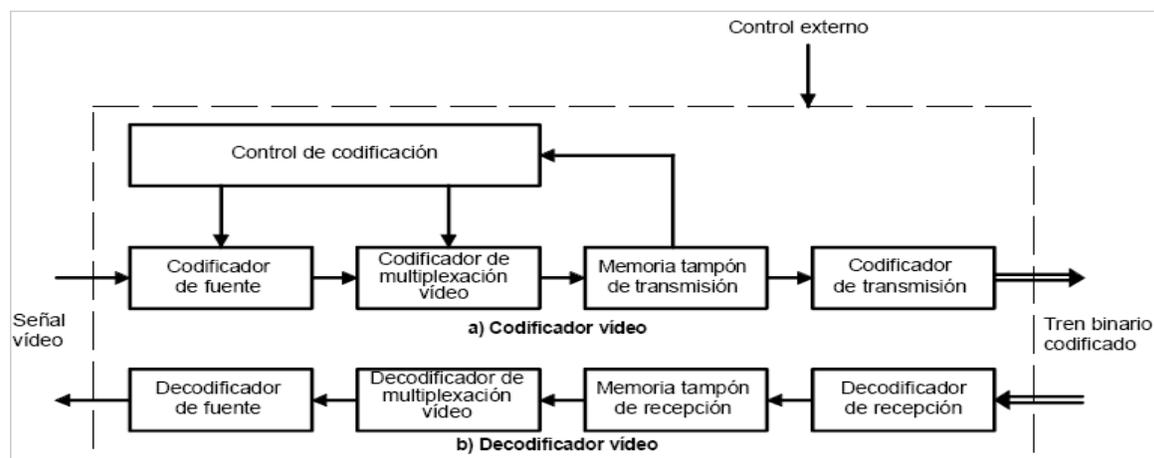
$$C_B = -\frac{44}{256} E'_{RD} - \frac{87}{256} E'_{GD} + \frac{131}{256} E'_{BD} + 128 \quad [13]$$

Dónde  $E'_{RD}$ ,  $E'_{GD}$  y  $E'_{BD}$  son los valores RGB de la señal digital de 8 bits.

Otras definiciones adicionales, dicen que  $Y$  puede tener 220 valores,  $C_R$ , 225 y  $C_B$  225, por lo que el número máximo de colores es de  $220 \times 225 \times 225$ , esto es, alrededor de once millones de colores. Además pueden producirse valores negativos, por lo que la gama debe limitarse.

También cabe mencionar que el valor de la luminancia para el color negro es 16 y para el color blanco es 235, la diferencia de color nula vale 128 y la diferencia de color máxima es 16 y 240.

Hasta este punto se conoce como realizar el sub-muestreo de una señal cuyos valores  $Y C_R C_B$  son conocidos a partir de las ecuaciones de conversión dadas anteriormente, provenientes de una señal digital RGB, pues bien esta es una parte de la definición de la recomendación ITU H.261, la cual especifica la definición del codec de video para servicios audiovisuales a  $p \times 64$  kbit/s, donde  $p$  va de 1 a 30.



**Figura 2.5 Diagrama resumido del codec de video especificado en la recomendación ITU H.261<sup>13</sup>**

Esta recomendación, como se mencionó anteriormente, también fue pensada para que trabajara en sistemas de televisión de 625 (PAL) y 525 (NTSC) líneas, y entre estas regiones, por lo que se definió un formato intermedio común (Common Intermediate Format, o CIF) sobre el cual trabajará el codificador de fuente. El mismo se define que trabajará con imágenes no entrelazadas que aparecen

<sup>13</sup> UIT-T H.261, UIT, Marzo 1993

30,000 / 1001 veces por segundo con una tolerancia de  $\pm 50$  ppm, las cuales se muestrean para obtener la componente de luminancia Y, y las dos componentes de diferencia de color ( $C_R$  y  $C_B$ ).

Pues bien ahora que se tienen claros los términos sobre los cuales trabaja el codificador de fuente (figura 2.5), es necesario definir en más detalle al CIF. En primer lugar la recomendación define que la estructura de muestreo de la luminancia es de 352 elementos de imagen por línea con 288 líneas por imagen, en una disposición ortogonal. El muestreo de cada una de las dos componentes de diferencia de color es de 176 elementos de imagen por línea con 144 líneas por imagen en disposición ortogonal, esto es, se realiza un sub-muestreo 4:2:2, en el cual las muestras de diferencia de color se sitúan de manera que sus límites de bloque coincidan con los límites de bloque de luminancia, tal como se muestra en la figura. La zona de imagen cubierta por estos números de elementos de imagen y líneas tienen una relación de aspecto 4:3 correspondiente a la porción activa de la entrada de video de norma local.



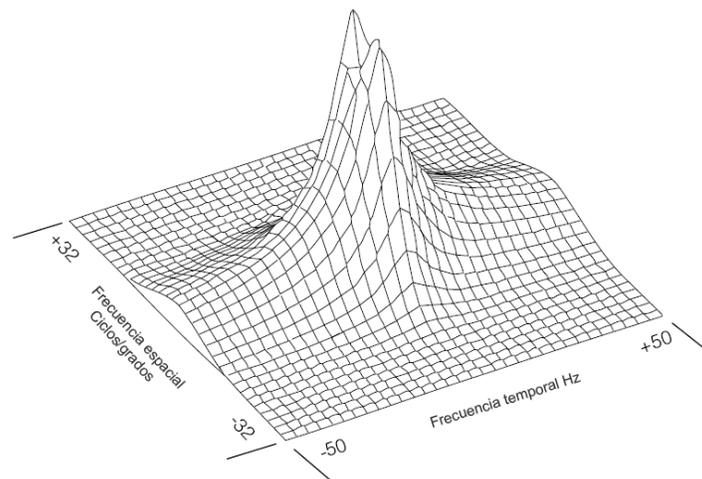
**Figura 2.6 Posición de las muestras de luminancia y crominancia en 4:2:2** <sup>13</sup>

La recomendación define un segundo formato de CIF, el QCIF (Quarter-CIF), el cual tiene la mitad de elementos de imagen y líneas que el formato anterior.

La recomendación también habla del período de transmisión de imagen de los codificadores, los cuales dice, deben dejar de transmitir al menos 0, 1, 2, ó 3 imágenes entre las imágenes transmitidas. La elección tanto del formato de CIF, como del número de imágenes que se dejan de transmitir no se define exactamente pero se sugiere el uso de la recomendación ITU H.221.

La predicción se la hace entre imágenes y puede aumentarse por compensación de movimiento y por un filtro espacial.

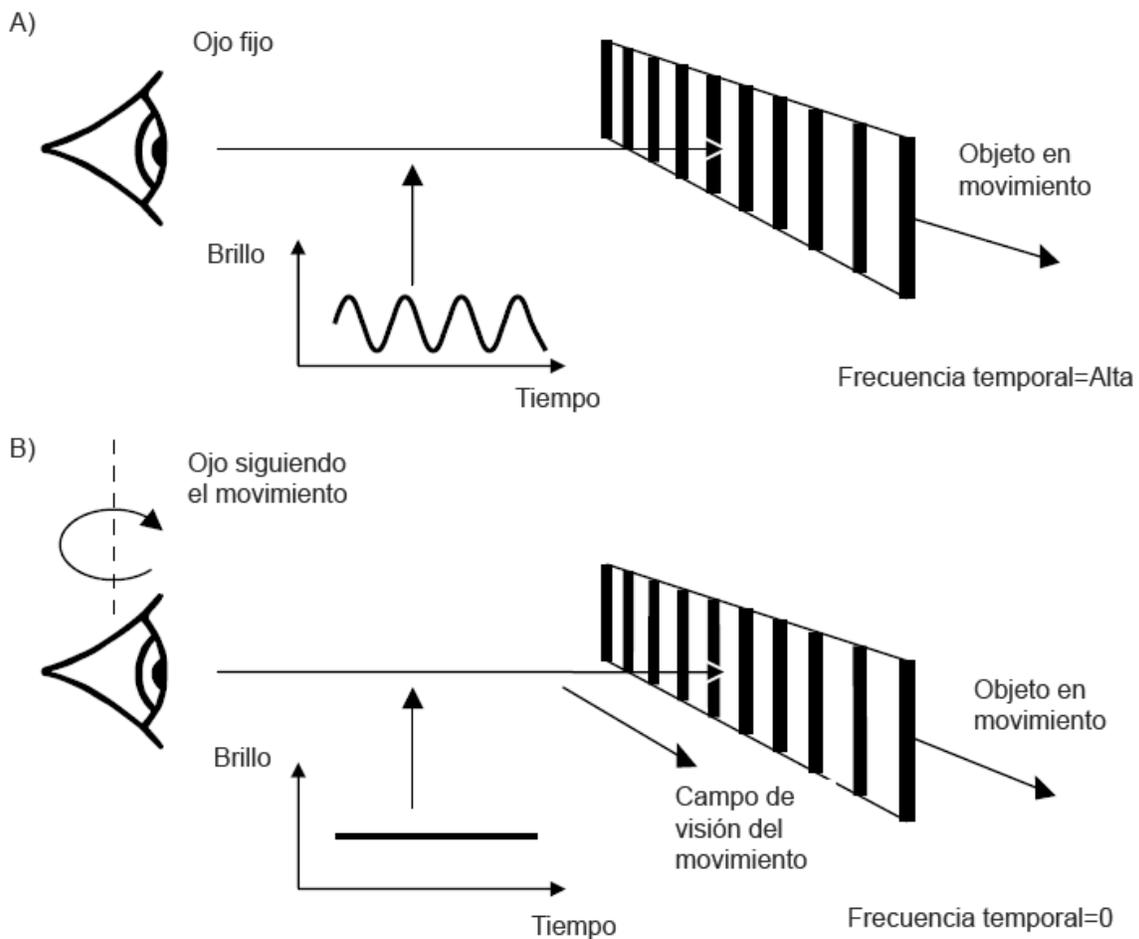
Para poder comprender como funciona la compensación de movimiento, se debe primero analizar otras limitaciones que son intrínsecas a la visión humana, en primer lugar se debe mencionar que esta tiene una mala respuesta a bajas frecuencias espaciales (el brillo medio de una escena) y además tiene una respuesta baja a las frecuencias temporales altas, la cual toma la forma de una demora, conocida como persistencia de la visión. El efecto de esta demora es que la resolución se pierde en las áreas donde la imagen se mueve rápidamente sobre la retina.



**Figura 2.7 Respuesta del ojo con respuesta a las frecuencias temporal (movimiento) y espacial (resolución).<sup>14</sup>**

<sup>14</sup> The Engineer's Guide to Motion Compensation, John Watkinson, Snell & Wilcox

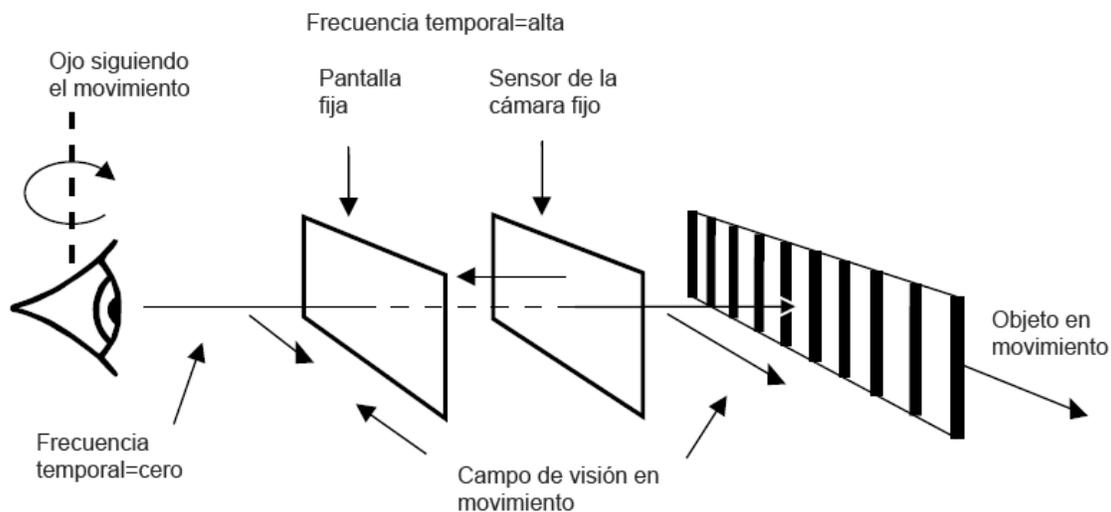
Como se puede observar en la figura 2.8 la respuesta del ojo cae muy rápidamente con la frecuencia temporal, a esto se le llama persistencia de la visión o borramiento por el movimiento.



**Figura 2.8 La persistencia de la visión humana. A) Un objeto pasa rápidamente causando borramiento por su movimiento. B) El mismo objeto es seguido por el movimiento del ojo evitando el borramiento.** <sup>14</sup>

El video es afectado de dos diferentes formas, la primera es el movimiento de un objeto que pasa frente a la cámara y en segundo término por el movimiento del ojo frente a la pantalla. La situación se pone ligeramente más complicada si se considera que las imágenes son muestreadas a la frecuencia de campo, la cual,

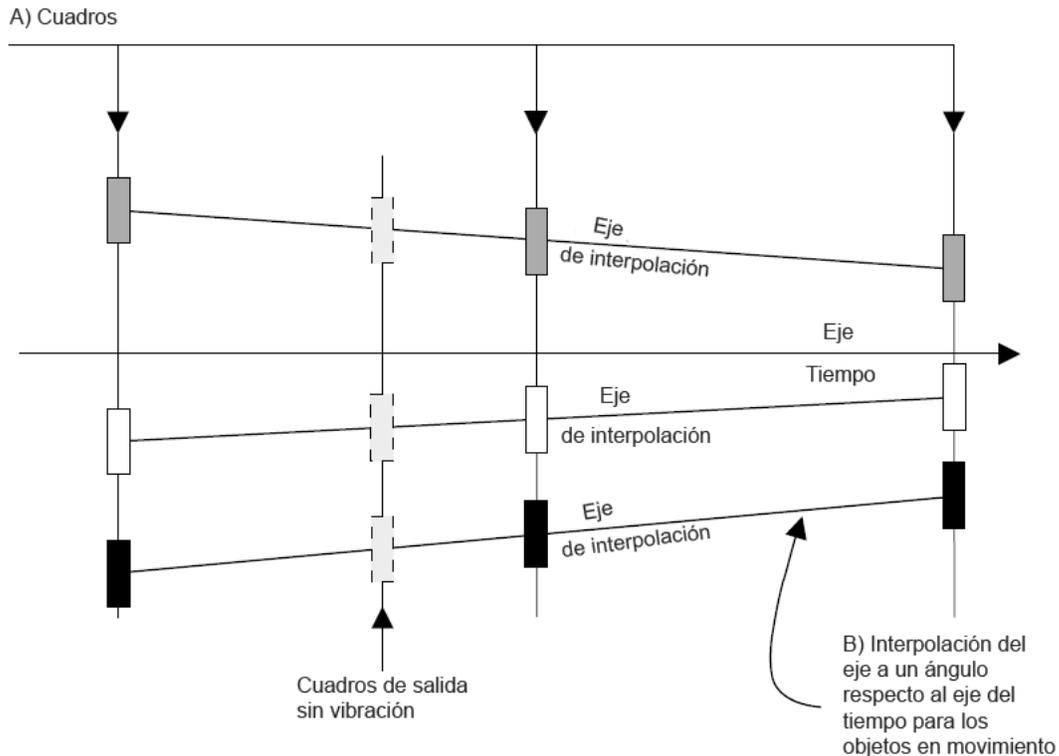
de acuerdo a la teoría del muestreo, no puede ser mayor a la mitad de la frecuencia de muestreo, de modo que la frecuencia temporal no puede ser mayor a 25 o 30 Hz. Ahora bien cuando un objeto pasa a gran velocidad por frente a la cámara, este movimiento produce frecuencias temporales superiores al límite de muestreo y se produce el efecto de aliasing (una copia de la señal muestreada a una frecuencia mayor que puede resultar en una distorsión al restituir la información). Sin embargo esta distorsión no es perceptible para el ojo debido a que este trata de seguir el movimiento del objeto en la pantalla.



**Figura 2.9 Se produce el efecto de aliasing pero no es percibido debido al moviendo del ojo al tratar de seguir al objeto en la pantalla. <sup>14</sup>**

En la figura 2.9 se puede observar que el movimiento se produce tanto relativo a la cámara, como después al proyectarse, relativo a la pantalla, de modo que se producen frecuencias temporales, las cuales reducen la resolución de la imagen, es por esto que las cámaras fotográficas deportivas tienen control de exposición, lo cual les permite tomar una exposición de una fracción de la frecuencia de muestreo de campo, sin embargo las cámaras de video no se desempeñan muy bien. Este control de apertura es efectivamente una forma de compensación de movimiento, que cuya ventaja evidente es el incremento en la resolución de las imágenes capturadas de objetos en movimiento.

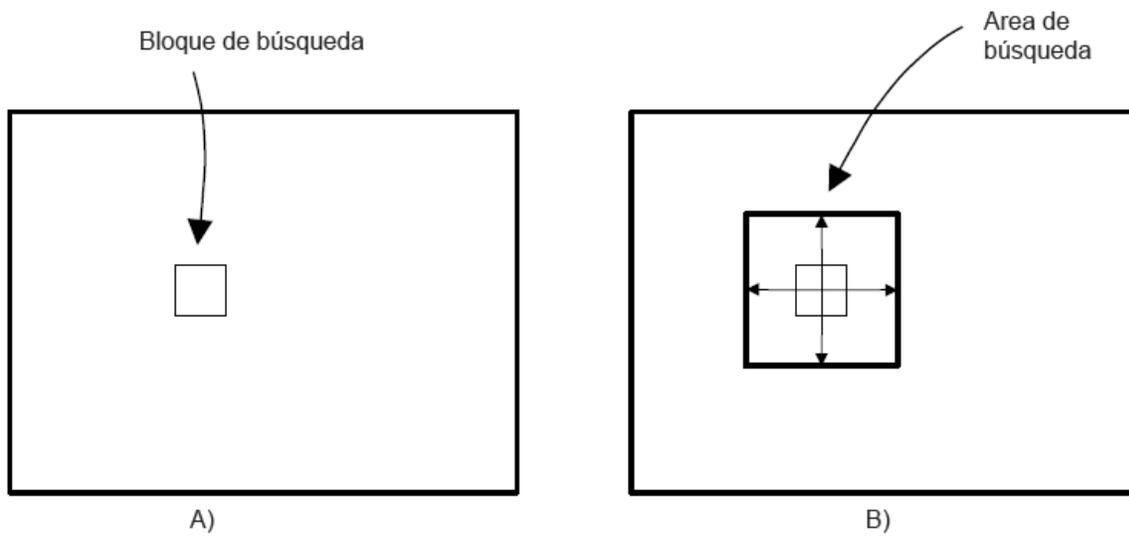
El principio de compensación de movimiento para el caso del video es bastante simple: se trata de computar dónde se moverá el objeto y proyectarlo en esa posición.



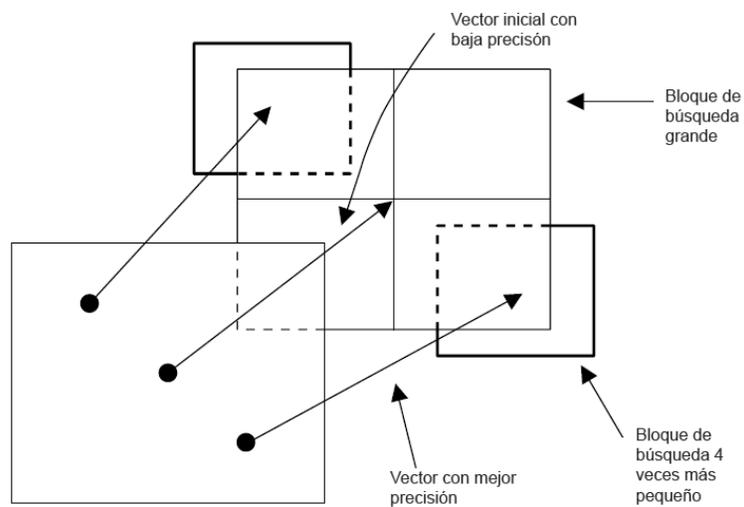
**Figura 2.10 Interpolación proyectada paralela al movimiento.** <sup>14</sup>

Una forma alternativa de ver la compensación de movimiento la tenemos en la figura 2.10, en la que consideramos lo que ocurre en el volumen espacio temporal, en ella podemos observar que el eje interpolado se encuentra alineado con el movimiento de cada objeto, de modo que cada objeto ya no se mueve respecto a su propio eje de interpolación y no genera más frecuencias temporales debidas al movimiento. El proceso requiere de un conversor estándar el cual contiene filtros para permitir que el eje de interpolación se mueva dinámicamente dentro de cada cuadro de salida. Las señales que mueven los ejes de interpolación se llaman *vectores de movimiento*. Es trabajo del sistema de estimación de movimiento, el proveer estos vectores de movimiento.

Además es mejor trabajar la estimación de movimiento empleando bloques de píxeles. En cada bloque se compara un píxel a la vez con un bloque similar en la misma posición en la siguiente imagen para detectar si ha ocurrido movimiento. Si no hay movimiento existe una alta correlación entre los valores de los píxeles. Si existe movimiento es necesario buscar entre los píxeles del *área de búsqueda* dentro del bloque hasta obtener la mejor correlación. Este concepto requiere de una enorme capacidad de computo, y aún más grande si tenemos objetos pequeños moviéndose como ocurre en las transmisiones deportivas. Es por ello que este procedimiento se lo lleva a cabo en etapas, en la primera se cubre un gran rango de movimiento pero con poca precisión, esto es, se emplea una imagen sub muestreada, para luego mejorar en la precisión, el inconveniente con este procedimiento es que se pueden perder los objetos pequeños que se encuentran en movimiento, en ciertos cuadros creando una impresión de mayor movimiento en los cuadros restantes. Esta técnica tiene otra desventaja, si el bloque de búsqueda es pequeño, es posible encontrar buena correlación de píxeles, pero creando una estimación errónea (ocurrencias falsas), en cambio si el bloque de búsqueda es grande, se producirán muchas menos correlaciones, por lo que es más difícil estimar el movimiento de todos los objetos (medianos o grandes). Una mejor técnica es la denominada *Correlación Espacial Jerárquica*, la cual nos permite trabajar de mejor forma. Esta consiste en que en primer lugar tenemos un bloque de búsqueda grande, el cual nos permite estimar de una forma poco precisa el movimiento. En la siguiente etapa, el bloque de búsqueda se divide en cuatro, y empleando el vector de movimiento de la primera etapa, se estima de una mejor forma el movimiento, y así subsecuentemente hasta encontrar el grado de predicción requerida.



