Transmisión Multicast en Tiempo Real de Vídeo MPEG-4

J. M. Dana, V. G. Ruiz, M. F. López, S. G. Rodríguez, J. P. Ortiz y I. García
Departamento de Arquitectura de Computadores y Electrónica
Universidad de Almería

Resumen

En este trabajo se presenta una solución al problema de la transmisión de vídeo en tiempo real en redes no confiables, como es el caso de Internet.

Para ello hemos utilizado comunicaciones multicast, así como una modificación (realizada por nosotros mismos) de la implementación de dominio público de MPEG-4 realizada por `FFMPEG`[5].

Los resultados visuales obtenidos y las pruebas realizadas demuestran que nuestro sistema es capaz de proporcionar una solución válida al problema de la transmisión de vídeo en tiempo real en redes comunicación con un ancho de banda limitado.

1. Motivación

Desde la creación de las llamadas redes de difusión y la tremenda popularidad de la que hoy goza Internet nos encontramos con una serie de necesidades, anteriormente inexistentes, que ahora se deben cubrir (información, comunicación, entretenimiento, etc.) Aquí es donde podemos encontrar la unión entre las redes de comunicación y los servicios multimedia; emisión de radio por Internet, retransmisión de conciertos en directo, charlas virtuales, etc. Todos estos servicios tienen una mayor demanda cada día y es misión de los sistemas informáticos hacerlos posibles.

El rendimiento de los procesadores crece de forma exponencial desde hace varias décadas, lo que permite realizar operaciones cada vez más complejas. Desgraciadamente, las redes de comunicación no crecen al mismo ritmo, de ahí que las técnicas de compresión hayan avanzado tanto en los últimos años y se hayan convertido en una herramienta de uso diario; cuando la cantidad de datos a transmitir crece mucho más rápido que el ancho de banda de nuestro enlace nos vemos obligados a utilizar técnicas alternativas que nos permitan realizar la transmisión en el menor tiempo posible.

Los sistemas de transmisión de vídeo actuales suelen tener limitaciones importantes en cuanto al ancho de banda se refiere. Los equipos que sirven las secuencias (en adelante, servidores) deben tener una gran velocidad de transmisión para hacer llegar la información a los clientes que se conectan al mismo. De esta forma, el número de clientes capaces de acomodar se encuentra íntimamente ligado al ancho de banda disponible.

Esta limitación es muy importante cuando hablamos de transmisiones de eventos multitudinarios (conciertos, deportivos, etc.) o de sistemas en tiempo real, donde la información no sólo debe llegar, sino que debe hacerlo en el menor tiempo posible.

Por todo lo anterior, hemos diseñado un sistema de transmisión que, utilizando comunicaciones multicast, es capaz de acomodar a una cantidad de clientes ilimitada, permitiendo emitir con un ancho de banda limitado por parte del servidor.

2. Descripción de una transmisión multicast

Tal y como dijimos anteriormente y como podemos apreciar en la Figura 1, en el caso de una transmisión multicast realizada mediantes conexiones unicast, los tramos del enlace cer-