**Figura 2.11 A) El algoritmo espera encontrar un bloque similar en la siguiente imagen en la misma posición. B) Como no ha sido posible encontrar el bloque, este se mueve por el área de búsqueda.** <sup>14</sup>

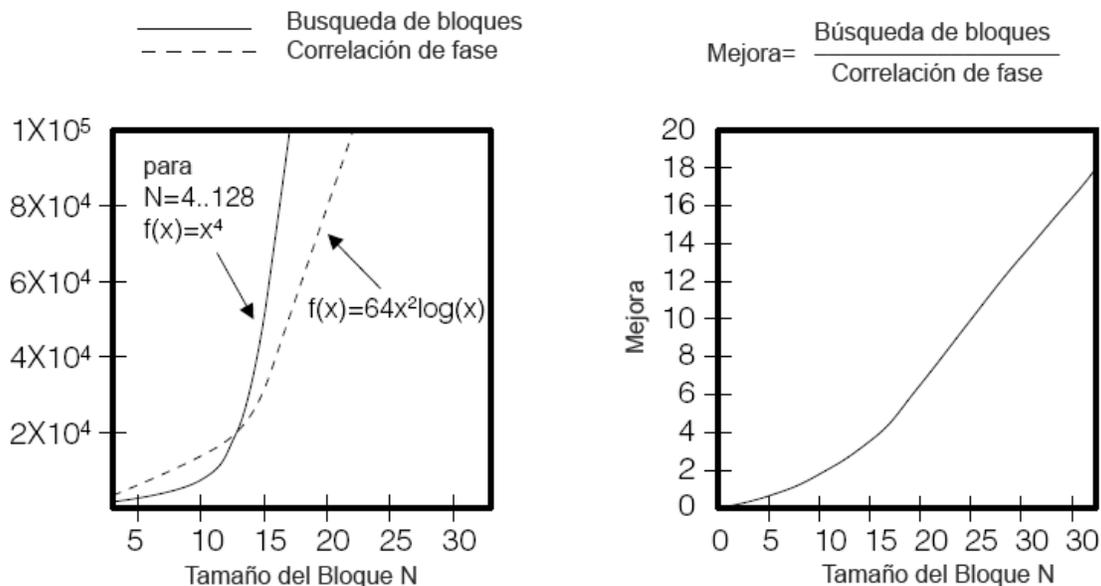


**Figura 2.12 Correlación Espacial Jerárquica.** <sup>14</sup>

Existe otra técnica para estimar el movimiento, la de *Gradientes*, en la cual se trata de encontrar dos gradientes de luminancia espacial en dos imágenes consecutivas

y con ellas encontrar los vectores de movimiento. Pero la mejor, tal vez la más compleja pero ciertamente la más empleada de todas es la de *Correlación de Fase*.

La Correlación de Fase, trabaja realizando un análisis espectral de dos cuadros sucesivos, y entonces se restan todas las fases de las componentes espectrales. Al resultado de esta operación se le aplica la operación inversa, la cual directamente revela picos, cuyas posiciones revelan claramente movimiento entre los cuadros. La naturaleza del dominio de la transformación nos permite medir con precisión la dirección y la distancia del movimiento, pero no es posible determinar directamente el área de la pantalla dónde este toma lugar, es por ello que a este procedimiento le sigue un procedimiento de *Correlación de Bloques*, pero en este caso el área de búsqueda se encuentra limitada por la información obtenida por la Correlación de Fase. Por otra parte, se debe mencionar que los cálculos matemáticos que se deben realizar para el análisis de la Correlación de Fase son sumamente complejos, y aunque estos reducen el volumen de cálculos en la búsqueda de la posición de los vectores de movimiento, estos, en conjunto, siguen teniendo un gran volumen.



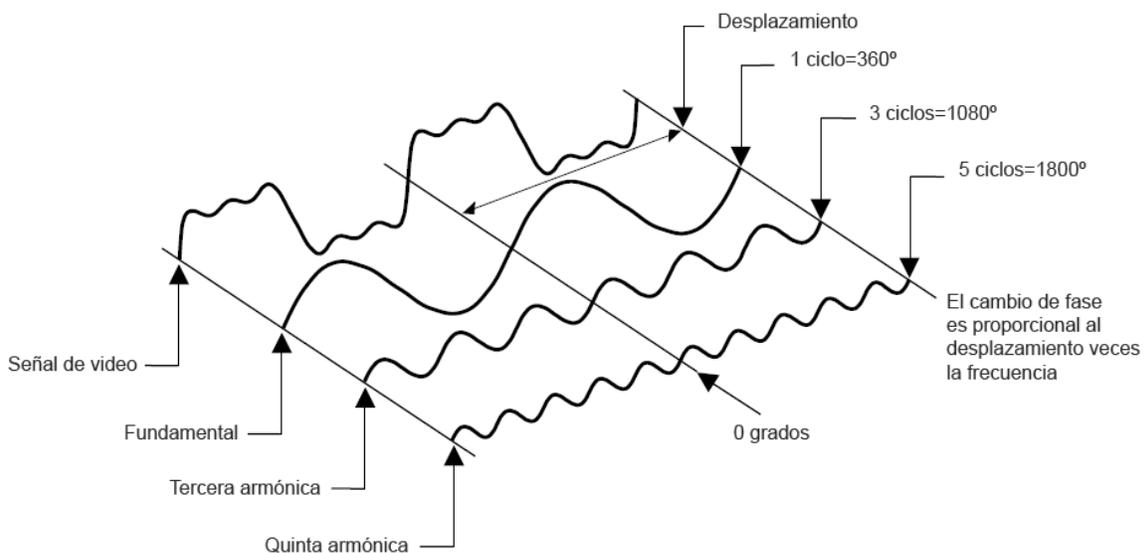
**Figura 2.13 Comparación del volumen de cálculos.** <sup>14</sup>

Una ventaja adicional, es que en el proceso de obtención de la correlación de fases, se elimina la información de la amplitud de la señal lo cual permite que se distinguen objetos moviéndose en sombras así como objetos tenues.

Pues bien, hasta aquí se ha visto el porque este método el mejor y por lo tanto el más empleado para la estimación de movimiento, pero no se ha visto porque se dice que también es el más complejo, esta explicación viene a continuación.

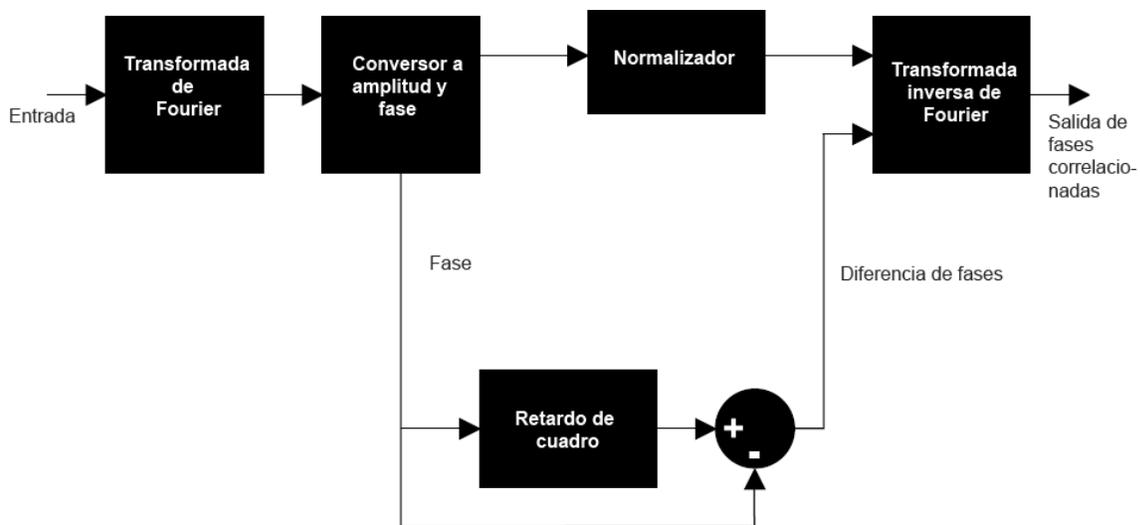
La Correlación de fases se la obtiene empleando la transformada de Fourier sobre los cuadros que se desea comparar. Una línea de luminancia en el dominio digital, consiste en una serie de muestras y es función del brillo con respecto a la distancia a través de la pantalla. La transformada de Fourier convierte esta función en un espectro de frecuencias espaciales (unidades o ciclos por el ancho de la imagen) y fases.

Todos los sistemas de video deben ser manejados bajo un sistema de *fase lineal*. Un sistema de fase lineal es aquel en el cual un retardo es el mismo en todas las frecuencias. Esto es un cambio de fase es proporcional a la frecuencia.



**Figura 2.14 Sistema de fase lineal.** <sup>14</sup>

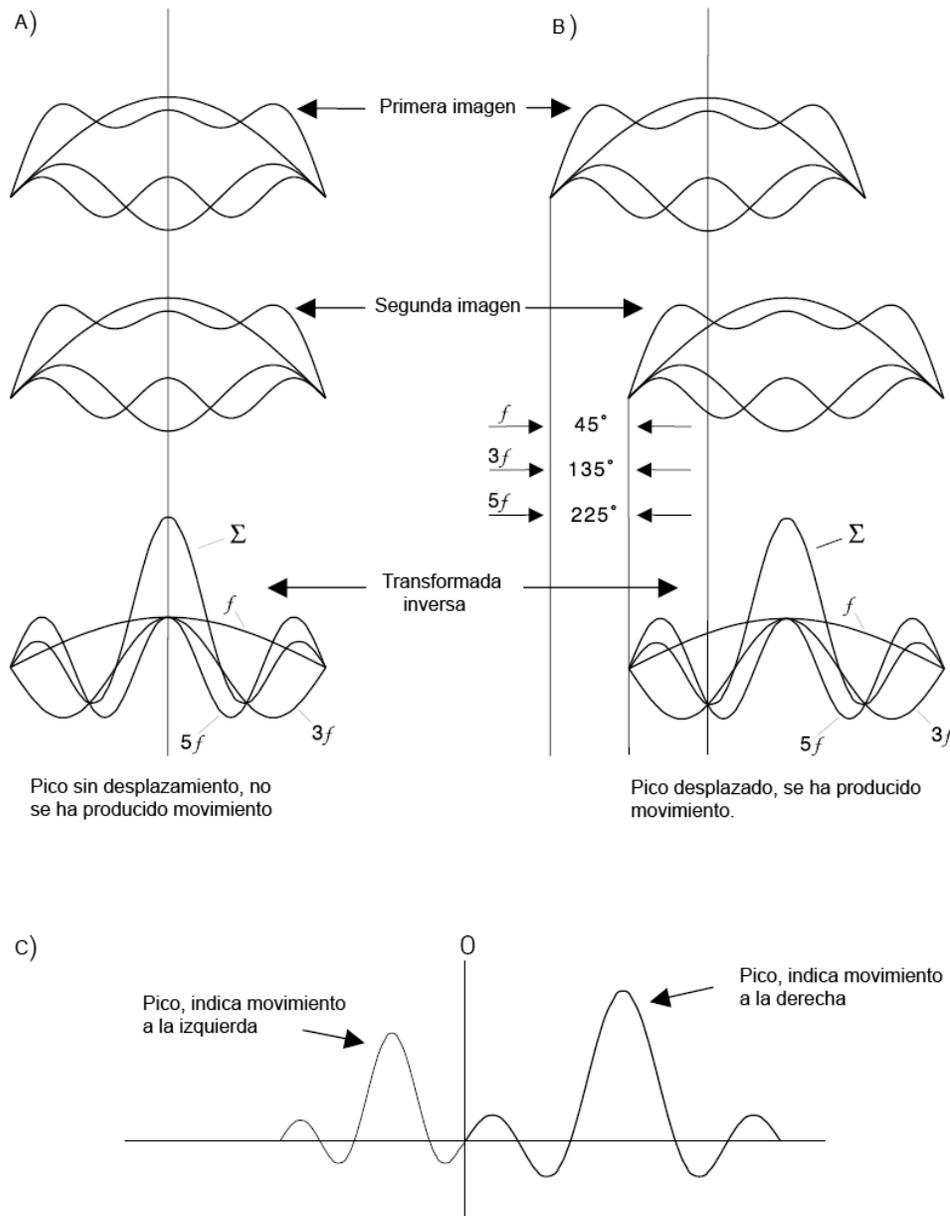
En el dominio espacial, un cambio de fase corresponde a un movimiento físico. Así en la figura se muestra que si una forma de onda se mueve entre cuadros, a lo largo de una línea, la menor frecuencia en la transformada de Fourier tendrá un determinado desplazamiento de fase, dos veces tal frecuencia tendrá dos veces el desplazamiento de fase, y así sucesivamente. Entonces es posible medir el movimiento entre dos cuadros sucesivos si se analizan las diferencias de fase en el espectro de Fourier. Este es el principio de la correlación de fase.



**Figura 2.15 Correlacionador de fase básico.** <sup>14</sup>

En la figura 2.15 se tiene la estructura de un correlacionador de fase unidimensional. La forma en la que trabaja es la siguiente: Las transformadas de Fourier de dos líneas de diferentes cuadros son computadas y expresadas en forma polar (amplitud y fase). Las fases de una transformada son sustraídas de las fases de la misma frecuencia de la otra transformada, cualquier componente de frecuencia que tenga una amplitud significativa, es normalizada. El resultado es un conjunto de componentes de frecuencia con la misma amplitud, pero con fases que corresponden a las diferencias entre los dos cuadros, estos componentes

forman la entrada para la transformada inversa, de la cual obtenemos las fases correlacionadas.

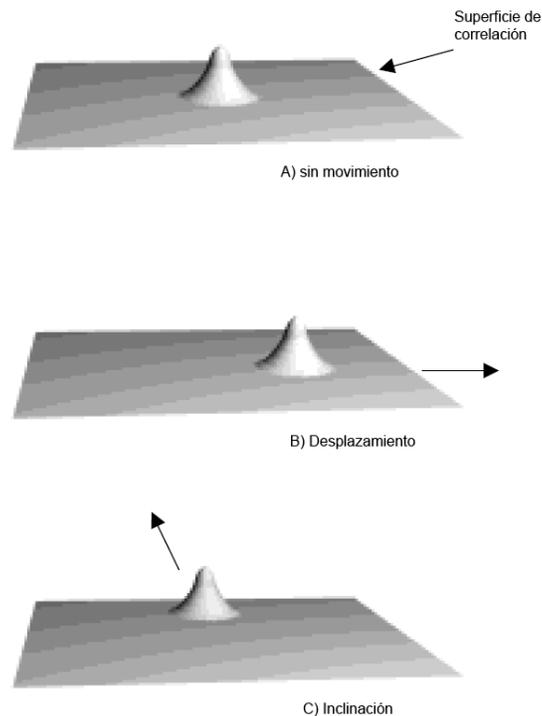


**Figura 2.16 Transformada inversa muestra desplazamientos. A) No hubo movimiento. B) Movimiento a la derecha. C) Movimiento a derecha e izquierda.<sup>14</sup>**

Un ejemplo de lo dicho se refleja en la figura 2.16.

La correlación de fases mide el movimiento entre cuadros en lugar de inferirla a través de las coincidencias de luminancia.

En la práctica el proceso completo es bidimensional y no unidimensional como se ha mostrado hasta el momento, esto es, se calculan dos transformadas de Fourier por cada cuadro, se substraen las fases, y se calcula la transformada inversa bidimensional, la salida de esto es un plano en el cual un pico tridimensional se eleva, se ha obtenido la *superficie de correlación*, como se muestra en la figura 2.17.



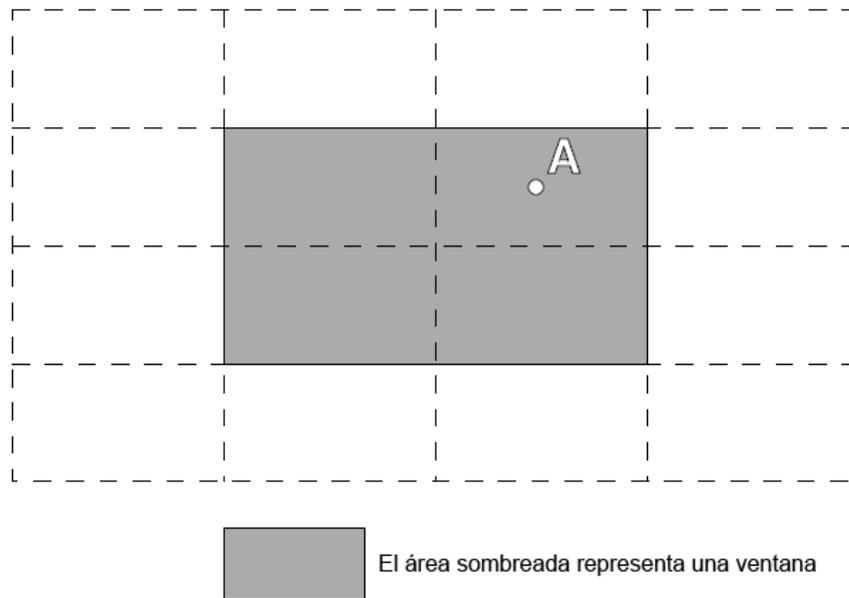
**Figura 2.17 Superficie de correlación.** <sup>14</sup>

La superficie de correlación tiene un pico central cuando no se ha producido movimiento.

Se ha producido un desplazamiento a la derecha (rotación de la cámara).

Se ha producido una inclinación de la cámara. El pico se mueve en ángulos rectos respecto al desplazamiento.

Como es de suponerse, el volumen de cálculo requerido para computar la superficie de correlación de cada cuadro se incrementa con el tamaño del cuadro, por ello a cada cuadro normalmente se los divide en bloques más pequeños empleando una función de ventana (figura 2.18), a esto se le llama el pre procesamiento.

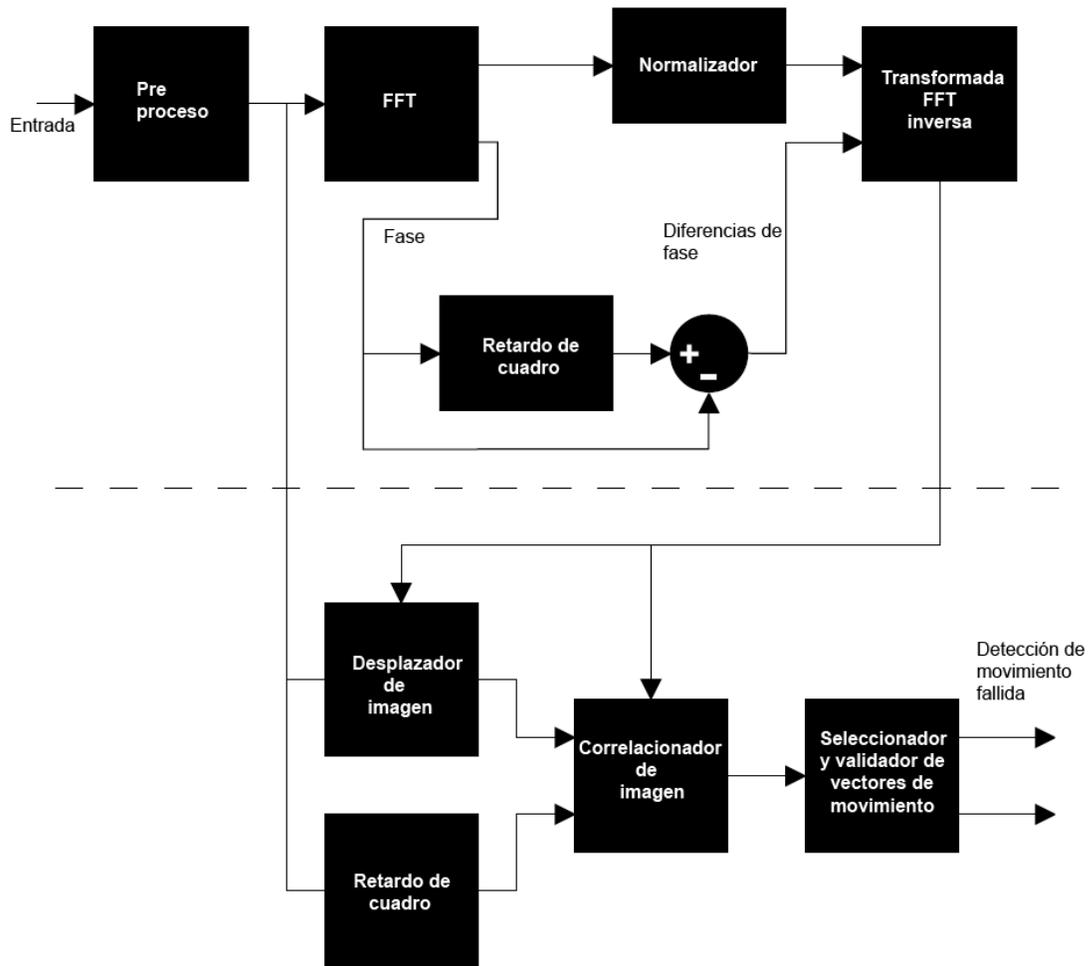


**Figura 2.18 Un cuadro subdividido en bloques.** <sup>14</sup>

El tamaño de la ventana que se escoge permite detectar el movimiento en los límites de la visión humana. La ventana suele ser rectangular debido a que es más común el movimiento horizontal.

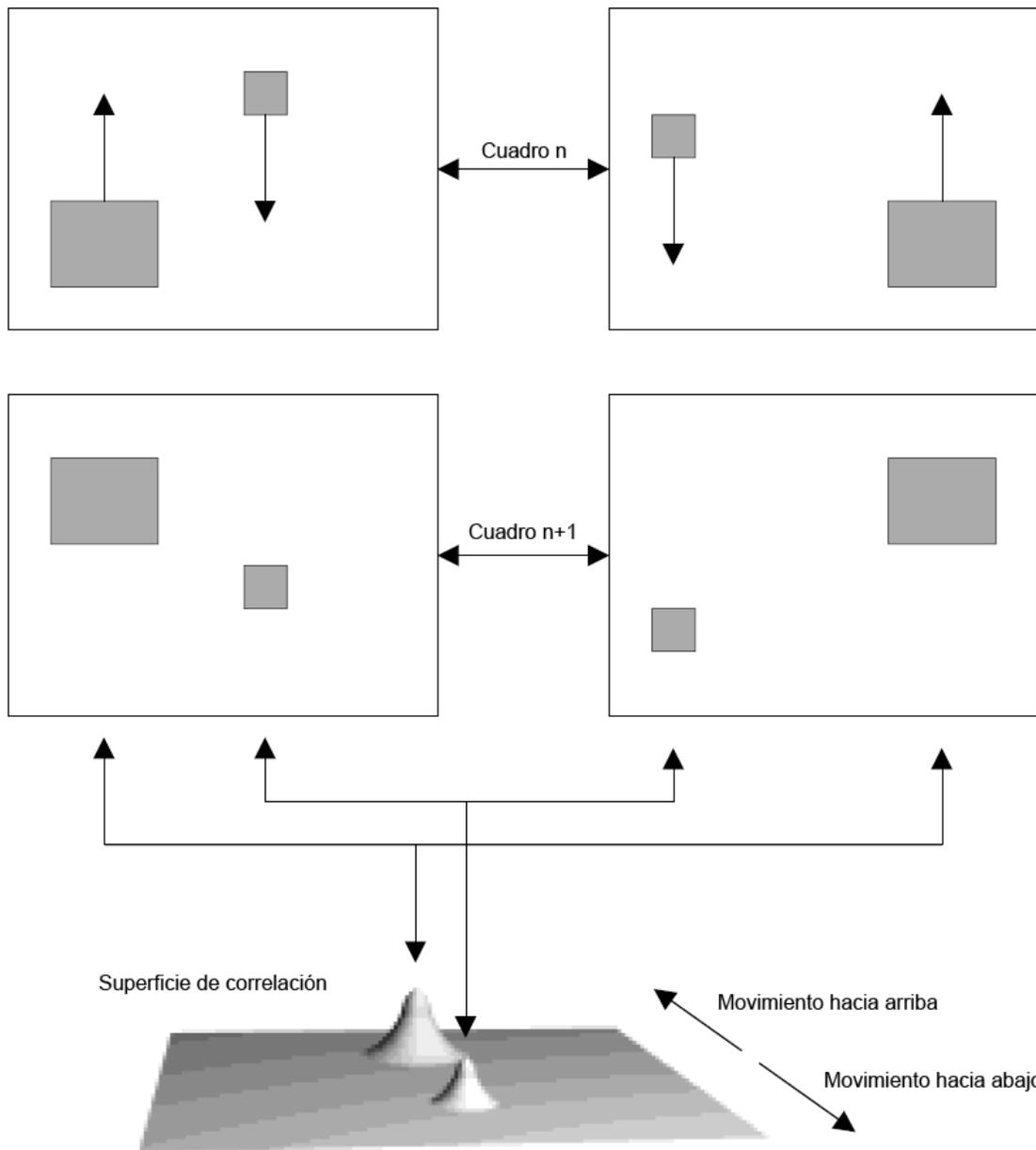
En la figura 2.19, se muestra a continuación se muestra el diagrama en bloques de un estimador de movimiento por desplazamiento de fases. En el se ve que seguido al pre procesamiento, en el cual se divide al cuadro en bloques ventaneados, tenemos que cada uno de estos elementos es sujeto a la

Transformada Rápida de Fourier (Fast Fourier Transform), el espectro de salida es convertido al formato polar de amplitud y fase para luego los valores de amplitud ser normalizados (para eliminar cualquier variación de iluminación o degradación) y las fases ser sustraídas del cuadro anterior en cada ventana, para luego de aplicar la transformada inversa, obtener la superficie de correlación, la misma que contiene picos cuyas posiciones miden, distancias y direcciones de los objetos que se han movido en cada ventana. Debido a que una de las características de la transformada de Fourier nos dice que mientras más precisamente conocido es el espectro de una señal menos precisamente conocido es su dominio espacial. Esto se traduce como: no es posible conocer la posición dentro de la ventana de los objetos en movimiento que han causado estos picos en la superficie de correlación.



**Figura 2.19 Diagrama en bloques de un estimador de movimiento por desplazamiento de fases.** <sup>14</sup>

Para ilustrar lo dicho tenemos la figura 2.20, en la que podemos observar dos bloques ventaneados en cuadros consecutivos. En ambos tenemos los mismos objetos moviéndose con la misma velocidad, pero desde diferentes puntos de inicio. La superficie de correlación será igual para ambos casos. De modo que el proceso de correlación de fase, necesita ser seguido por un proceso denominado asignación de vectores, el cual identifica las áreas de la imagen dónde el movimiento medido tuvo lugar y establece un nivel de confianza para la identificación. Esta etapa también se la puede observar en la figura anterior.

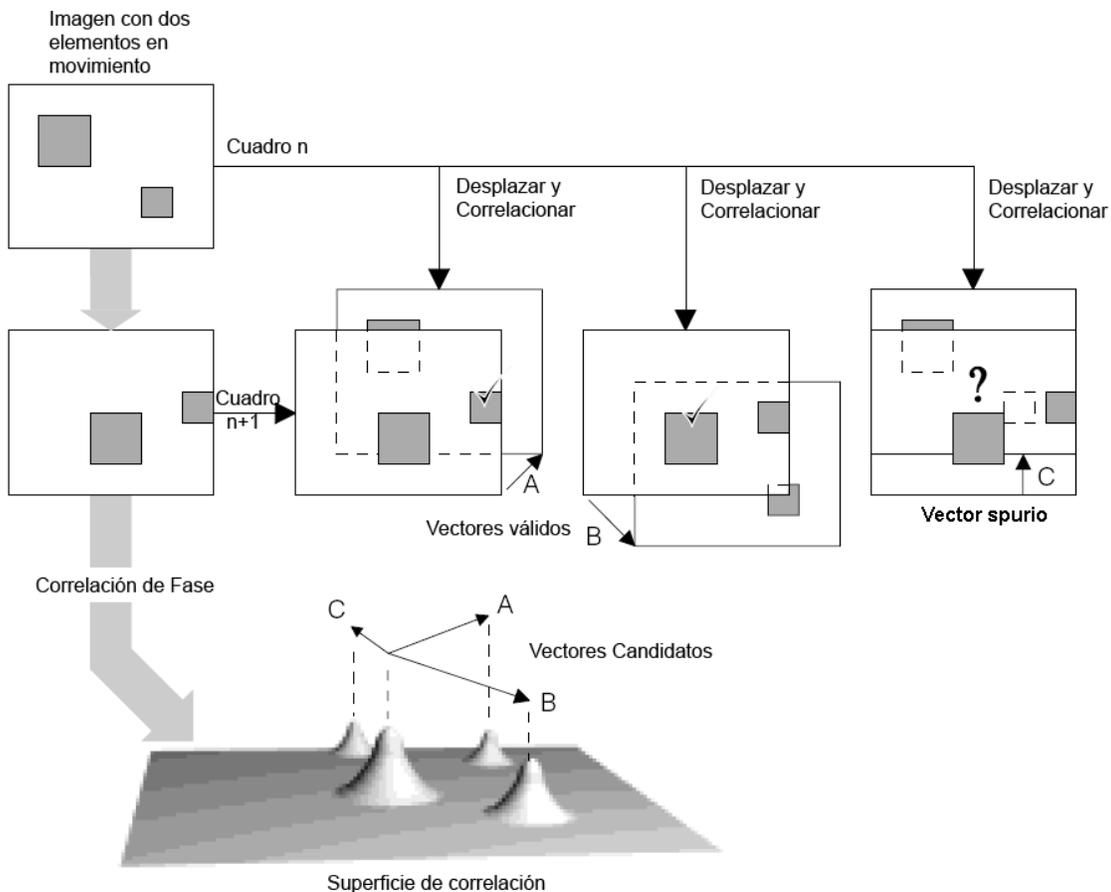


**Figura 2.20 Superficie de correlación vs. identificación de movimiento.** <sup>14</sup>

Para emplear la terminología estimación de movimiento, el proceso de correlación de fase produce vectores candidatos, y en un proceso denominado correlación de imagen se asignan estos vectores a las diferentes áreas de la imagen. De muchas formas el proceso de asignación de vectores es más complejo que el proceso de

encontrar la superficie de correlación debido a que la asignación de vectores responde a un número infinito de variaciones en la imagen.

A continuación tenemos la figura 2.21, en la cual podemos observar el proceso de correlación de imagen. En este ejemplo la superficie de correlación tiene tres picos, los cuales definen tres posibles movimientos entre cuadros sucesivos. Un cuadro pre procesado es sucesivamente desplazado por cada uno de los vectores candidatos, y comparado con el siguiente cuadro píxel a píxel.

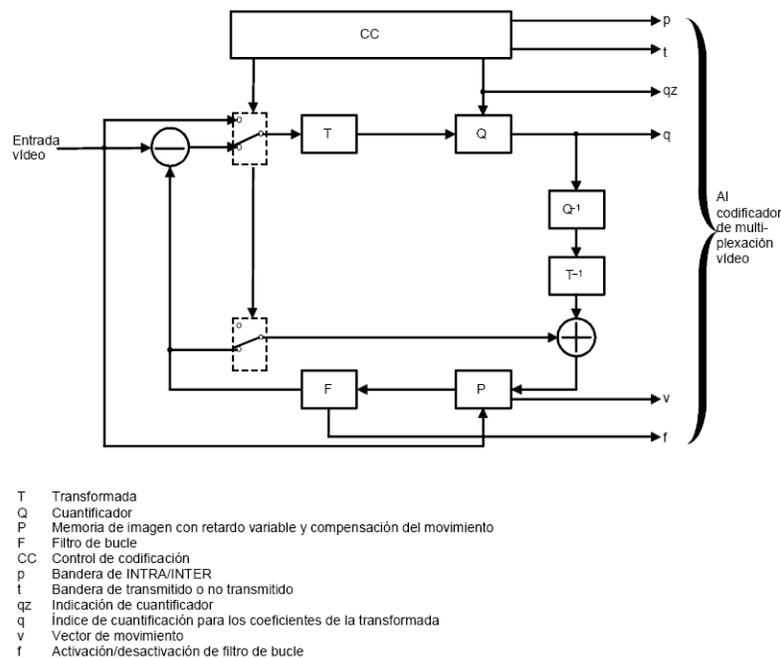


**Figura 2.21 Asignación de vectores.** <sup>14</sup>

Las similitudes o correlaciones entre los valores de los píxeles, indican que un área con movimiento medido ha sido encontrada, esto ocurre para dos de los vectores candidatos, y estos son asignados a esas áreas. Sin embargo el

desplazamiento para el tercer no resulta en una correlación significativa, por lo que es considerado espurio. La habilidad de un algoritmo de eliminar los vectores espurios y la asignación de niveles de confianza a los vectores asignados son fundamentales para la implementación del estimador de movimientos.

El desplazamiento del área de un cuadro toma lugar en dos etapas, el desplazamiento que será medido en píxeles y el valor, que es dividido por la parte entera del número de píxeles más cercano, la parte fraccionaria se expresa en sub – píxeles. Los píxeles del cuadro de entrada se almacenan en la RAM, la cual el interpolador direcciona para obtener el filtrado de entrada. La parte entera del impulso de desplazamiento es simplemente agregado a la RAM, de modo que parece que los píxeles del cuadro de entrada han sido desplazados, el desplazamiento vertical cambia las direcciones de las filas y el desplazamiento horizontal cambia las direcciones de las columnas, a esto se le llama mapeo de direccionamiento. Todo esto describe un sistema de compensación de movimiento, que como se podrá imaginar es sumamente complejo de implementar.



**Figura 2.22 El codificador de fuente según ITU H.261. 13**

En la recomendación se emplea una variación de la transformada discreta de Fourier, la denominada *transformada discreta de cosenos* (Discrete Cosine Transform), la cual emplea únicamente números reales, lo cual la hace más aplicable para este tipo de algoritmos.

H.261 también define lo referente a como se comprimirán imágenes fijas, a través de la recomendación T.81, y esto se lo hace de la siguiente manera:

En primer lugar se cambia el espacio de color de RGB a  $Y C_R C_B$ , para luego dividir la imagen en bloques de 8x8 píxeles (figura 2.23), y macro bloques consistentes en 4 bloques de luminancia más un bloque de crominancia en rojo, más un bloque de crominancia en azul.

Si no se obtiene un número entero de bloques el espacio se rellena con color, para luego realizar el procedimiento de sub-muestreo de la imagen, para luego obtener un conjunto de bloques de 8x8 bits de la forma mencionada anteriormente.

$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

**Figura 2.23 Bloque de 8x8 luminancia / crominancia de una imagen.**

De aquí tenemos dos caminos, el primero que nos permite comprimir bastante la imagen y tiene pérdidas o simplemente comprimir sin pérdidas. En el primer caso nosotros aplicamos la DCT bidimensional de tipo II, a cada bloque, y con ello

obtenemos los coeficientes DCT, en el extremo superior izquierdo de la matriz del bloque tendremos la componente DC de la transformada (figura 2.25), siendo los otros 63 elementos componentes AC de la misma.

$$\begin{bmatrix} -76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\ -65 & -69 & -73 & -38 & -19 & -43 & -59 & -56 \\ -66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\ -65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\ -61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\ -49 & -63 & -68 & -58 & -51 & -60 & -70 & -53 \\ -43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\ -41 & -49 & -59 & -60 & -63 & -52 & -50 & -34 \end{bmatrix}$$

**Figura 2.24 Bloque de luminancia / crominancia restado 128.**

$$\begin{bmatrix} -415 & -30 & -61 & 27 & 56 & -20 & -2 & 0 \\ 4 & -22 & -61 & 10 & 13 & -7 & -9 & 5 \\ -47 & 7 & 77 & -25 & -29 & 10 & 5 & -6 \\ -49 & 12 & 34 & -15 & -10 & 6 & 2 & 2 \\ 12 & -7 & -13 & -4 & -2 & 2 & -3 & 3 \\ -8 & 3 & 2 & -6 & -2 & 1 & 4 & 2 \\ -1 & 0 & 0 & -2 & -1 & -3 & 4 & -1 \\ 0 & 0 & -1 & -4 & -1 & 0 & 1 & 2 \end{bmatrix}$$

**Figura 2.25 Bloque con los índices de la DCT.**

Debido a que el ojo humano es bueno detectando la diferencia de brillo sobre un área relativamente grande, pero no puede distinguir la intensidad del brillo, o las frecuencias altas de la imagen, es posible eliminar la mayor parte de la información de alta frecuencia de la imagen, reduciendo con ello la cantidad de información que se va a almacenar de la imagen.

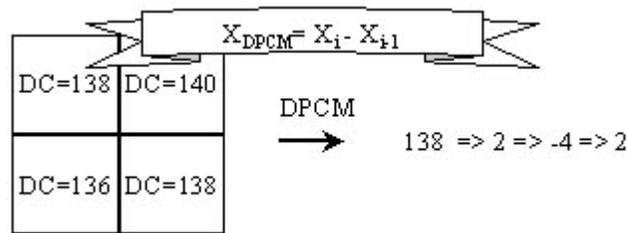
$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

**Figura 2.26 Matriz de cuantificación.**

$$\begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

**Figura 2.27 Bloque DCT luego de aplicar la matriz de cuantificación.**

Seguidamente se agrupan los valores de la matriz de cada bloque en zig-zag para juntar los ceros, para luego emplear el algoritmo RLE para agrupar los valores de magnitudes similares. A continuación se pueden emplear algoritmos de codificación predictiva como DPCM, el cual permite reducir la redundancia estadística de la información. Este se base en el valor presente para predecir el valor siguiente y codificar su diferencia (denominada error de predicción). DPCM se aplica únicamente a los valores DC de los bloques.



**Figura 2.28 Codificación DPCM**

Finalmente para comprimir aun más la información de la imagen es posible emplear Huffman (VLC Variable Length Coding, método en el cual se emplean palabras más cortas para los valores que se repiten más frecuentemente y palabras más largas para los valores que se repiten menos frecuentemente.)

Debido a que la tabla de símbolos de Huffman puede ser muy grande aplicamos otro alcance para la creación de símbolos basados en el tamaño (exponente base 2 ) y definimos el campo de bits adicionales para la polaridad, como se muestra en la tabla 2.1., de modo que solo necesitamos 12 códigos para realizar el proceso de compresión del bloque empleando Huffman.

Diferencia de Coef. DC	Tam.	Códigos típicos de Huffman	Bits adicionales (binario)
0	0	00	-
-1,1	1	010	0,1
-3,-2,2,3	2	011	00,01,10,11
-7,...,-4,4,...,7	3	100	000,...,011,100,...,111
-15,...,-8,8,...,15	4	101	0000,...,0111,1000,...,1111
⋮	⋮	⋮	⋮
-1023,...,-512,512,...,1023	10	1111 1110	00 0000 0000,...,11 1111 1111
-2047,...,-1024,1024,...,2047	11	1 1111 1110	000 0000 0000,...,111 1111 1111

**Tabla 2.1 Coeficientes de Huffman**

A este procedimiento se le conoce como codificación *Intra* (figra 2.29) de las imágenes.

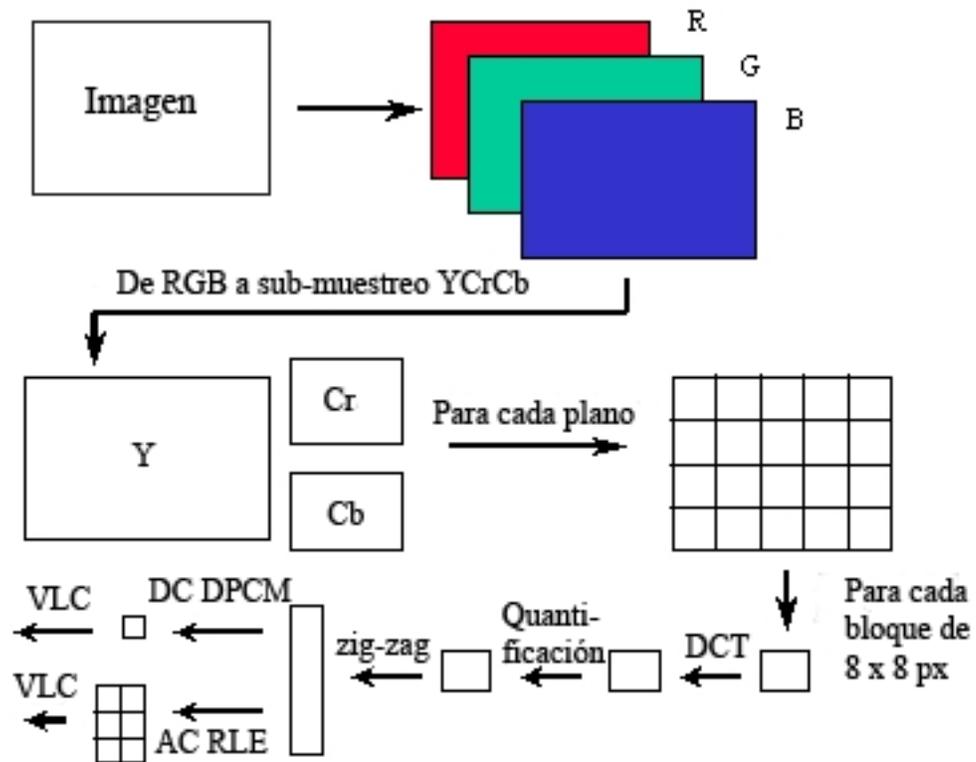


Figura 2.29. Resumen de la codificación Intra de imágenes.<sup>15</sup>

El segundo método también conocido como *codificación por entropía* es el siguiente:

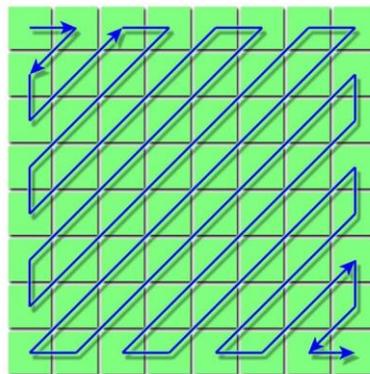


Figura 2.30 Orden de los elementos de la imagen en "zig-zag"

<sup>15</sup> ITU-T H.263, ITU, Junio 2001

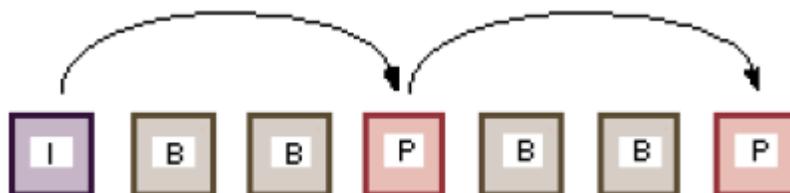
En primer lugar se ordenan los componentes de la imagen en “zig-zag”, empleando el algoritmo RLE (Run Length Encoding), el cual permite agrupar las frecuencias similares, y se emplea la codificación de Huffman para la compresión.

La decodificación sigue los mismos pasos en orden inverso.

Bien, queda claro como codificar las imágenes capturadas, ahora es necesario aclarar como se codifican las imágenes predichas, esto es las imágenes que serán recreadas a partir de una imagen y sus vectores de movimiento, a esta codificación se la denomina *Inter*. Existen dos tipos de cuadros predichos los P y los B. Los cuadros P sirven como referencia para un cálculo posterior mientras que los B no, de modo que en los primeros hay que tener más cuidado con las variaciones (errores) que en los cuadros de tipo B.

Pues bien los cuadros de tipo P se calculan macro bloque por macro bloque, como la diferencia entre el macro bloque de movimiento compensado y el macro bloque actual para luego ejecutar una DCT bidimensional, lo cual nos da como resultado una matriz de coeficientes, los cuales son después cuantificados son agrupados mediante RLE y codificados mediante VLC o Huffman.

Para crear un bloque de tipo P se tienen que tomar varias decisiones, en primer lugar, como se va a dividir la imagen, luego se debe determinar los vectores de movimiento, a continuación se debe determinar cómo se va a codificar el macro bloque, como Intra o como predicho, para finalmente determinar la escala de cuantificación.



**Figura 2.31 Los cuadros tipo P**

La creación de los cuadros B es similar a los cuadros P, al menos en su procedimiento, pero no requieren tanta exactitud.