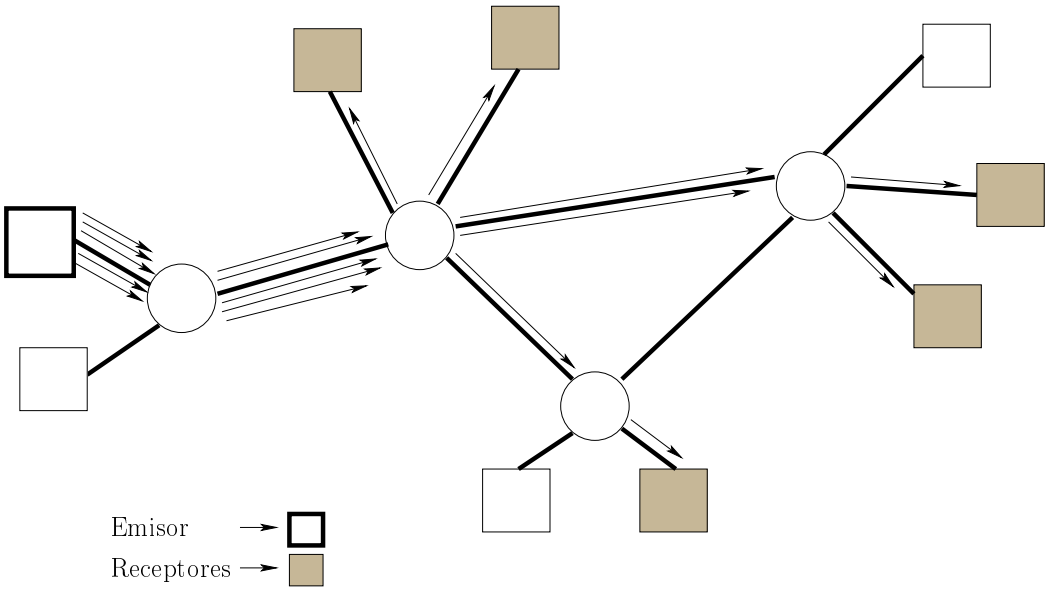

Figura 1: Transmisión unicast en una red conmutada.

canos al servidor se ven saturados por los distintos flujos de datos que van hasta los clientes. Si, además, existe una realimentación que permite a los clientes enviar información al servidor, el enlace se ve doblemente saturado.

Sin embargo, para una transmisión multicast (ver Figura 2), el enlace no se ve afectado por el número de clientes, ya que existe un único flujo de datos para todos.

Por supuesto, existen varios problemas en una transmisión multicast[2], todos debidos a la naturaleza de la misma. Como primer problema, encontramos que una dirección multicast nunca puede ser una dirección de destino, es decir, los clientes no podrán transmitir información al servidor por medio del canal. Por supuesto, este problema no es significativo, dado que nuestra finalidad es la de conseguir un sistema capaz de acomodar a cuantos clientes sean necesarios, y si todos compartiesen información con el servidor, nos encontraríamos en la misma situación descrita en la Figura 1.

Por otra parte, una comunicación multicast precisa de redes IPv6 o, en su defecto, de routers multicast capaces de hacer encapsulado IP-in-IP (ver Figura 3), para poder llevar los paquetes de emisor a receptor sin que routers intermedios los desechen.

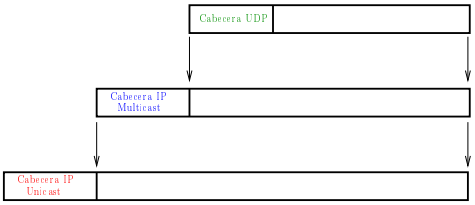

Figura 3: Encapsulamiento IP-in-IP.

3. Descripción del estándar MPEG-4

El estándar MPEG-4[1], como sus antecesores, utiliza un sistema de codificación que relaciona los distintos *Video Plane Objects* (en adelante VOP) entre sí según el esquema de la Figura 4. De esta forma, los I-VOPs (o *intrafra-*

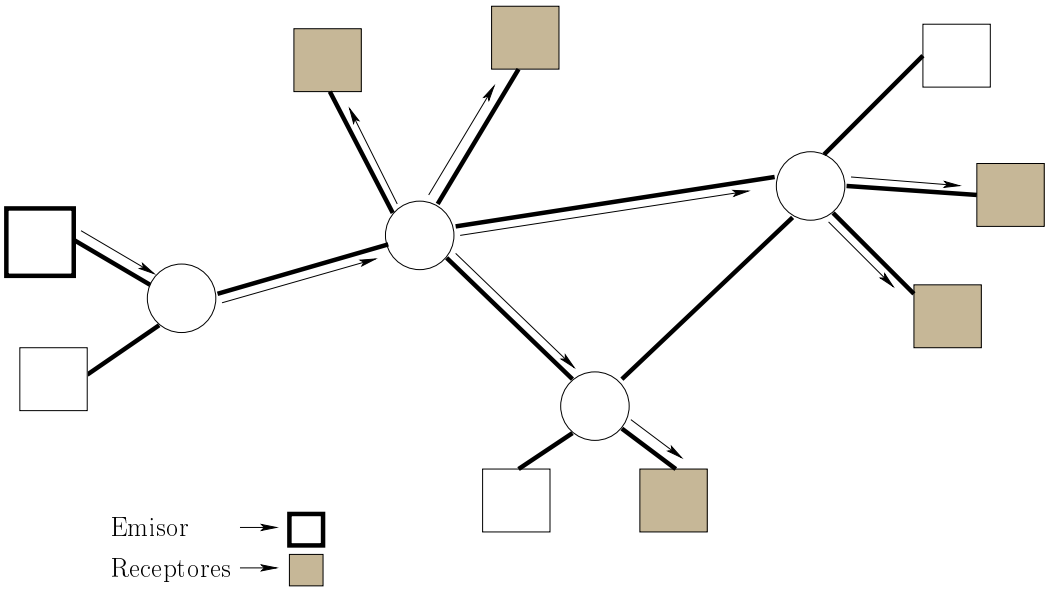

Figura 2: Transmisión multicast en una red conmutada.