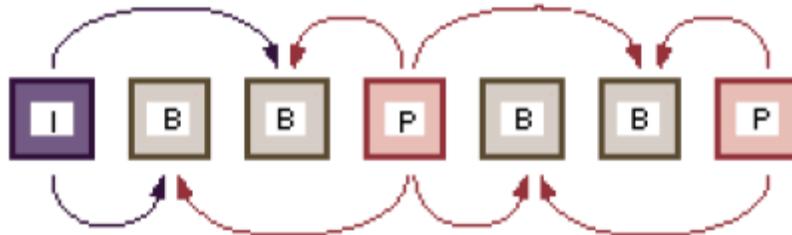


Figura 2.32 Los cuadros tipo B

En la figura 2.33 se muestra de forma resumida el procedimiento para la creación de estos cuadros, que como se mencionó anteriormente está muy ligado con la estimación de movimiento.

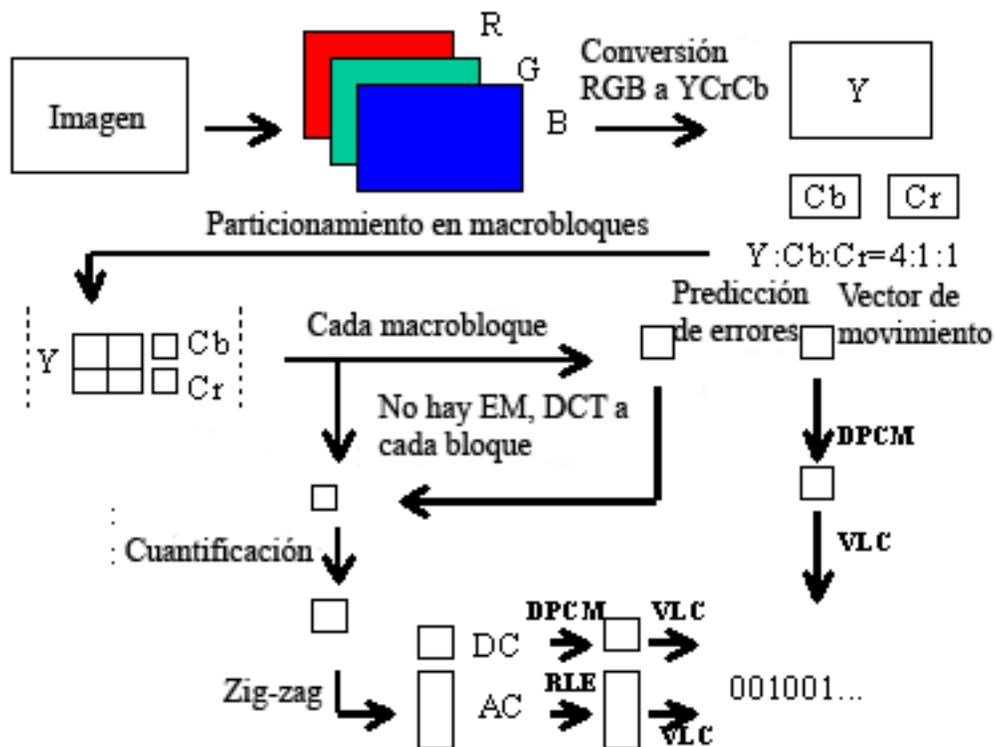


Figura 2.33 Resumen de la codificación Inter de imágenes. <sup>15</sup>

El detalle de cómo se realiza cada procedimiento esta claramente explicado en la recomendación H.263

Una vez que se tienen claras todas las definiciones necesarias para la implementación del codec de video, se debe mencionar que debido a la alta complejidad que implica la predicción de movimiento empleando superficies de correlación, se ha decidido realizar la predicción de movimiento por correlación espacial jerárquica.

Formato de imagen	Número de píxels de luminancia (dx)	Número de líneas de luminancia (dy)	Número de píxels de crominancia (dx/2)	Número de líneas de crominancia (dy/2)
Sub-cuarto de CIF	128	96	64	48
cuarto de CIF	176	144	88	72
CIF	352	288	176	144
4 veces CIF	704	576	352	288
16 veces CIF	1408	1152	704	576

**Tabla 2.2 Número de píxeles por línea y número de líneas de cada formato de imagen H.263 normalizado.** <sup>15</sup>

Para el formato de cuadro se empleará QCIF, por lo que la resolución de la ventana de video deberá ser, según la tabla 2.2, de 176 x 144 píxeles, cuyo espacio de color será  $Y C_R C_B$ , de forma que se cumpla la recomendación H.263.

### 2.1.2 El codec de audio

La recomendación H.263 hace referencia a la recomendación ITU G.711 en lo referente a la codificación de audio.

Esta recomendación define que la codificación se la hará por modulación de impulsos, y se emplearán la ley A ó la ley  $\mu$  para mejorar la respuesta de frecuencia de la voz humana.

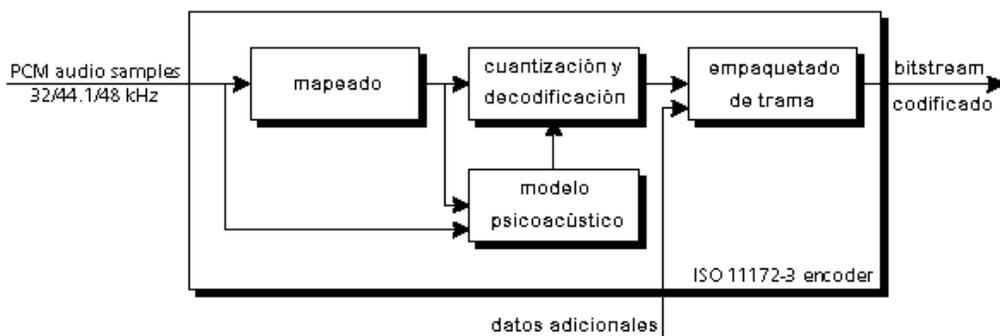


Figura 2.34 Diagrama de bloques del codificador.<sup>16</sup>

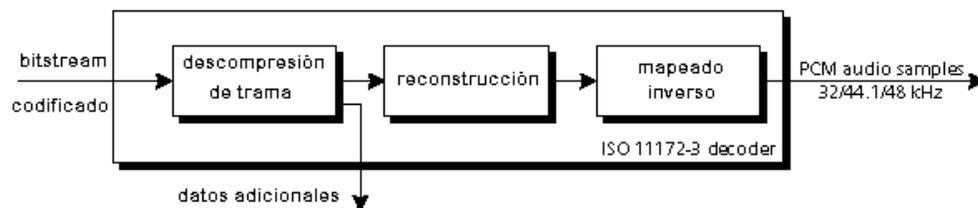


Figura 2.35 Diagrama de bloques del decodificador.<sup>16</sup>

La recomendación también sugiere que se emplee con preferencia la ley A.

<sup>16</sup> ITU-T G.711, 1988

La presente tesis de grado implementará entonces la recomendación G.711 empleando la ley A.

### **2.1.3 La captura del video**

Antes de tratar de definir la tecnología que se empleará para la captura de video, primero es necesario orientar la aplicación a la plataforma bajo la cual será desarrollada, sea esta Windows®, Linux, OS-X® o todas las mencionadas.

El sistema se orienta a proporcionar video conferencia de forma económica, y sin demasiada complicación. Por lo tanto esta se encuentra orientada a la realidad socio – económica de nuestro país, este limitante permite fácilmente escoger la plataforma. Para comenzar se descarta la plataforma OS-X de Apple, debido a sus costos, aunque es mucho más amigable con el usuario que las otras dos. También se descarta a Linux, el cual aunque es gratis en muchas de sus distribuciones, no es fácil trabajar con el, ya que requiere bastante más conocimientos de computación que las otras dos plataformas, por ejemplo no existen instaladores como tales, sino tar-balls, los cuales se tiene que descomprimir, y en muchos casos compilar, para luego de configurar la aplicación, hacerla finalmente utilizable. Por lo tanto la plataforma escogida es Windows de Microsoft. Se tiene ahora una nueva consideración respecto a esta plataforma, y es que existen varias versiones de la misma: Windows 95, Windows 98, Windows 98 SE, Windows Millenium, Windows 2000, y Windows XP. Las cuales tienen dentro de si algunas diferencias que pueden ser importantes como el tamaño de la palabra con la que trabajan, así por ejemplo Windows 98 SE es un sistema operativo de 16 bits de palabra, mientras que Windows XP es de 32 y 64 bits.

Uno de los objetivos de la aplicación es que esta pueda correr bajo la mayoría de las versiones de Windows, por lo cual se requiere emplear las APIs (Aplication Programming Interfaces) que sean comunes a todas las plataformas, y cuya complejidad no sea excesiva. Se tiene dos opciones Video for Windows (VFW) y Direct X.

Para decidir se analizarán sus ventajas y sus limitaciones.

## Direct X

### Las ventajas:

- Direct Show es una versión evolucionada de Video for Windows
- Emplea la tecnología Windows Device Model, para el control de los dispositivos, la cual emplea mini drivers para cada dispositivo específico.
- Maneja un mayor número de opciones.
- Se puede acceder a cualquier dispositivo, a través de su driver.

### Las limitaciones:

- La última versión del código ejecutable de Direct X se lo debe descargar periódicamente del Internet, y el instalador no es pequeño.
- Usualmente existen incompatibilidades entre versiones.
- Funciona óptimamente con aceleración de Hardware.
- Es relativamente complejo de utilizar.

## Video for Windows

### Las ventajas:

- Las librerías vienen instaladas por defecto en todas las versiones de Windows.
- No ha sido modificado desde hace algunos años, por lo que no es posible tener problemas de incompatibilidad.
- Se puede acceder a cualquier dispositivo, una vez que sus drivers estén instalados.
- Se puede acceder directamente a los dispositivos.
- Es relativamente fácil de emplear.

### Las limitaciones:

- Es antiguo, y por lo tanto algo obsoleto, debido a su popularidad anterior se mantiene, de hecho Direct Show tiene un filtro para Video for Windows.
- Sus librerías no son muy evolucionadas.

- Tiene un soporte limitado por parte de Microsoft.

Analizando lo dicho anteriormente y teniendo en mente adicionalmente el objetivo de hacer a la aplicación lo más ampliamente distribuible posible, Video for Windows contiene las APIs más adaptables al desarrollo de la presente Tesis de Grado.

#### **2.1.4 La captura del audio**

Una vez definida la tecnología para la captura del video, también se encuentra definida la tecnología que se empleará para la captura del audio, ya que tanto audio como video se encuentran íntimamente ligados en ambas tecnologías, por lo tanto es claro que Video for Windows también se empleará para la captura de audio.

#### **2.1.5 El canal de datos**

Como se hizo anteriormente, primero es necesario definir como se desea que opere el sistema antes de adelantar los detalles del medio de transporte.

En primer lugar es necesario que el sistema sea punto a punto, ya que se trata de un prototipo, se requiere que la comunicación sea bidireccional y que sea capaz de operar en cualquier conexión Ethernet, ya que se necesita que sea posible que cualquier persona emplee esta tecnología.

Por otra parte una conexión punto a punto nos brinda la seguridad suficiente para la operación, ya que no es posible que un tercero no consentido ingrese en medio de la comunicación ya establecida. No al menos sin elevados conocimientos del protocolo de transporte.

Pues bien, con estas limitantes, existen dos protocolos que son candidatos para el transporte de la información entre equipos, y estos protocolos son TCP/IP y RTP/RTCP.

Como siempre en primer lugar se analizarán sus ventajas y desventajas para escoger el protocolo que más conveniente.

RTP/RTCP, por *Real Time transport Protocol/Real time Transport Control Protocol*

Las ventajas:

- Estos protocolos definidos en la RFC3551, fueron especialmente diseñados para el transporte de información multimedia.
- Ofrecen control de calidad de servicio.
- Están diseñados para transportar la trama ITU H.261
- Es relativamente nuevo y está mejorando continuamente.

Las limitaciones

- Debido a que relativamente nuevo no existe mucha implementación de este protocolo.
- Es un protocolo de capa aplicación, por lo tanto o hay que comprarlo o hay que implementarlo.

TCP/IP Por *Transport Control Protocol/Internet Protocol*:

Las ventajas

- Es sumamente flexible
- Existe mucho soporte para este juego de protocolos
- La trama definida en ITU H.262, fue pensada para que sea transportada por este protocolo.
- Es un protocolo orientado a conexión

Las limitaciones:

- No fue específicamente diseñado para la tarea de transporte de multimedios.
- Su evolución se ha frenado dando paso a IPv6

Ya que se desea que la mayor cantidad de personas tengan acceso al presente sistema y que este se trata específicamente del diseño e implementación de un sistema de video conferencia y no de la implementación de un protocolo, además que se facilitan un poco las cosas, ya que la trama de la recomendación H.262 lo

tiene como base se escoge a TCP/IP como medio de transporte del flujo de datos de la aplicación.

De lo dicho anteriormente, el protocolo será punto a punto, bidireccional, y como se ha escogido TCP/IP es posible agregar orientado a conexión. Por lo tanto claramente se elige a Windows Sockets® como el método a emplear para comunicar los dos equipos de computo.

## CAPITULO 3

### 3.1 DISEÑO DEL SISTEMA

#### 3.1.1 LAS HERRAMIENTAS DE DESARROLLO

Es en esta etapa de diseño dónde se deberá escoger las herramientas que permitirán desarrollar las aplicaciones. Y se ha dicho “las” porque se desarrollará una aplicación complementaria a la video conferencia.

Pues bien se tienen a disposición varias herramientas de desarrollo como son Visual Studio® 6, Visual Studio NET®, Java®, y Delphi®. En este caso no se hará una comparación entre ventajas y desventajas, sino más bien iremos descartando las herramientas que no se ajuste al esquema de desarrollo previo a un análisis.

En primer lugar y antes de cualquier otra cosa es necesario definir claramente este esquema:

Para el sistema de video conferencia se requiere que:

- La aplicación se ejecute rápidamente a pesar del alto número de cálculos que realizará.
- La instalación no requiere librerías más allá de las propias del sistema operativo para funcionar. Prácticamente que no requiera instalador.
- Sea una aplicación ejecutable.
- La plataforma de desarrollo (IDE), provea todas las facilidades necesarias.
- La aplicación corra en la mayor parte de sistemas operativos de Microsoft.
- 

Para el sistema de presentación remota se requiere que:

- Sea rápida y sencilla de desarrollar.
- Corra con el menor número de librerías adicionales.

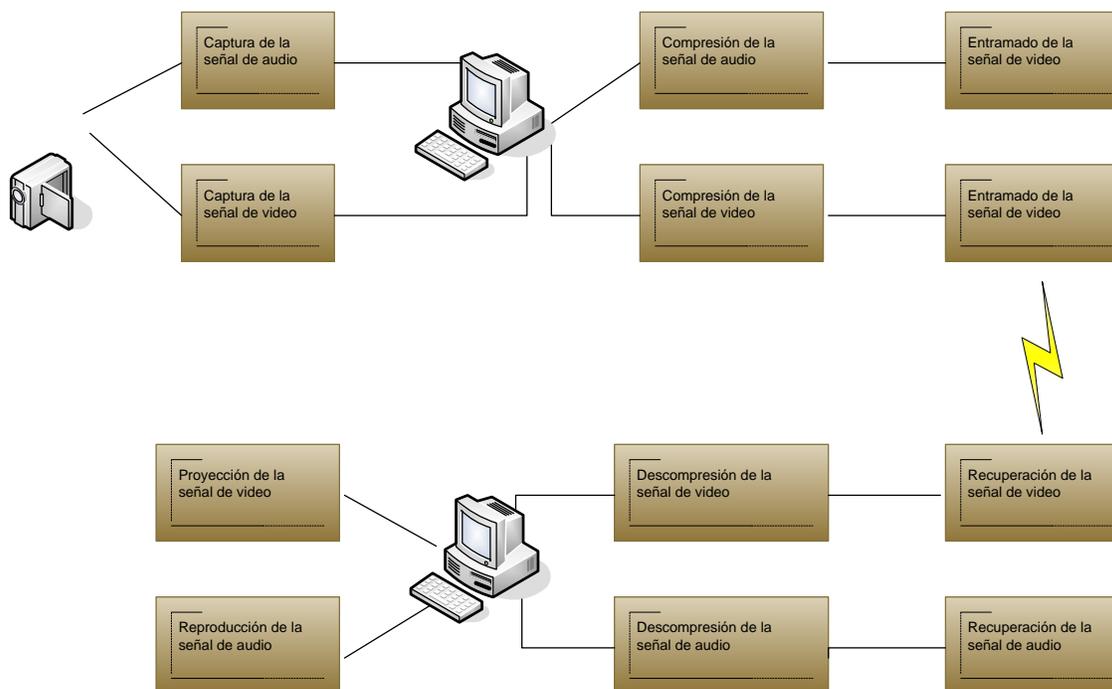
- Sea posible tener acceso a las API de Windows, de una forma sencilla.

Analizando la herramienta que más conveniente para el desarrollo de la video conferencia, se descarta a Java®, de inmediato por que al compilar la aplicación el compilador como resultado del proceso no proporciona un ejecutable sino una compilación intermedia en un archivo tipo .jar. Se descarta Visual Studio NET, debido a que se requiere que la aplicación corra con las librerías existentes en cada equipo, es decir que prácticamente no se necesiten de instaladores. Quedan entonces Visual Studio 6® y Delphi®. De Visual Studio 6 solo se tomará en cuenta Visual C++®, ya que el resto de herramientas de desarrollo, incumplen con alguna o varias de las características requeridas. De modo que restan Visual C++® y Delphi®, ambos lenguajes cumplen con todos los requerimientos, queda a favor de C++, que los algoritmos se encuentran en su mayor parte descritos en C++, por lo tanto elegimos Visual C++.

Para el caso de la presentación remota, se descarta en primer lugar Java® por similares razones, lo mismo ocurre con Visual Studio Net. Quedando unicamente Visual Studio 6® y Delphi®. La herramienta que más conviene, de Visual Studio 6 es Visual Basic, por ser una herramienta de desarrollo muy rápida y sencilla, dejando de lado las demás o bien porque su entorno no se aplica al objetivo, de tener una aplicación ejecutable, o bien por que su programación visual es muy larga y tediosa. De entre las dos, se elige a Visual Basic, justamente por ser la herramienta más simple para desarrollo aun cuando no es totalmente orientada a objetos como lo es Delphi®, y con ello balancear un poco los tiempos de desarrollo.

### 3.1.2 EL ESQUEMA GENERAL DE LA APLICACIÓN DE VIDEOCONFERENCIA

En primer lugar se debe tener una visión general de la estructura que se desea para el sistema de video conferencia, para a partir de ahí ir entrando en detalles sobre el diseño de cada parte de modo que se cumpla con los objetivos expuestos en el capítulo anterior, y previamente en este capítulo.

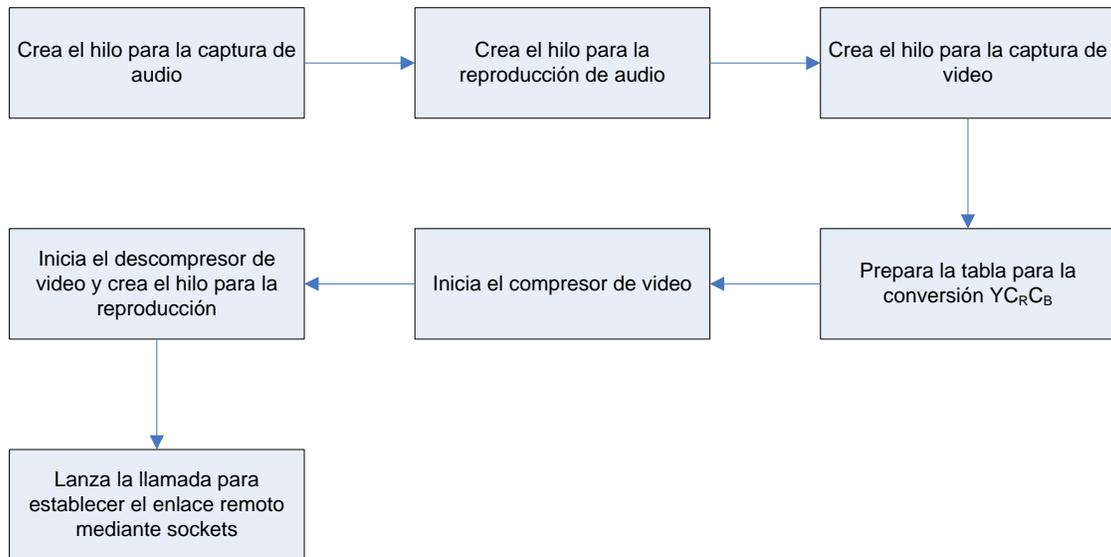


**Figura 3.1 El sistema de video conferencia en bloques**

La figura 3.1 muestra de forma muy breve la estructura que se necesita para el sistema, la misma que refleja claramente los objetivos planteados.

### 3.1.3 LA INICIALIZACIÓN DEL SISTEMA

Todo programa al ejecutarse requiere correr ciertos procesos para que el resto de la aplicación se corra adecuadamente. Por lo tanto se necesita detallar los procesos y el orden que seguirán, preparando los recursos necesarios para las otras partes de la aplicación.



**Figura 3.2 Esquema general de inicio de procesos**

En la figura 3.2 se puede observar que para incrementar el rendimiento del sistema se necesita planear una programación multi-hilos, es decir se requiere crear un procesamiento paralelo (real con múltiples procesadores, temporizado con un único procesador) en el cual se maneje separadamente el audio y el video. En el bloque que crea el hilo para la captura de audio, se verificará si existe hardware disponible para esta función (a través de los drivers), para luego iniciar apropiadamente el dispositivo.

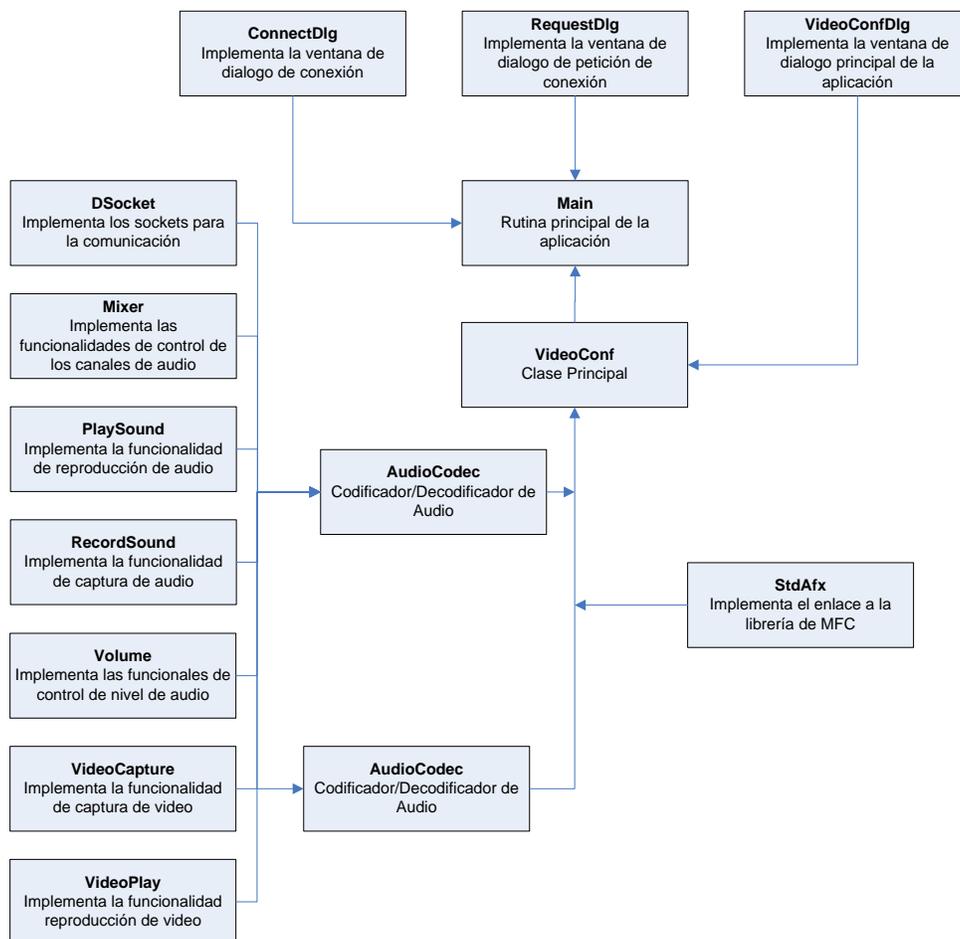
En el bloque de captura de video realizamos igualmente una verificación del hardware no solo para comprobar su existencia (si se encuentra conectada al equipo), sino también la verificación relacionada a que si el equipo soportará la resolución, y en la relación de aspecto que han sido planteadas.

Seguidamente se preparan las tablas de conversión del espacio de color, esto es se crea la matriz de cuantificación y además se asigna el espacio de memoria suficiente para manejar los bloques de color. El paso siguiente es iniciar el

compresor de video y audio, esto básicamente es configuración de variables, de forma similar lo hace el siguiente bloque, la inicialización del descompresor, pero adicionalmente se crea un hilo que manejará la reproducción del video, finalmente el sistema estará listo para mostrar la pantalla de inicio al usuario, y esperará iniciar la conexión con el equipo remoto.

### 3.1.3 LA ESTRUCTURA GENERAL DEL SISTEMA

Con las bases claramente definidas es posible crear la estructura general que tendrá la aplicación.



**Figura 3.3 Estructura general del sistema**

Como todo programa escrito en C++, el sistema tiene su rutina principal, denominada Main, la clase principal se la denominará VideoConf, la cual se

encargará de administrar toda la operación. Para no complicar demasiado la programación y además siendo el presente sistema un prototipo no se definirán ventanas como tales, sino mas bien cajas de dialogo, debido a que estas tienen un entorno visual en C++, y no requieren mayor esfuerzo en programación como sería implementar ventanas y sus artefactos, así como administrar sus eventos. Estas cajas de dialogo se las define de forma separada y son: ConnectDialog, la caja de dialogo que se encargará de pedir al usuario la dirección IP del equipo remoto con el cual conectarse, RequestDialog, es la que muestra la solicitud de conexión en el equipo remoto, realizada por el equipo local y finalmente VideoConf, que es la caja de dialogo que hará las veces de ventana principal de la aplicación.

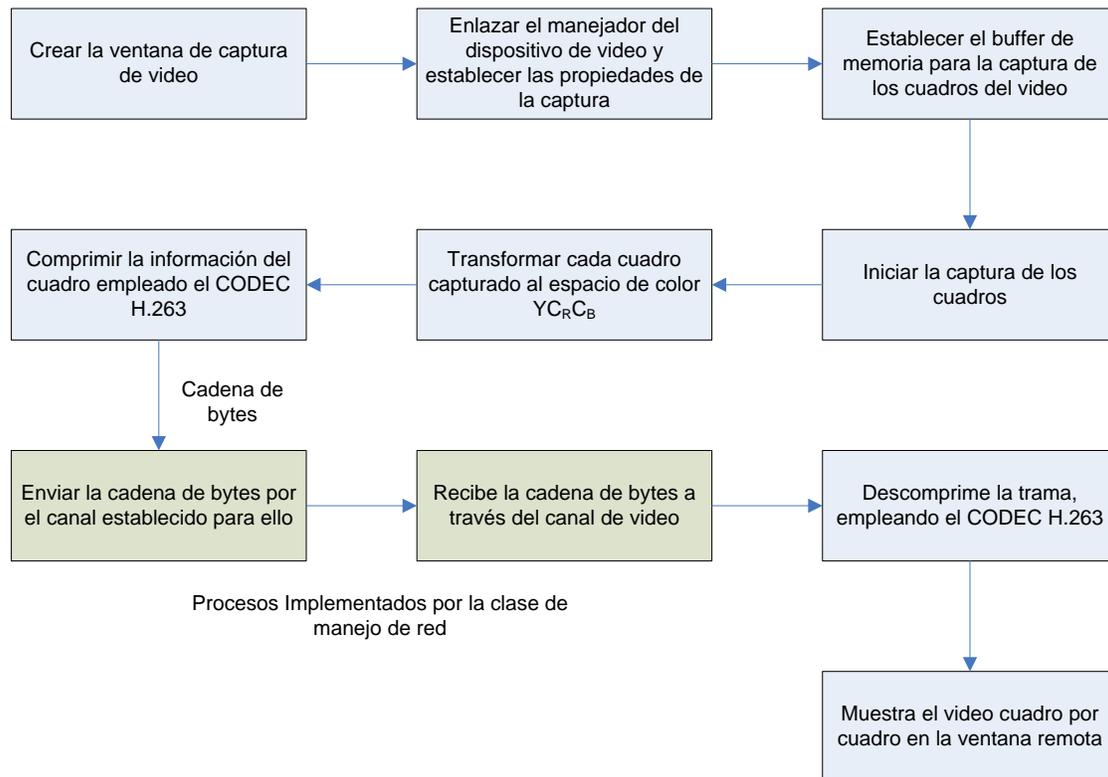
Por otra parte se tienen las clases principales que constituirán la aplicación, estas son: DSocket, que se encargará del manejo del canal de datos; Mixer, que implementará las funcionalidades de Windows® para el control del audio junto con la clase Volume; PlaySound y RecordSound que se encargarán de la reproducción y de la captura de audio; así mismo se tiene sus equivalente para video, estas son VideoCapture y VideoPlay, como se puede observar tanto las clases de manejo de audio como aquellas destinadas al manejo de video enviarán su información o la recuperarán de sus respectivos codecs, los cuales será definidos separadamente como codificador y decodificador y en archivo separados, implementando las funcionalidades requeridas.

### **3.1.4 EL CODEC DE VIDEO**

En primer lugar se deben definir los parámetros generales para la operación de esta parte del sistema. Los mismos que básicamente ya fueron mencionados en el capítulo anterior:

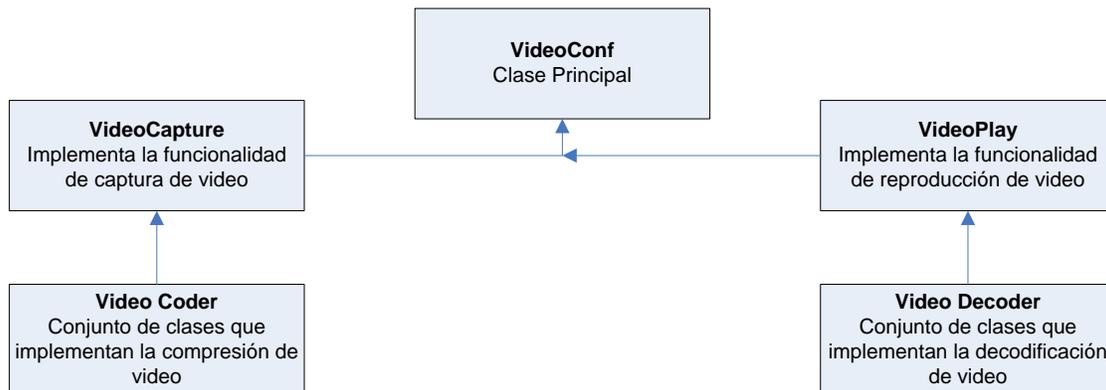
- Compresión de color mediante sub-muestro en  $Y C_R C_B$  4:2:2
- Relación de aspecto 4:3
- Cumple con QCIF
- Estimación de movimiento en base a Correlación Espacial Jerárquica.

➤ Compresión de imágenes Inter / Intra



**Figura 3.4 Diseño general del sistema de video**

En la figura 3.4, los dos primeros bloques le corresponde a la sección de captura de video. El siguiente bloque, el correspondiente al buffer de video, es el bloque en el cual se prepara la memoria para realizar la estimación de movimiento por una parte y la compresión (sub-muestreo) del color por otra, el siguiente bloque es el encargado de capturar cada cuadro en memoria para su proceso, en el siguiente bloque, de compresión de color, y en el siguiente de compresión DCT + Huffman, luego de haber sido estimados los vectores de movimiento y con todo ello, crear el entramado que será transmitido hacia el equipo cliente, dónde ocurrirá el proceso inverso.



**Figura 3.5 Clases generales del sistema de video**

En la figura 3.6 se puede observar más en detalle los procesos que se llevarán a cabo en el codec de video.

De cada proceso ya ha sido explicado anteriormente su procedimiento de cálculo excepto de la DCT:

Una secuencia DCT  $x(n), (n = 0, \dots, N-1)$ , puede ser expresada por la FFT, para ello primero se define una nueva secuencia  $y(n), (n = 0, \dots, N-1)$ , donde

$$\begin{cases} y(n) \equiv x(m) \\ y(N-1-m) \equiv x(2m+1) \end{cases} \left( i = 0, \dots, \frac{N}{2}-1 \right)$$

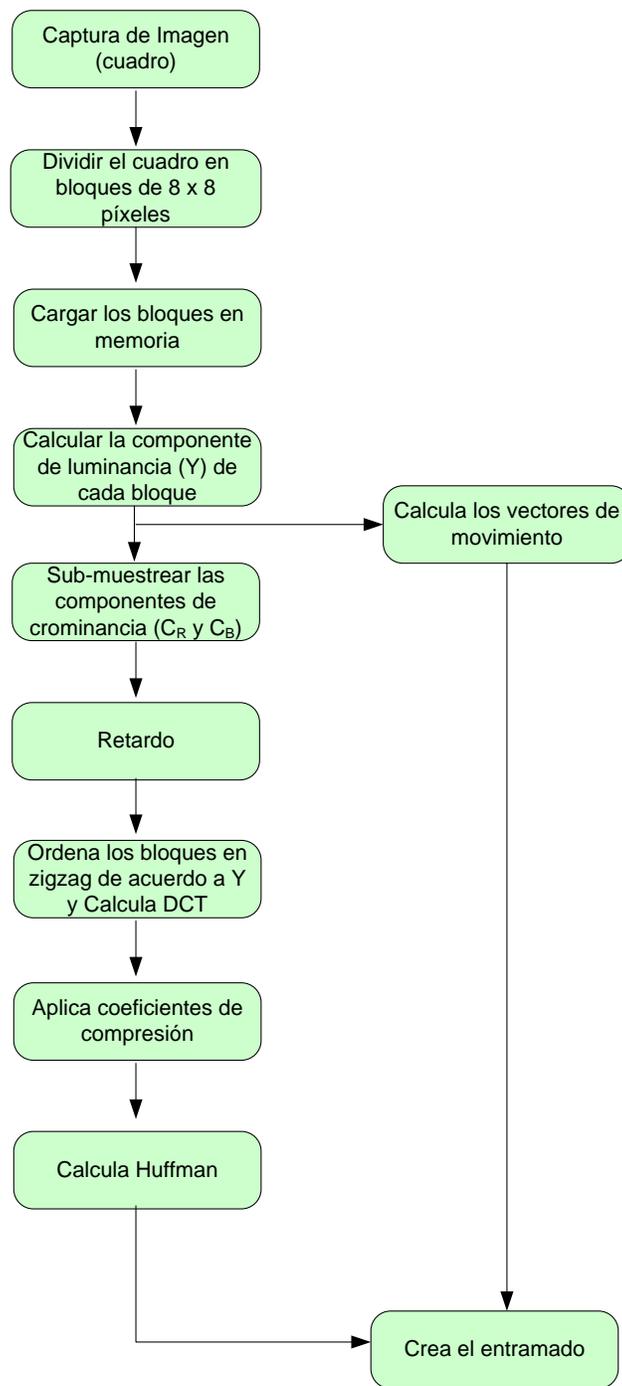
Entonces la DCT de  $x(n)$ , puede ser escrita de la siguiente manera:

$$X(n) = \sum_{m=0}^{N-1} x(m) \cos\left(\frac{(m+1)n\pi}{2N}\right)$$

O también:

$$\sum_{m=0}^{N/2-1} x(m) \cos\left(\frac{(m+1)n\pi}{2N}\right) + \sum_{m=0}^{N/2-1} x(m+1) \cos\left(\frac{(m+3)n\pi}{2N}\right)$$

En la ecuación el primer componente representa a los términos pares y el segundo a los términos impares.



**Figura 3.6 Diagrama de bloques del codificador**

Se define para la segunda sumatoria  $m' \equiv N-1-m$ , entonces los límites de la sumatoria van desde 0 hasta  $\frac{N}{2}-1$ , para  $m$  viene a ser  $N-1$  y  $\frac{N}{2}$  para  $m'$ , por lo que la segunda sumatoria puede ser escrita así:

$$\sum_{m=N/2}^{N-1} y(n) \cos\left(2\pi m - \frac{(m'+1)n\pi}{2N}\right) = \sum_{m'=n/2}^{N-1} y(n) \cos\left(\frac{(m'+1)n\pi}{2N}\right), \text{ puesto que:}$$

$$\cos(\alpha - \beta) = \cos\alpha \cdot \cos\beta + \sin\alpha \cdot \sin\beta$$

Combinando la expresión de la sumatoria de  $X(n)$ , tenemos:

$$X(n) = \sum_{m=0}^{N-1} y(n) \cos\left(\frac{(m+1)n\pi}{2N}\right)$$

Consideremos ahora la DFT (*Discrete Fourier Transform*) de  $y(n)$ :

$$Y(n) = \sum_{m=0}^{N-1} y(n) e^{-j2\pi mn/N} = \sum_{m=0}^{N-1} y(n) \cos\left(\frac{2\pi mn}{N}\right) - j \sin\left(\frac{2\pi mn}{N}\right)$$

Multiplicando ambos lados por:

$$e^{-jn\pi/2N} = \cos\left(\frac{n\pi}{2N}\right) - j \sin\left(\frac{n\pi}{2N}\right)$$

Y tomando la parte real del resultado, tenemos:

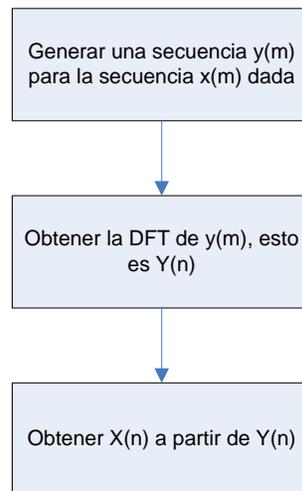
$$\begin{aligned} \operatorname{Re} \left[ e^{-jn\pi/2N} \cdot Y(n) \right] &= \sum_{m=0}^{N-1} y(n) \left[ \cos\left(\frac{2\pi mn}{N}\right) \cos\left(\frac{n\pi}{2N}\right) - \sin\left(\frac{2\pi mn}{N}\right) \sin\left(\frac{n\pi}{2N}\right) \right] = \\ &= \sum_{m=0}^{N-1} y(m) \cos\left(\frac{(m+1)n\pi}{2N}\right) \end{aligned}$$

De modo que la DCT puede ser calculada como:

$$X(n) = \operatorname{Re} \left[ e^{-jn\pi/2N} \cdot Y(n) \right]$$

Dónde  $Y(n)$  es la DFT de  $y(n)$

Del argumento matemático dado hasta aquí se puede definir un algoritmo que permitirá calcular la DCT:



**Figura 3.7 Algoritmo para obtener la DCT<sup>17</sup>**

El algoritmo presentado para hacerlo bidimensional primero se debe aplicar a las filas y luego a las columnas.

Al resultado de este se aplica Huffman, para lograr una aún mejor compresión.

Hasta aquí solo queda por aclarar como funcionaría el decodificador; pues basta decir que básicamente es el proceso inverso, a partir de la recuperación de la información, en cuyos datos primero obtenemos la IDCT bidimensional, a partir del algoritmo de Chen-Wang, luego se multiplicará por la matriz de compresión y se obtendrá la matriz de luminancias, y también la información de crominancias, con esta información se reconstruye el cuadro en memoria y se lo proyecta en la ventana correspondiente.

### 3.1.5 EL CODEC DE AUDIO

La codificación de audio cumple con ITU G.711, “Modulación de Impulsos Codificados (*Pulse Coded Modulation*) de Frecuencias Vocales”. Esta es una recomendación para el muestreo digital de la voz, en la cual se determina que el valor del muestreo debe ser de 8000 muestras por segundo con una tolerancia de

<sup>17</sup> Electronic Engineers Handbook, Fourth Edition, Donald Christiansen, McGraw-Hill, 1996

$\pm 50$  ppm, y se debe emplear la ley A para mejorar la respuesta de frecuencia de la entrada. La codificación debe ser de 8 bits.

El algoritmo del codificador como se puede observar es bastante simple.

En el caso del decodificador, se tiene como entrada los valores digitalizados por el muestreo PCM, debido a que lo que se obtiene es una señal cuadrada (un valor binario), la señal de audio original se ve distorsionada por la señal de la frecuencia de muestreo cuadrada, que podría requerir que se eliminen la primera y quinta armónicas para una mejor fidelidad mediante un filtro digital, pero en el presente sistema, esto no será tan necesario porque se incremento la respuesta de frecuencia de la voz empleando previamente al muestreo digital, la ley A.

$$x(t) = \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{\text{Sen}((k-1)2\pi ft)}{2k-1} = \frac{4}{\pi} \left( \text{Sen}(\pi ft) - \frac{1}{3} \text{Sen}(3\pi ft) + \frac{1}{5} \text{Sen}(5\pi ft) - \dots \right)^{18}$$

**Figura 3.8 Las componentes armónicas de una señal cuadrada**

### 3.1.6 LA CAPTURA DEL VIDEO

La captura de la señal de video tiene que ser independiente del hardware, esto es debe ser administrado por el sistema operativo, el mismo que para el efecto permite la instalación de los manejadores de los dispositivos, los cuales identifican automáticamente al hardware y lo conectan de una forma transparente con las librerías del sistema operativo, y es a través de estas que se tendrá acceso a las funcionalidades del hardware.

Estas librerías tienen definidas un conjunto de funciones y procedimientos (por eso se las conoce como librerías) que permiten la comunicación con el hardware, a este tipo de librerías se las conoce como API (*Application Programming Interface*), y por lo tanto no es necesario trabajar directamente con el hardware, ya que simplemente se requiere llamar a un procedimiento o función de la API correcta

<sup>18</sup> Enciclopedia de la Electrónica Ingeniería y Técnica, McHugh James A. M., Vol. 7, Oceano-Centrum.

para poder leer la información de video que transmita el dispositivo, y es así como se realizará la captura de la señal de video.<sup>19</sup>

### **3.1.7 LA CAPTURA DEL AUDIO**

Igual que en el caso del video, la captura de la señal de audio se la realiza a través de las API del sistema operativo, Al requerirse mejorar la respuesta armónica de la voz y cumplir con la frecuencia de muestreo mencionada anteriormente, se requiere acceder directamente al hardware a través de su manejador (driver). Por lo que no se tiene mayor inconveniente con la captura del dispositivo de audio.

### **3.1.8 EL CANAL DE DATOS**

Tal como se mencionó en el capítulo anterior será implementado con Windows Sockets®, esto es, se necesita de una dirección IP y de un puerto para la conexión. La dirección IP que se utilizará se la podría solicitar al momento de realizar la conexión, siendo la dirección designada al equipo, o se la podría tener almacenada en un archivo de configuración, o se la podría detectar en cada equipo. La segunda opción requeriría de más desarrollo, una ventana de edición y salvar los cambios, probar que los datos ingresados sean válidos y las operaciones salvar y recuperar archivos, así también la tercera. La única ventaja que presentan al usuario es no tener que recordar la dirección IP, pero al ser este un prototipo práctico no se trabajará en las facilidades para el usuario, supondremos que el mismo tiene algún conocimiento sobre redes y su configuración, por lo tanto se descartan estas opciones.

Adicionalmente se puede observar que el canal de datos debe transmitir video, audio y señales de control. Todas las señales podrían transmitirse serialmente de forma síncrona (o por paquetes), o podrían ser transmitidas de forma paralela asíncrona. Debido a la velocidad que se requiere en el procesamiento y transmisión de la información y para que la video conferencia sea de buena

---

<sup>19</sup> MSDN, Microsoft Corporation, 2003

calidad, se selecciona la forma paralela asíncrona, en la cual se tiene un puerto para cada tipo de señal, esto es, se tendrá un puerto que maneje la señal de video, uno que maneje la señal de audio, uno que maneje las señales de control, y adicionamos uno que maneje la comunicación por texto (se incorporará una versión sencilla de Chat). La configuración de envío – recepción de paquetes, que sería la que cumpla fielmente con la estructura de paquetes de H.263, obligaría a implementar programación a nivel de capa 3, lo cual queda fuera del objetivo de la presente Tesis de Grado.

Como se dijo se va a configurar un puerto para cada tipo de comunicación, este puerto no tiene que pertenecer a los WKP (Well Known Ports) de IANA(Internet Assigned Number Authority), es decir tienen que ser números de puertos superiores as 1023, de 1024 a 49151 son reservados, de modo que tampoco se debería utilizar estos puertos, de 49152 as 65535, son de asignación dinámica, y pueden ser utilizados sin ningún inconveniente. El problema es que los programas antivirus normalmente revisan los puertos WKP y los de asignación dinámica en busca de programas maliciosos, por lo que se tomarán prestados unos puertos de la sección de reservados, esto son, los puertos 6000 (Control), 6002 (Audio), 6004 (Video), 6006 (Texto). Estos puertos están reservados por el sistema X Windows (Unix), por lo que se los puede utilizar bajo Windows® sin ningún inconveniente.

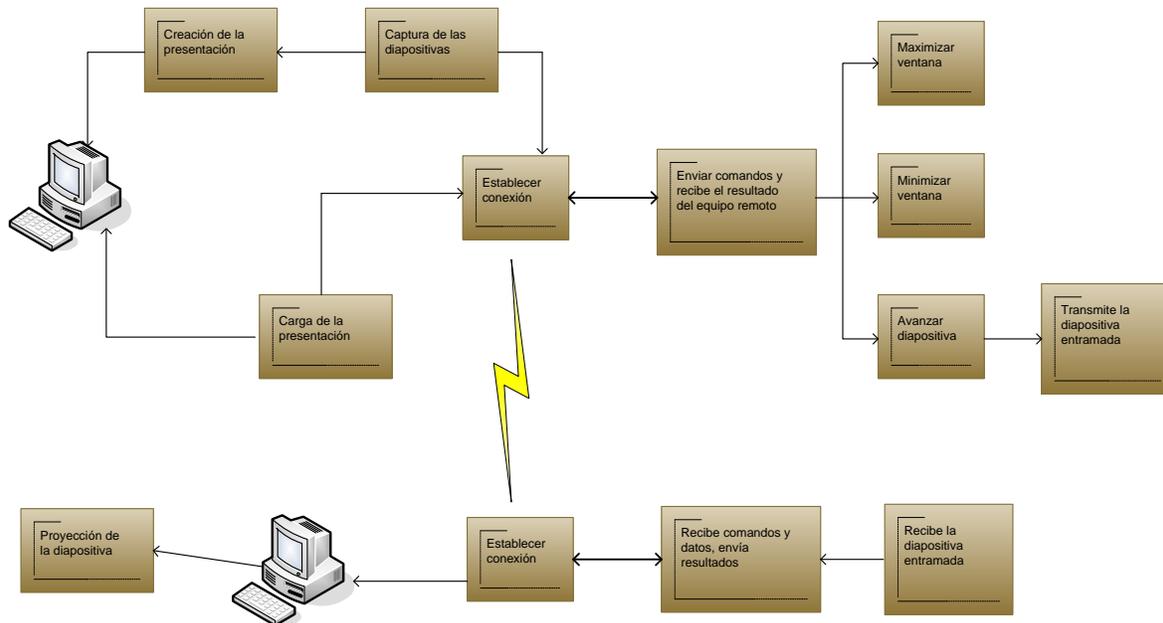
Definidos los parámetros de trabajo se determinarán las definiciones del sistema: Básicamente como se puede observar en la figura 3.3, todo lo referente a las comunicaciones de la aplicación lo manejará la clase DSocket, la cual se comunicará directamente con la clase principal.



**Figura 3.9 La estructura del canal de datos**

Respecto a esta clase queda por mencionar que se la implementará, trabajando directamente con las API de Windows® que proporcionan las funcionalidades de red, y no empleando librerías especializadas como MFC (*Microsoft Foundation Classes*).

### 3.1.10 LA APLICACIÓN DE PRESENTACIÓN REMOTA

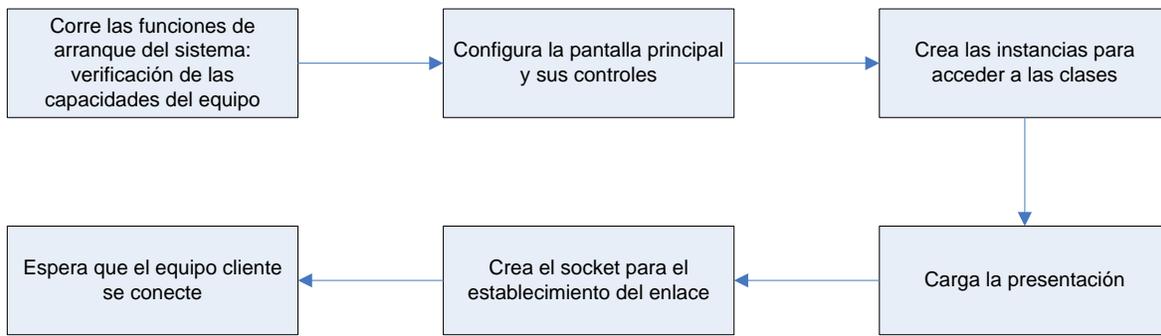


**Figura 3.10** La aplicación de presentación remota en bloques

Como se puede observar en la figura 3.10, el sistema de presentación remota se encuentra dividido en dos secciones, la primera sección que es la que contiene todo el control del sistema al cual se la denomina, el programa servidor, la segunda sección es la que recibirá la información y se limitará a desplegar la información transmitida, a la cual se la denomina, el programa cliente. En la figura también se puede observar que la aplicación servidor permitirá la creación de un archivo de presentación a partir de un conjunto de diapositivas, las cuales deberán estar almacenadas en una misma carpeta y tener el formato jpeg (*Joint Photographic Experts Group*), o bien permitirán cargar el archivo de presentación que contiene las diapositivas. Esta misma parte del sistema permitirá la transmisión al lado cliente de las diapositivas, así también permitirá controlar el

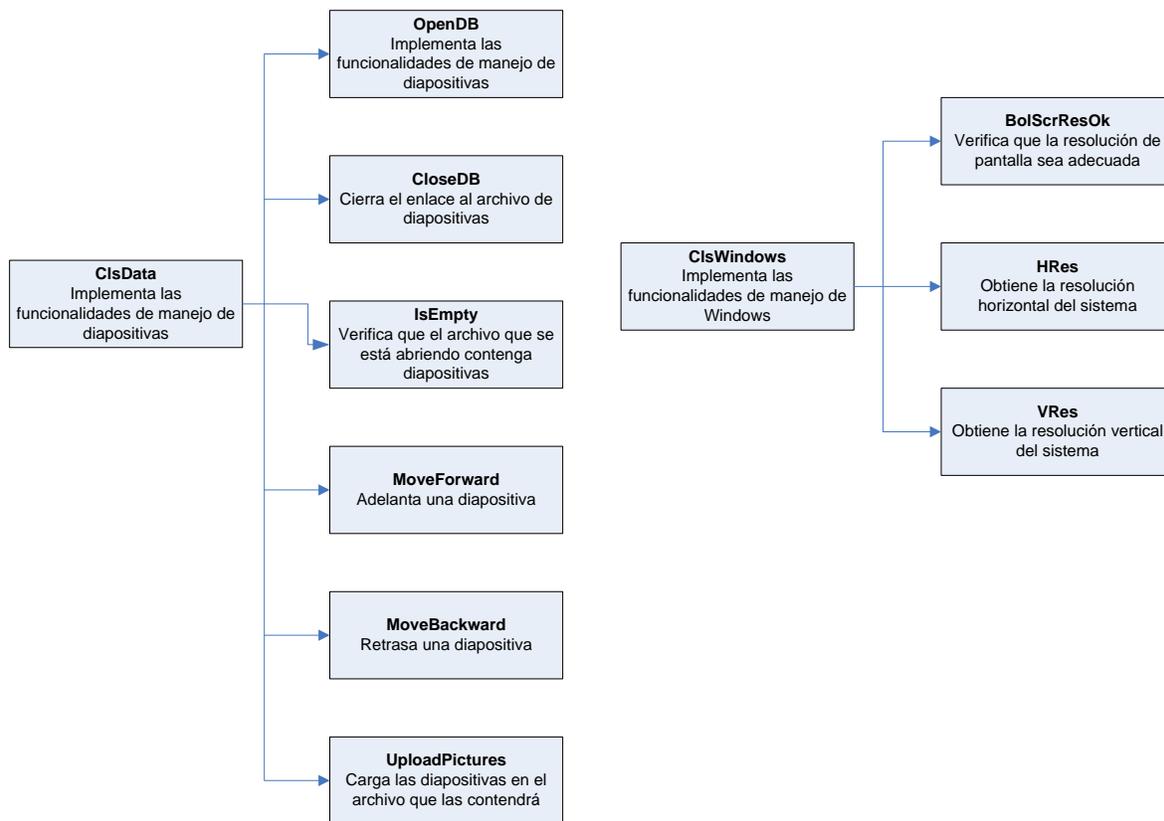
tamaño de la ventana en la cual se despliegan. Todo esto se lo hará estableciendo una conexión virtual entre el lado servidor y el lado cliente, mediante un único socket (transmisión tanto de datos como de señales de control), el cual será controlado de forma síncrona entre los lados del sistema, esto es requeriremos de un reloj de sincronismo.

Seguidamente se analizan las operaciones que realiza cada lado del sistema al iniciarse.



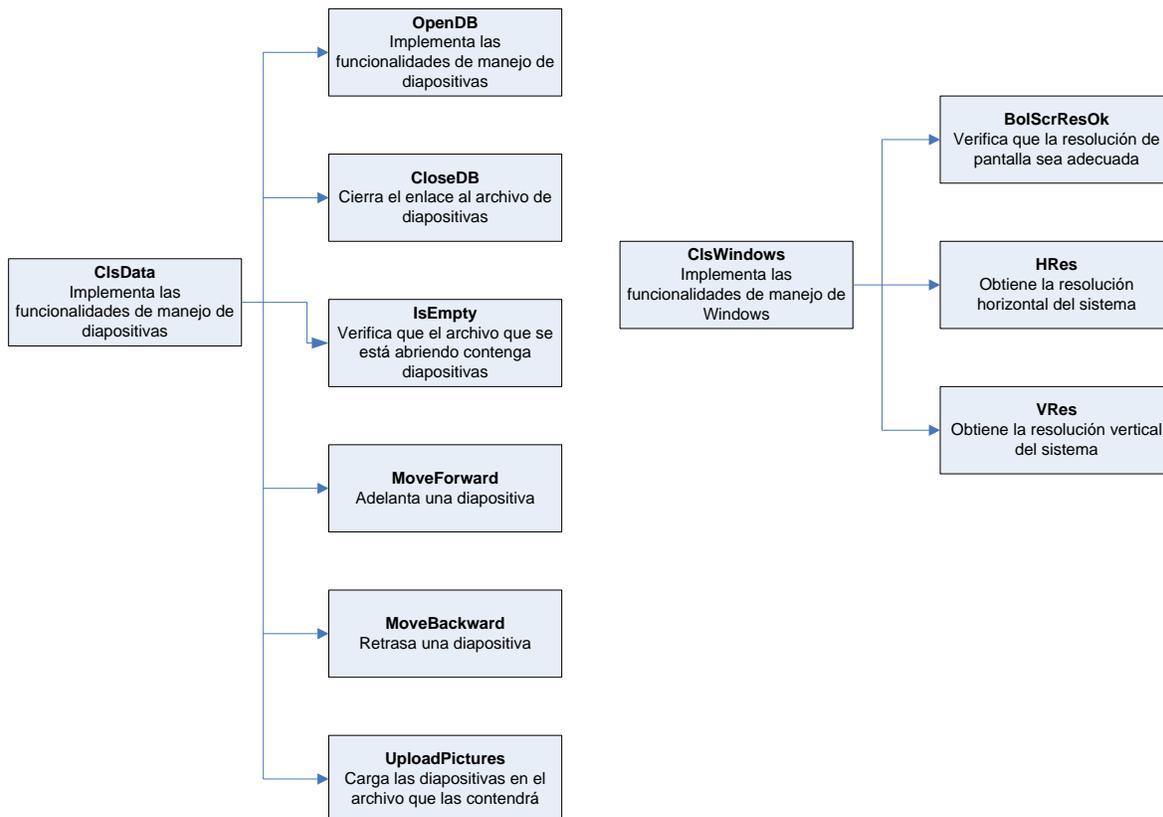
**Figura 3.11. Los procesos de inicio del lado servidor**

La figura 3.11 muestra en resumen, los procesos que correrá la aplicación servidor, para implementar todas las funcionalidades de las que se habló anteriormente, y son descritas en la figura 3.10. Todas estas funcionalidades son implementadas mediante clases y módulos, en la figura 3.12 se muestra como están organizadas estas clases.



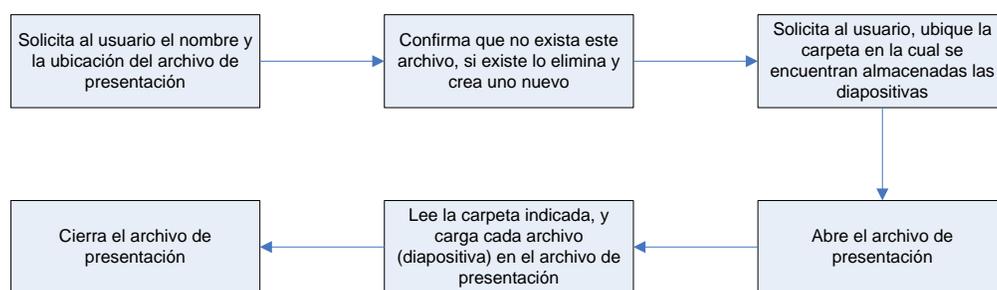
**Figura 3.12. Las clases de la aplicación del lado servidor**

Seguidamente se analiza en mayor profundidad las secciones de las que esta constituida la aplicación y como se emplean las clases que se muestran en la figura anterior.



**Figura 3.13. El procedimiento de creación de una nueva presentación**

El procedimiento de la creación de una nueva presentación será administrado por el procedimiento PresentationNew, el cual llamará a la clase clsData, más bien dicho a los métodos que esta implementará para efectuar la operación de la creación de una nueva presentación, esta operación es descrita en mayor detalle en la siguiente figura.



**Figura 3.14. La operación de creación de una nueva presentación**

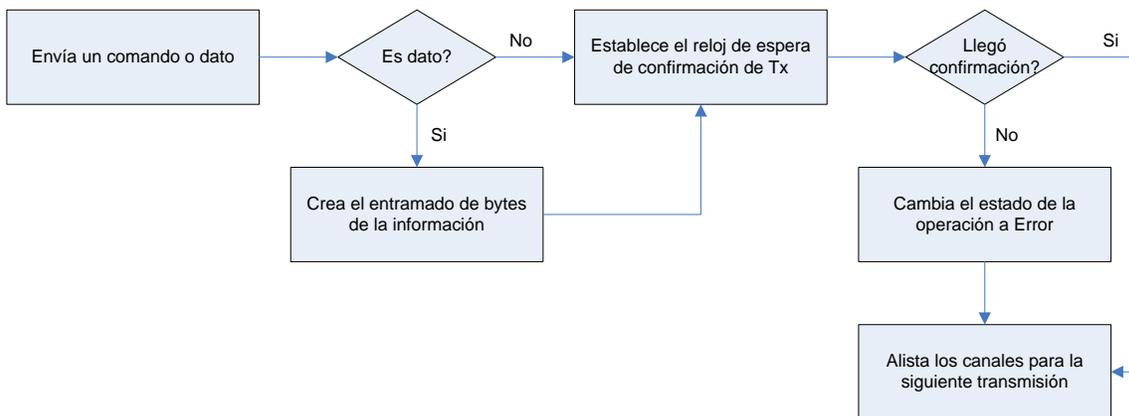
En primer lugar se le solicitará al usuario que le de un nombre y elija la carpeta donde se almacenará el archivo de presentación, este archivo de presentación no es más que un archivo de base de datos Microsoft Jet® (mejor conocida como Access), el cual se copia con otro nombre a partir de uno existente, en la ubicación elegida por el usuario, luego a este se le solicita que determine la carpeta que contiene las diapositivas que van a ser cargadas al archivo, para luego abriendo una conexión con esta base de datos, proceder a cargar las imágenes (diapositivas) una por una, para una vez realizada esta operación cerrar la conexión con la base de datos, habiendo quedado cargadas en la misma todas las diapositivas (imágenes de tipo jpeg).

Si ya existe la base de datos con la cual se desea trabajar, el procedimiento es algo diferente, únicamente se requiere establecer la conexión con la base de datos, una vez que se ha comprobado que esta no se encuentra vacía.



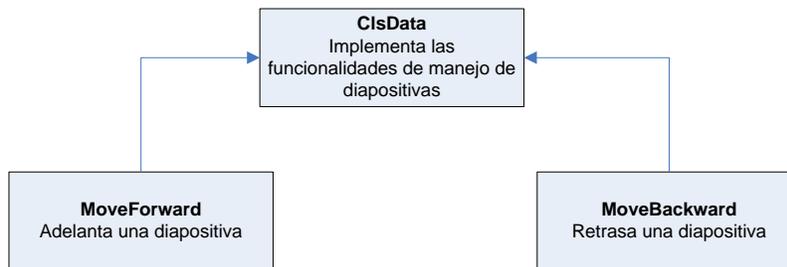
**Figura 3.15. El establecimiento de la conexión.**

A continuación se describe el procedimiento de la creación de un socket para la conexión virtual entre los dos lados (el lado servidor y el lado cliente): como se puede observar el lado servidor espera una solicitud de conexión de parte del lado cliente, esta se la maneja de una forma similar a la aplicación anterior, esto es, mediante las direcciones IP que tienen los equipos en la red.



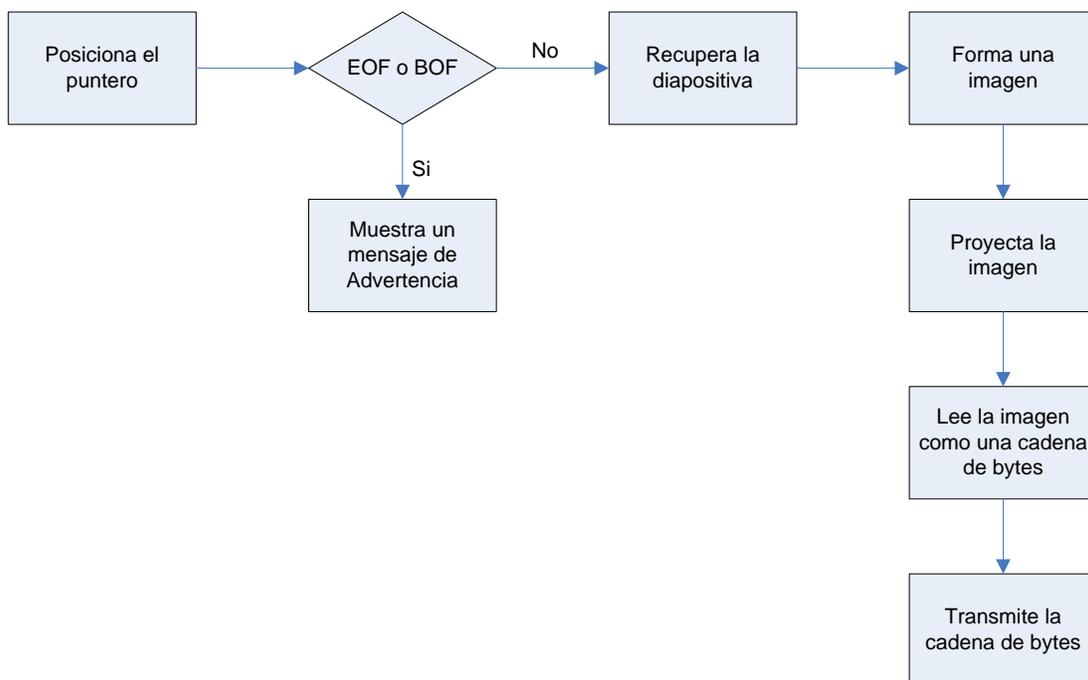
**Figura 3.16. El control de transmisión.**

Seguidamente se analizará la rutina de control de transmisión – recepción, la cual se puede observar esquematizada en la figura 3.16. El esquema nos muestra que la rutina funciona con un control basado en reloj, esta rutina únicamente se encarga de validar si la transmisión fue exitosa o no lo fue, más como se puede esperar, en caso de no haberlo sido, se limita simplemente a indicar al usuario, que se produjo un error en la transmisión, más no trata de volver a transmitir la información. Esta (la información) puede ser de dos tipos, paquetes de control, que son los comandos de control del tamaño de la ventana remota, o bien pueden ser datos, en este caso la transmisión de una diapositiva, si este es el caso antes de iniciar la transmisión, la información se entrama en una cadena de bytes (esto se lo analizará en más detalle, más adelante), la cual es transmitida toda en una misma trama, en el caso de ser un paquete de control, es igualmente transmitido como un único paquete. En ambos casos como se puede observar se dispara un reloj de control de transmisión, pero de diferente valor debido a la longitud de la trama.



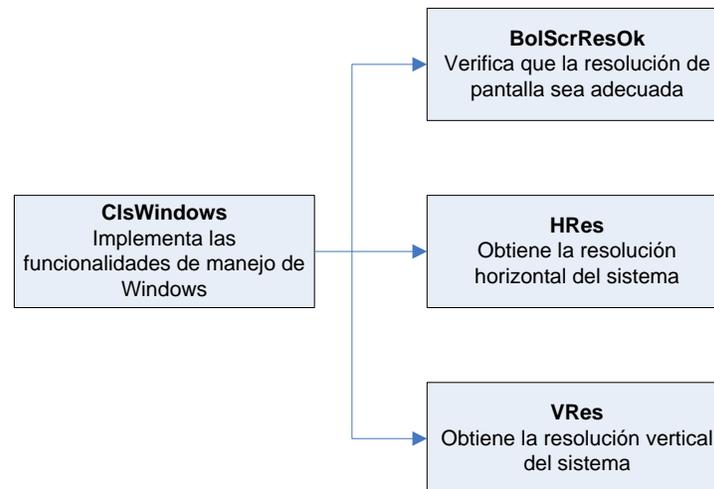
**Figura 3.17. La operación con las diapositivas.**

En la figura 3.17, se muestra un esquema de las clases empleadas para implementar las operaciones con las diapositivas, las cuales son dos, adelantar y retroceder, básicamente lo que estas clases implementan es una lectura del archivo de base de datos que contiene las diapositivas, además de mover el puntero que ubica cual diapositiva debe ser recuperada, y las demás funcionalidades de control que evitan que el puntero vaya demasiado adelante o demasiado atrás.



**Figura 3.18. Detalle de la operación con las diapositivas.**

De la figura 3.18 se desprende un análisis mas detallado de la operación con las diapositivas que vimos anteriormente, en el cual se puede observar el mecanismo de control que permite la navegación adelante y atrás, así como también el procedimiento que sigue la recuperación de una diapositiva desde el archivo de base de datos hasta su transmisión por medio del canal virtual, con lo cual concluimos el análisis de la aplicación servidor.



**Figura 3.19. Las clases de la aplicación cliente.**

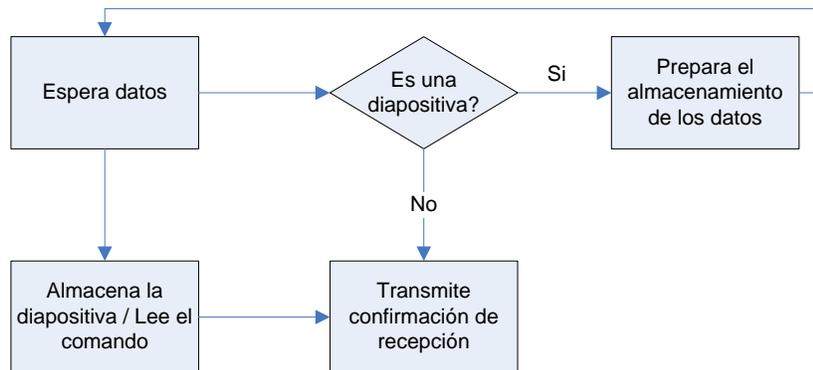
Como se puede observar en la figura las clases de la aplicación cliente solo incluyen el control de ventanas, esto se debe a que como se explicó anteriormente esta aplicación únicamente se encarga de recibir datos y transmitir confirmaciones desde y hacia la aplicación servidor, lo cual hace que esta aplicación sea realmente simple.



**Figura 3.20. Establecimiento de la conexión en el lado cliente.**

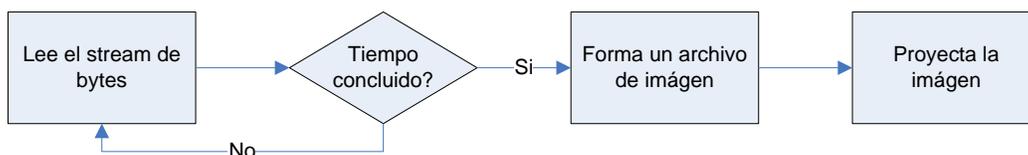
La aplicación como se puede observar en la figura 3.10 consta de los bloques de inicio de sesión y de recepción de información, el primer bloque se lo analiza en

mayor detalle en la figura 3.20, el cual como se podría haber esperado es bastante similar a su equivalente del lado servidor, con la diferencia que es este bloque el que realiza la solicitud de conexión, llamando a la dirección IP del equipo que corre la aplicación servidor. A diferencia de la aplicación de video conferencia esta conexión se la hace de forma unidireccional, ya que solo se requiere un sentido de transmisión, del servidor al cliente.



**Figura 3.21. La operación de Transmisión – Recepción.**

A continuación, en la figura 3.21 se puede observar, la forma en que se llevará a cabo la operación de transmisión – recepción de datos en el lado cliente, como se puede ver, esta distingue los datos entre las diapositivas y señales de control, mediante precisamente el uso de estas últimas que hacen la función de banderas que previenen a la aplicación del tipo de datos que serán transmitidos.

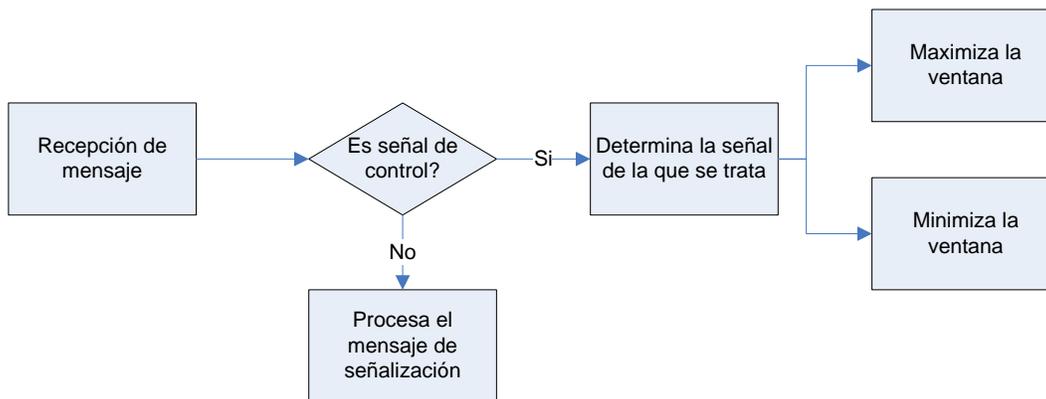


**Figura 3.22. La operación con las diapositivas en el lado cliente.**

En el siguiente bloque, el de la operación con las diapositivas, se detalla en la figura 3.22, se puede apreciar nuevamente la sincronización mediante el empleo

de un reloj para la recepción de la información. Una vez concluido el tiempo de espera, el cual fue calibrado para operar en una red de banda ancha, el sistema supone que toda la información de la imagen ha sido recibida y por lo tanto se dedica a formar el archivo de imagen a partir de la cadena de bytes recibidos para luego proyectarla por pantalla.

Falta por describir únicamente un pequeño bloque, el que nos permite manipulara el tamaño de la ventana, es decir el bloque de control del sistema.



**Figura 3.23. El procesamiento de las señales de control.**

Como se puede apreciar en la figura 3.23, la entrada de esta etapa es la recepción de la señal por parte del canal virtual, desde el cual se direcciona la señal a la etapa correspondiente como se pudo observar en la figura 3.21. Se procesa esta señal y se la separa en mensajes de señalización o mensajes de control, si el mensaje es de control, se determina de qué mensaje de control se trata y se lo procesa, realizando la acción que corresponda.

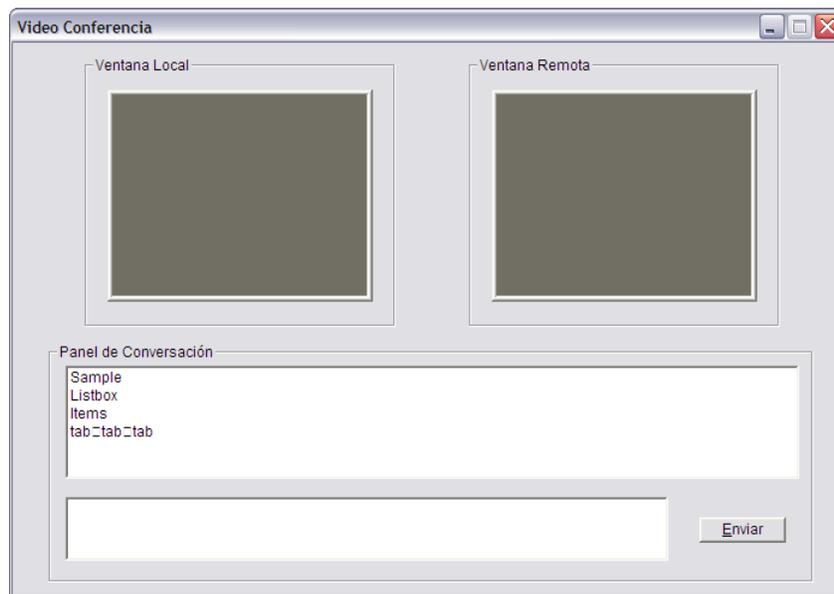
## **CAPITULO 4**

Debido a la gran complejidad que implica la implementación del proyecto de la presente Tesis de Grado, en este capítulo se ha tratado con poco detalle los algoritmos empleados para las diferentes funcionalidades que tiene la aplicación, funcionalidades que han sido explicadas sin demasiada profundidad en capítulos anteriores para facilitar la comprensión del funcionamiento de un sistema de video conferencia. Si el lector requiere adentrarse aún más en el conocimiento de cómo opera este tipo de sistemas, se sugiere que revise el código fuente adjunto a la presente Tesis de Grado.

### **4.1 EL SISTEMA DE VIDEO CONFERENCIA**

Como se analizó anteriormente, la implementación del sistema de video conferencia se la hará empleando Visual C++, este lenguaje de programación no proporciona muchas facilidades para el desarrollo de un entorno gráfico (cada ventana que ve el usuario, cada control y cada evento debe ser programado separadamente como una instancia de un objeto), por lo que se optó por emplear ventanas de dialogo, estas brindan un diseñador gráfico con el cual se puede visualizar la ventana y además ubicar los controles, como se lo haría en un entorno más amigable, como el de Visual Basic.

En primer lugar se implementará la ventana de dialogo principal, la cual muestra al usuario todas las funcionalidades que se le han dado a la aplicación:



**Figura 4.1 La ventana de dialogo principal.**

En esta ventana de dialogo se puede observar las funcionalidades que se le han dado a la aplicación, esto es, tiene un sector para la representación del video local, un sector para la representación del video que proviene del otro lado de la conexión, posee un área en la cual se muestra los mensajes enviados y recibidos, y un sector que permite justamente enviar un mensaje de texto, esta última funcionalidad se la mencionó brevemente en el análisis del capítulo anterior. Como se puede observar en la figura 4.1, esta ventana de dialogo no presenta un menú de opciones aunque realmente si lo tiene, esto se debe a que Visual C++, implementa el menú como un elemento separado, el cual se añade a la ventana de dialogo mediante programación, dicho esto, a continuación se listan las funcionalidades que se le han dado a la aplicación a través de su menú:

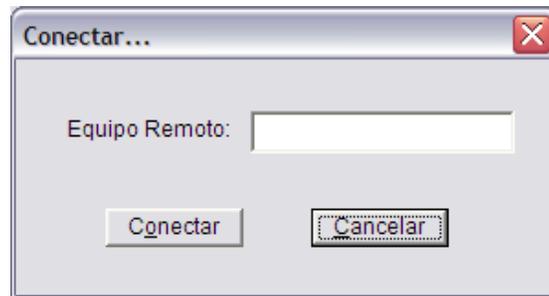


**Figura 4.2. El menú desplegable.**

Opción	Funcionalidad	Descripción
Archivo	Salir	Cierra la aplicación.
Conferencia	Conectar	Muestra una ventana de de dialogo para iniciar la conexión con el equipo remoto
	Desconectar	Termina la conexión con el equipo remoto
Ver	Venta de video local	Visualiza el área de la ventana que muestra el video local
	Ventana de video remoto	Visualiza el área de la ventana que muestra el video remoto
Audio	Control de volumen	Muestra una ventana de dialogo con la cual se puede calibrar los niveles de audio
	Enviar señal	Habilita la transmisión de la señal de audio
	Recibir señal	Habilita la recepción de la señal de audio
Video	Enviar señal	Habilita la transmisión de la señal de video
	Recibir señal	Habilita la recepción de la señal de video
Ayuda	Acerca de	Muestra una ventana de dialogo con información referente a la aplicación

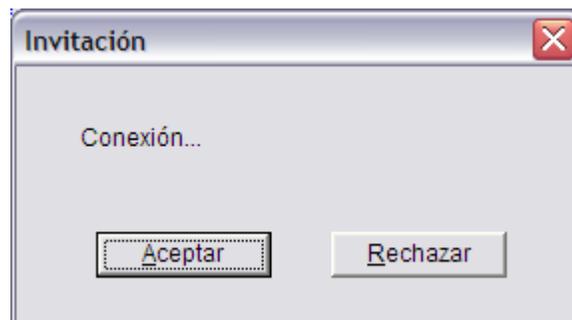
**Tabla 4.1. Descripción del menú de la ventana de dialogo principal.**

Seguidamente se muestran las ventanas de dialogo secundarias, que se han mencionado en la tabla 4.1, las cuales le proporcionan al usuario la interacción con las funcionalidades de la aplicación. En primer lugar tenemos, la ventana de dialogo de solicitud de conexión:



**Figura 4.3. La ventana de dialogo de conexión.**

Esta ventana de dialogo le solicita al usuario la dirección IP del equipo remoto al que se desea conectar, e inicia el proceso de conexión. Hecho esto, en el equipo remoto se muestra una ventana de dialogo de confirmación de conexión:

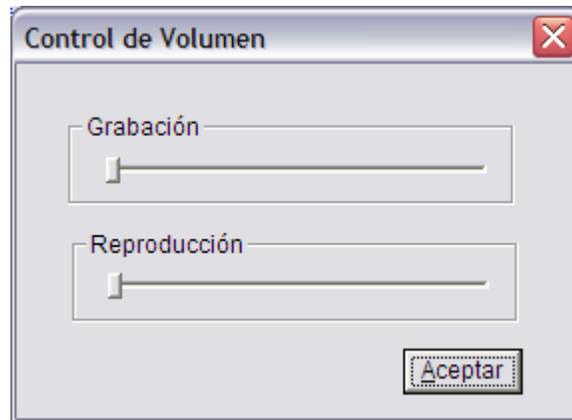


**Figura 4.4. La confirmación de conexión.**

Esta ventana le permite al usuario remoto aceptar o rechazar la conexión, en caso de aceptarla, se establece una conexión virtual entre el equipo local y el equipo remoto, permitiendo transmitir la señal del equipo local al remoto. Se debe realizar una operación similar en el lado remoto, para obtener en el equipo local señal del equipo remoto. Todo este proceso se lo hace de este modo debido a que la señales (de audio y video) que se van a transmitir son bidireccionales, y por lo

tanto se deben establecer dos canales virtuales para completar el enlace, el canal local del equipo llamante se convierte en el canal remoto en el otro equipo y con ello se establece el enlace bidireccional.

Seguidamente se observa la siguiente ventana de diálogo que nos indica la tabla 4.1, esto es la ventana de diálogo que permite calibrar los niveles de audio:



**Figura 4.5. La ventana de diálogo de calibración de los niveles de audio.**

Como se puede observar en la figura 4.5, esta ventana de diálogo permite calibrar los niveles de audio (volumen) tanto de la grabación (captura) de la señal, esto es, el nivel de audio de entrada o del micrófono, como de la reproducción de la señal, esto es el nivel de audio de salida o de los parlantes, esta utilidad es muy práctica debido a que siempre es necesario calibrar estos valores debido a la cercanía de los periféricos entre sí, lo cual como se podría esperar, puede llegar a producir un silbido de retroalimentación, y con ello saturar los circuitos de entrada y salida de audio.

Finalmente se tiene la ventana de diálogo que nos muestra información referente a la aplicación, esto es, la ventana de Acerca de, ventana que muestra la información referente al presente proyecto.



**Figura 4.6. La ventana Acerca De**

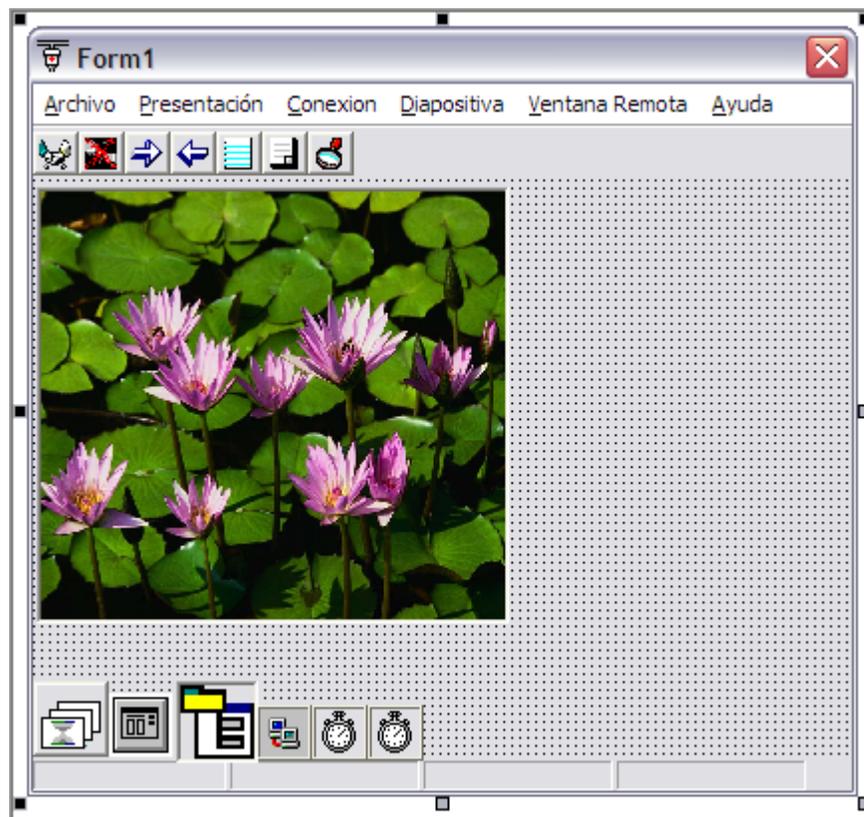
## 4.2 EL SISTEMA DE PRESENTACIÓN REMOTA

### 4.2.1 EL LADO SERVIDOR

En este caso la aplicación como se vio en el capítulo anterior será desarrollada enteramente en Visual Basic 6.0, por lo que el desarrollo de esta aplicación es más amigable y requiere de menos programación.

La aplicación servidor expone toda su funcionalidad en su formulario principal, sin embargo existen dos formularios adicionales, el primero es el formulario que nos muestra la información referente a la aplicación y el segundo en una ventana que nos muestra el progreso de una operación.

En primer lugar se analiza el formulario principal:



**Figura 4.7. El formulario principal**

Como se puede observar en la figura 4.7 el formulario principal consta de un menú, una barra de botones, un control gráfico, un conjunto de botones que nos

proporcionan gran parte de la funcionalidad de la aplicación, y una barra de estado, en la parte inferior de la ventana.

A continuación se detalla las funcionalidades expone el menú de este formulario:

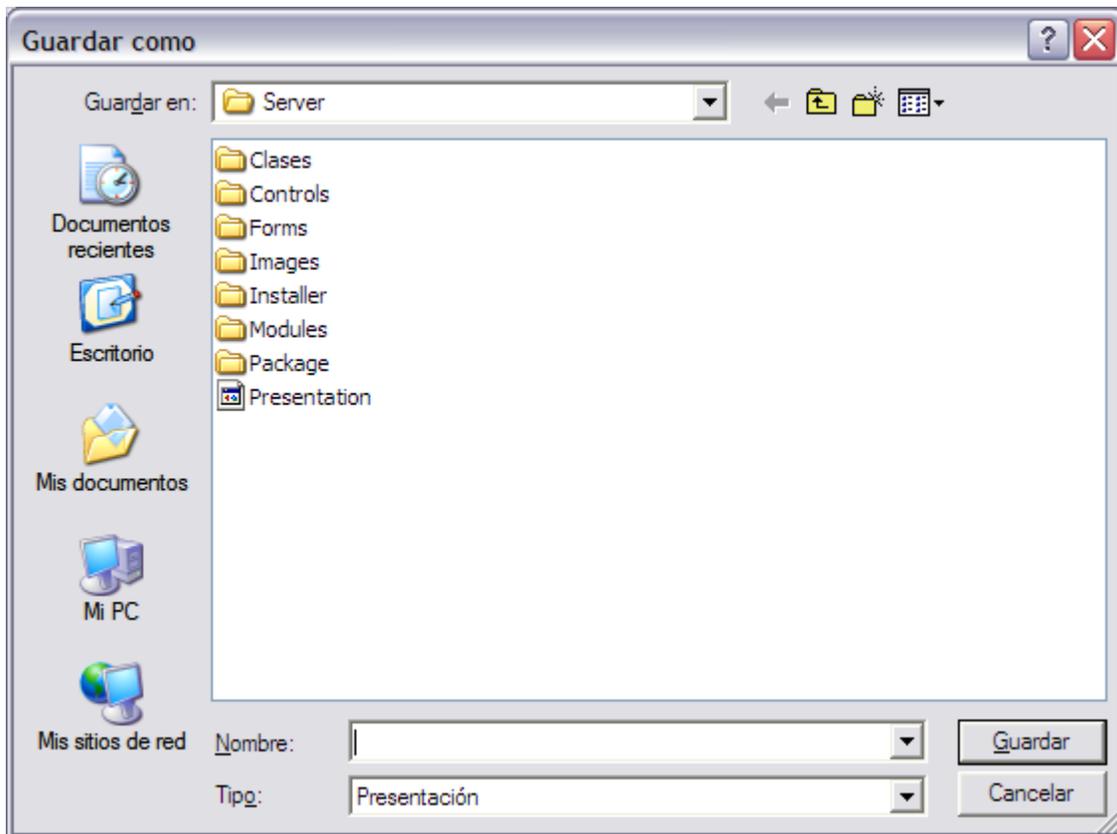
Opción	Funcionalidad	Atajo	Descripción
Archivo	Salir	Ctrl + Q	Cierra la aplicación.
Presentación	Crear Nueva	Ctrl + N	Crea una nueva presentación en blanco
	Abrir Existente	Ctrl + O	Carga en memoria una presentación previamente creada
Conexión	Conectar		Prepara el canal virtual y comienza a esperar una solicitud de conexión
	Desconectar		Cierra la conexión con el equipo remoto
Diapositiva	Avanzar	Ctrl + F	Adelanta una diapositiva
	Retroceder	Ctrl + B	Retrocede una diapositiva
Ventana Remota	Maximizar		Maximiza la ventana en el equipo remoto

Opción	Funcionalidad	Atajo	Descripción
	Minimizar		Minimiza la ventana en el equipo remota
Ayuda	Acerca De		Muestra información referente a la aplicación.

**Tabla 4.2. El menú desplegable del formulario principal.**

La implementación de cada una de las opciones del menú fueron analizadas en el capítulo anterior, por lo que no se profundizará en ello, sin embargo es necesario aclarar como fue implementada cada funcionalidad, ya que ha diferencia de la programación en Visual C++, esta no es implementada en forma de código sino más bien empleando controles Activex (modelo COM de Microsoft).

Seguidamente se analiza la creación de la presentación en blanco, esta opción permite crear una copia renombrada de un archivo existente, para ello solicita al usuario el nombre y la ubicación del archivo a ser creado mediante una ventana de dialogo, proporcionada por uno de los controles que se muestran en la parte inferior del formulario principal, como se muestra en la figura 4.7.



**Figura 4.8. La ventana de dialogo guardar como.**

Como se puede observar en la figura 4.8, solo se pueden guardar, en este caso crear, archivos del tipo “presentación”, que es un formato propietario del sistema. Algo más que se debe observar respecto a este formulario es que es un formulario estándar de Windows, por lo que nuestro control, internamente llama a las librerías de Windows para desplegar esta ventana.

Continuando, se describe en más detalle de que se trata el archivo de presentaciones. Este archivo no es más que una base de datos de Microsoft Access; se eligió Microsoft Access por dos razones, la primera es que Visual Basic, proporciona de forma nativa, control sobre Microsoft Jet, que es el motor de esta base de datos; y la segunda es que Microsoft Jet es el único sistema de base de datos de Microsoft que se puede portar fácilmente, ya que no requiere de un motor externo, como por ejemplo SQL Server.

La estructura de la base de datos es bastante sencilla, ya que su única finalidad es la de almacenar imágenes en forma de cadenas binarias, por lo que consta de una única tabla con dos campos el primero que contiene la clave primaria y es de tipo auto numérico (id), y el segundo que contiene las cadenas binarias, y que es de tipo OLE (Picture).

Una vez creado el archivo en el que se almacenarán las diapositivas, se requiere que se carguen las diapositivas al mismo, esto se lo hace mediante la siguiente ventana de dialogo que se le presenta al usuario:



**Figura 4.9. La ventana de dialogo buscar carpeta.**

Esta ventana de dialogo se despliega empleando otro control de los que se encuentran ubicados en parte inferior del formulario principal y despliega la ventana de árbol de Windows, este control fue previamente creado, y al igual al otro control que si pertenece a Visual Studio, llama a las API de Windows para implementar su funcionalidad. Esta ventana de dialogo ubica únicamente carpetas, ya que la funcionalidad de esta parte de la aplicación es ubicar la carpeta que contiene las diapositivas (imágenes) que van a ser cargadas en el sistema.

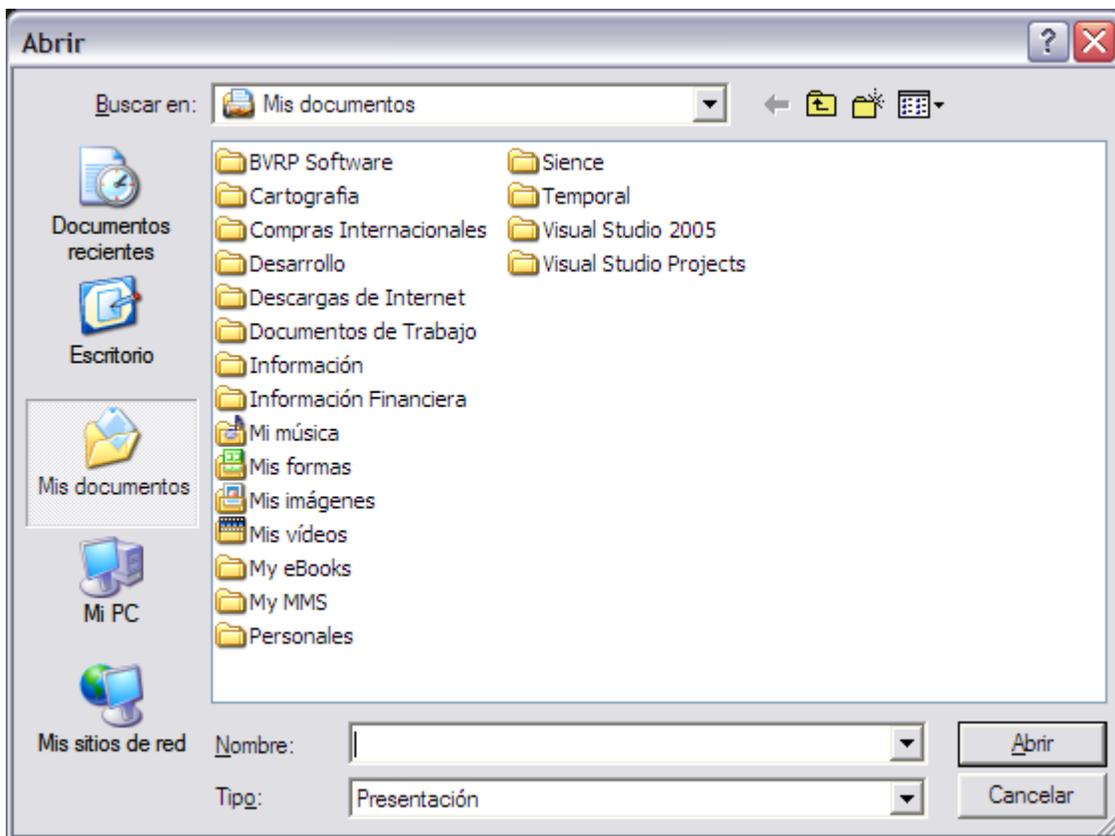
Ubicada la carpeta que contiene las imágenes es dónde entra en juego el siguiente formulario, el cual muestra una barra de progreso, del proceso que se esta llevando a cabo, en este caso, la carga de las diapositivas en el archivo de presentación.



**Figura 4.10. El formulario de progreso.**

Como se puede observar en la figura 4.10, el formulario contiene además de la barra de progreso, un control etiqueta, el cual permite mostrar el archivo que está siendo cargado en un determinado momento.

Continuando con el análisis del menú, la siguiente opción, cargar una presentación existente, no hace otra cosa que abrir un archivo de presentación que fue creado mediante el proceso anteriormente descrito. Esta opción también muestra una ventana de dialogo, que le permite al usuario escoger el archivo de presentación que desea. Como se puede observar en la figura 4.11, la ventana de dialogo es muy similar a la ventana de dialogo de creación del archivo, esta también es una ventana de dialogo estándar de Windows y es desplegada por el mismo control que se emplea para desplegar la ventana de dialogo de la figura 4.8.



**Figura 4.11. La ventana de dialogo abrir una presentación existente.**

La siguiente opción es la de conectar, esta opción configura el canal virtual y espera por una solicitud de conexión. El estado del canal virtual se muestra en la barra de estado, en el la parte inferior de la pantalla.

La opción desconectar, simplemente cierra el canal virtual desconectando cualquier enlace que se haya encontrado establecido.

Las siguientes opciones corresponden a las funcionalidades de las diapositivas, estas simplemente permiten avanzar o retroceder una diapositiva a la vez. Cabe mencionara que estas opciones solo se encontrarán disponibles una vez que se haya cargado en memoria un archivo de presentación.

El siguiente conjunto de opciones corresponde a los comandos remotos de la ventana del lado cliente, esta opciones nos permiten maximizar la ventana cliente

al tamaño máximo del área visual del sistema de video en el cual corre la aplicación cliente, o minimizar la ventana a un tercio del área antes mencionada, ubicando la ventana a dos tercios a la derecha, y uno hacia abajo, permitiendo al área de video de la aplicación video conferencia, quedar totalmente visible.

Finalmente se tiene la opción acerca de, la cual muestra el último formulario, el cual como se puede observar en la figura 4.12, despliega información referente a la aplicación.



**Figura 4.12. La ventana de dialogo acerca de.**

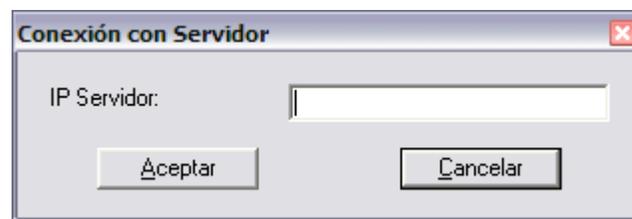
Como punto final cabe mencionar que la barra de botones, simplemente pone de forma más accesible al usuario algunas de las funcionalidades descritas en el menú:

- Conectar
- Desconectar
- Avanzar diapositiva
- Retroceder diapositiva
- Maximizar ventana remota
- Minimizar ventana remota

- Maximizar / minimizar la ventana local, esta opción es similar a las anteriores, referentes a la ventana del lado cliente, que corre en el equipo remoto, pero se aplica a la venta principal del lado servidor. Esta funcionalidad solo se encuentra disponible en este botón.

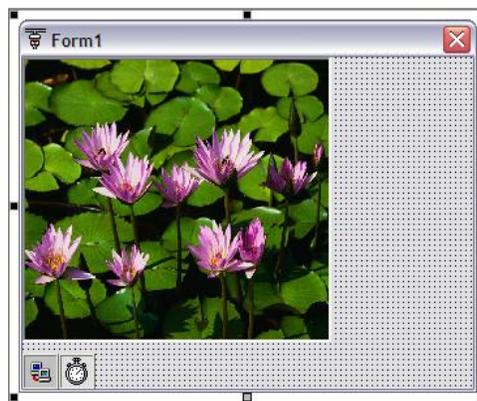
#### 4.2.2 EL LADO CLIENTE

Esta aplicación al igual que su aplicación complementaria ha será desarrollada enteramente en Visual Basic 6.0. Su funcionalidad como se ha venido viendo, es bastante limitada, esta aplicación consta únicamente de dos formularios, el primero un formulario de dialogo que le permite solicitar una conexión con el equipo remoto que corre la aplicación servidor.



**Figura 4.13. La ventana de dialogo de solicitud de conexión.**

Realizada la solicitud, la aplicación muestra su formulario principal, el cual es una ventana para poder visualizar las diapositivas enviadas desde el equipo que corre la aplicación servidor.



**Figura 4.14. La ventana principal de la aplicación cliente.**

Esta ventana como se puede observar en la figura 4.14, no tiene ninguna opción o funcionalidad visible para maximizarla o minimizarla, todas estas funcionalidades se las administra remotamente desde la aplicación servidor, de modo que esta aplicación una vez conectada con la aplicación servidor es administrada en su totalidad de forma remota.

También es posible observar como en su aplicación complementaria, la existencia de, en este caso dos controles Activex, que son los que nos proporcionan las funcionalidades en el primer caso de red, específicamente, de la administración de sockets, y en el segundo caso, nuestro cronómetro para la sincronización de las comunicaciones.

## **CAPITULO 5**

### **5.1 PRUEBAS Y RESULTADOS**

Seguidamente se describe el protocolo de pruebas realizadas a los sistemas, el mismo que contiene las características de los equipos utilizados, así como sus parámetros de operación y los resultados obtenidos de estas pruebas.

A ambos sistemas les sometimos a las siguientes pruebas:

#### **5.1.1 CONEXIÓN ENTRE REDES LAN EMPLEANDO RUTEADORES Y ADMINISTRANDO LA VELOCIDAD DEL CANAL**

En esta prueba se emplearon dos ruteadores CISCO, al puerto ethernet de cada ruteador se le conectó un computador con su respectivo equipo, esto es, la cámara, el micrófono, y el tablero digitalizador, en el caso del equipo servidor, y la cámara y el micrófono en el caso del equipo cliente.

La prueba comenzó con una velocidad de canal de 56 Kbps, para el caso del sistema de video conferencia, se produjo un retardo de alrededor de 4 segundos entre imágenes, y luego el sistema acumuló demasiados errores de transmisión que simplemente dejó de transmitir y las imágenes en cada lado quedaron congeladas, el tiempo de retardo entre los equipos al transmitir la información se lo midió en alrededor de 20 ms. Para el caso del sistema de presentación remota, este ancho de canal fue suficiente para transmitir la información, ya que la cantidad de información que transmite no es tan grande como en el caso del otro sistema, simplemente transmite una imagen, y paquetes de comandos y sincronización, de modo que este ancho de canal fue suficiente.

A continuación se subió la velocidad del canal a 128 Kbps, nuevamente para el caso del sistema de video conferencia se produjeron retrasos muy similares que en el caso de la prueba anterior, con la diferencia que demoró más tiempo que el sistema congelara las imágenes. El otro sistema, evidentemente continuó trabajando correctamente. En este caso el retraso medido fue de alrededor de 15 ms.

Finalmente se subió la velocidad del canal a 256 Kbps, el sistema de video conferencia tampoco trabajo correctamente. En esta ocasión se midió el retraso en un valor cercano a los 10 ms.

### **5.1.2 CONEXIÓN ENTRE EQUIPOS EN UNA RED WAN**

Para esta prueba se emplearon dos equipos de cable modem, de marca Motorola, los cuales proporcionaron el canal de transmisión, en un lado el canal era de 256 Kbps y en el otro de 400 Kbps; y dos computadores debidamente equipados de igual manera que en el caso anterior.

La prueba se la hizo empleando a un computador para administrar remotamente al otro, de modo que el procedimiento de conexión de las aplicaciones se las realizó de forma centralizada en el computador que servía de servidor.

En esta ocasión la prueba tampoco fue muy exitosa para el caso de la video conferencia, se logró realizar la conexión y simplemente se obtuvo una imagen la cual luego quedó congelada. En el caso del sistema de presentación remota como era de esperarse de la prueba anterior, funcionó correctamente. En este caso el retardo medido fue de alrededor de 30 ms.

### **5.1.3 CONEXIÓN EN UNA RED LAN A TRAVÉS DE EQUIPOS INALÁMBRICOS**

Para esta prueba se emplearon dos condiciones diferentes, el primer caso fue empleado una red bajo el protocolo 802.11g, en una red poco ruidosa, pero congestionada de tráfico. La congestión se simuló al transmitir una gran cantidad de información de un equipo al otro. En el segundo caso se probó empleando el protocolo 802.11b, en una red ruidosa y congestionada.

Para la primera prueba se emplearon un Switch D-Link 10/100, un Access Point D-Link, y una tarjeta PCMCIA Wireless, de la misma marca; además de dos computadores debidamente equipados. En este caso el ancho de banda fue de ente 11 y 54 Mbps, dependiendo de la ubicación del equipo portátil, con un retardo que oscilaba en alrededor de 3 ms. En esta prueba ambas aplicaciones trabajaron correctamente, tanto la aplicación de video conferencia como la aplicación de

presentación remota. En el caso de la presentación de video conferencia el retardo entre imágenes (Tx – Rx), fue de alrededor de 1 s.

Para la segunda prueba se emplearon equipos inalámbricos CISCO. El ruido en la red fue inducido por interferencia entre otras redes inalámbricas muy cercanas trabajando bajo el mismo protocolo, aunque en canales diferentes, el trafico fue el generado por alrededor de 32 computadores. El ancho de banda en este caso fue de 12 Mbps, con un retardo de alrededor de 4 ms.

En esta ocasión el sistema de video conferencia funcionó apropiadamente, mientras que el sistema de presentación remota tuvo problemas para transmitir, debido principalmente a su trabajo síncrono y su pobre control de recepción de paquetes. Este último problema fue superado realizando modificaciones al sistema, de modo que el sistema final ya no es tan susceptible a desconectarse debido a la pérdida de paquetes.

## **5.1.4 CONCLUSIONES Y RECOMENDACIONES**

### **5.1.4.1 Conclusiones**

- En primer lugar, debemos concluir que el sistema funciona correctamente en el ámbito para el cual fue diseñado, es decir en redes LAN, de cualquier tipo, en las cuales tenemos al menos 1 Mbps de ancho de banda.
- El sistema de video conferencia funciona adecuadamente con retardos inferiores a los 4 ms.
- El sistema de presentación remota funciona correctamente con retardos inferiores a 20 ms.
- Ni el sistema de video conferencia ni el de presentación remota son sensibles a problemas de tráfico.
- Ni el sistema de video conferencia ni el de presentación remota son sensibles al ruido del canal.

#### **5.1.4.2 Recomendaciones**

- El sistema se debe emplear únicamente en redes LAN, con un ancho de banda de al menos 1 Mbps.
- Debido a la cantidad de tráfico y procesamiento que genera la aplicación de video conferencia, es recomendable que los equipos que se empleen para implementar el sistema sean de buena calidad, esto es equipos con procesadores Intel pentium 4, movile, o superior, a velocidades de 2 GHz o superiores, con al menos 256 Mb de memoria RAM.
- Algo similar se aplica a las tarjetas de red, tienen que ser de buena calidad, en este caso, es preferible que sean de marcas reconocidas.

#### **5.1.5 LO QUE SE PUEDE MEJORAR DEL SISTEMA**

De lo estudiado hasta el momento, de la presente Tesis de Grado, las metas propuestas se han cumplido, sin embargo queda mucho trabajo por hacer para obtener un sistema de video conferencia que rivalice con aquellos propuestos a nivel comercial, a continuación se lista las mejoras propuestas:

- Mejorar la estimación de movimiento al cálculo de superficies de correlación.
- Implementar el algoritmo de superficies de correlación.
- Permitir al usuario escoger el formato CIF de transmisión recepción de cuadros (dimensiones del cuadro).
- Implementar un algoritmo de amplificación digital de imagen para poder incrementar el tamaño local del cuadro sin afectar el ancho de banda.
- Integrar ambos sistemas en una sola aplicación.
- Permitir conexiones múltiples, ya no solo punto a punto.
- Mejorar la interfaz con el usuario, llevando todo el interfaz gráfico a ventanas, en el caso del sistema de video conferencia.

## REFERENCIAS BIBLIOGRÁFICAS

- [1] MSDN, Microsoft Corporation, 2003
- [2] WikiPedia, [www.wikipedia.com](http://www.wikipedia.com)
- [3] Redes de Ordenadores, Segunda Edición, Tanenbaum Andrew S.,  
Prentice-Hall Hispanoamericana, S.A., 1991
- [4] Electronic Engineers Handbook, Fourth Edition, Donald Christiansen,  
McGraw-Hill, 1996
- [5] Visual Basic 6.0 Manual del Programador, Microsoft Corporation, Microsoft  
Press, 1998.
- [6] Progrese con Visual C++, Kruglinsky David J., McGraw-Hill, 1994
- [7] Como Programar en C++, Deitel H.M.; Deitel P.J., Prentice Hall, 1994
- [8] C Premier Plus, Prata Stephen, Sams Publishing, 2005
- [9] Visual C++ Unleashed, Williams Myckey, Sams Publishing, 2000
- [10] Enciclopedia de la Electrónica Ingeniería y Técnica, McHugh James  
A. M., Vol. 7, Oceano-Centrum.
- [11] AVI Overview, McGowan John, [www.jmcgowan.com](http://www.jmcgowan.com)
- [12] AVI Video Windows Files (article 205087), [www.microsoft.com](http://www.microsoft.com)
- [13] Video Codec and Pixel Information, FourCC, [www.fourcc.org](http://www.fourcc.org)
- [14] An Embedded Dct-Based Still Image Coding Algorithm, David Nister;  
Charilaos Christopoulos, Ericsson Telecom AB.
- [15] The Engineer's Guide to Motion Compensation, John Watkinson,  
Snell & Wilcox.
- [16] International Telecommunications Union, [www.itu.int](http://www.itu.int)
- [17] RFC Editor, [www.rfc-editor.org/rfc.html](http://www.rfc-editor.org/rfc.html)
- [18] Inverse Two Dimensional DCT, Chen-Wang algorithm, IEEE ASSP-  
32, pp. 803-816, Aug. 1984

## **ANEXOS**

- [1] Código del sistema de videoconferencia
- [2] Código del sistema de presentación remota.
- [3] Instaladores del sistema
- [4] Descripción de archivos del sistema de video conferencia

Debido a la extensión de esta información, se suministra adjunta en un CD.

## Descripción de archivos del sistema de video conferencia

Archivo	Num . Lineas
root\ConnectDlg.cpp	79
root\ConnectDlg.h	134
root\DSocket.cpp	436
root\DSocket.h	515
root\Mixer.cpp	676
root\Mixer.h	718
root\PlaySound.cpp	953
root\PlaySound.h	1,022
root\RecordSound.cpp	1,269
root\RecordSound.h	1,338
root\RequestDlg.cpp	1,426
root\RequestDlg.h	1,485
root\resource.h	1,537
root\StdAfx.cpp	1,553
root\StdAfx.h	1,587
root\VideoCapture.cpp	1,823
root\VideoCapture.h	1,889
root\VideoConf.cpp	1,949
root\VideoConf.h	1,999
root\VideoConfDlg.cpp	2,771
root\VideoConfDlg.h	2,898
root\Volume.cpp	2,970
root\Volume.h	3,008
root\encoder\coder.cpp	37
root\encoder\coder.h	67
root\encoder\config.h	187
root\encoder\countbit.cpp	191
root\encoder\countbit.h	221
root\encoder\ctables.h	471
root\encoder\dct.cpp	478
root\encoder\dct.h	492
root\encoder\def.h	658
root\encoder\Global.cpp	685
root\encoder\Global.h	738
root\encoder\huffman.cpp	873
root\encoder\huffman.h	911
root\encoder\libr263.cpp	1,152
root\encoder\libr263.h	1,183
root\encoder\mot_est.cpp	1,189
root\encoder\mot_est.h	1,214
root\encoder\owntypes.h	1,233
root\encoder\pred.cpp	1,240
root\encoder\pred.h	1,271
root\encoder\quant.cpp	1,272
root\encoder\quant.h	1,283
root\encoder\rlib.cpp	1,397

Archivo	Num . Lineas
root\encoder\rlib.h	1,417
root\encoder\stream.cpp	1,419
root\encoder\stream.h	1,432
root\encoder\struct.h	1,627
root\decoder\convert.cpp	248
root\decoder\convert.h	266
root\decoder\decdef.h	329
root\decoder\decstruct.h	374
root\decoder\DGlobal.cpp	579
root\decoder\DGlobal.h	676
root\decoder\GetBits.cpp	838
root\decoder\GetBits.h	863
root\decoder\GetBlk.cpp	1,135
root\decoder\GetBlk.h	1,161
root\decoder\GetHdr.cpp	1,290
root\decoder\GetHdr.h	1,311
root\decoder\GetPic.cpp	2,348
root\decoder\GetPic.h	2,384
root\decoder\GetVlc.cpp	2,702
root\decoder\GetVlc.h	2,737
root\decoder\ldct.cpp	2,898
root\decoder\ldct.h	2,928
root\decoder\ldctref.cpp	3,005
root\decoder\ldctref.h	3,021
root\decoder\Indices.cpp	3,060
root\decoder\Indices.h	3,088
root\decoder\main.cpp	3,224
root\decoder\Recon.cpp	3,978
root\decoder\Recon.h	4,005
root\decoder\Sac.cpp	4,161
root\decoder\Sac.h	4,182
root\decoder\Sactbls.cpp	4,241
root\decoder\Sactbls.h	4,280
root\decoder\Tmndec.cpp	4,541
root\decoder\Tmndec.h	4,568
root\decoder\YuvToRgb.cpp	4,715
<b>TOTAL</b>	<b>137,509</b>