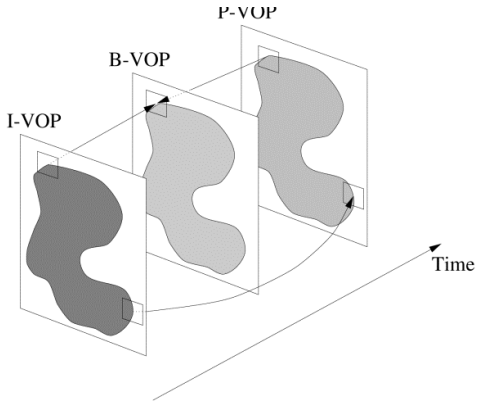

Figura 4: Relación entre VOPs de una secuencia MPEG-4.

mes) son codificados independientemente, los P-VOPs dependen del I-VOP anterior y los B-VOPs dependen del I-VOP o P-VOP anterior y posterior (de ahí que tanto a los P-VOPs como a los B-VOPs se les conozca también como *interframes*). A partir de lo anterior, podemos definir el *Group of VOPs* (en adelante, GOV), que es el grupo de VOPs inter-relacionados, es decir, un grupo donde sólo encontramos un I-VOP del que, directa o indirectamente, dependen todos los demás.

Por tanto, cuanto mayor sea el GOV, mayor será la tasa de compresión y menor la resistencia a errores del sistema, ya que estos se propagaran por los VOPs debido a la dependencia entre ellos, hasta encontrar el siguiente intraframe. Esto último, es de especial importancia en nuestro trabajo. Un GOV excesivamente largo podría deteriorar la imagen significativamente durante un periodo de tiempo excesivo, mientras que uno demasiado corto requeriría de un ancho de banda extra para su transmisión.

4. Descripción de nuestro esquema de transmisión

Teniendo en cuenta las descripciones anteriormente presentadas, hemos desarrollado un esquema de transmisión donde:

- El servidor emite utilizando un canal multicast por secuencia de vídeo.
- Los clientes realizan la recogida de datos, descodifican las secuencias de vídeo y las muestran por pantalla.
- El servidor no recibe información de los clientes, es decir, no tiene información sobre el ancho de banda disponible o el número de los mismos.
- El cliente no recibe información por parte del servidor, salvo la referente estrictamente al vídeo, de forma que desconoce la situación del mismo en cuanto a recursos disponibles.

Por otra parte, el cliente contará con tres procesos ejecutándose concurrentemente (recepción de datos, descodificación y visualización) que nunca deberán sincronizarse mediante semáforos o mutex, dada la necesidad de un sistema en tiempo real. Por tanto, para mantener la sincronía entre los procesos y controlar el acceso a elementos comunes (buffers circulares para la recogida de datos y la descodificación) se han utilizado interrupciones del sistema (que dada su naturaleza no suponen una latencia extra para el sistema al completo) y punteros a memoria comunes.

4.1. Recepción de datos

Dadas las premisas anteriores, cada cliente debe controlar el buffer de datos de forma independiente, utilizando sistemas de control internos y sin realizar especulaciones sobre la velocidad de transmisión o la capacidad computacional del sistema (dado que, por otra parte, carece de esa información).

Además, el buffer de recogida de datos es dinámico, es decir, su tamaño se modifica en tiempo real según la situación del cliente (ancho de banda disponible, capacidad computacional, etc.).

4.2. Descodificación de datos

En cuanto al proceso de descodificación, se han realizado modificaciones sobre la biblioteca `FFMPEG`, sobre todo en lo referente a las etapa de cuantificación y descuantificación, donde la biblioteca retornaba valores inválidos cuando la cantidad de datos perdida era excesiva y no podía reconstruir convenientemente el flujo de datos. Podríamos decir que los cambios realizados sobre la biblioteca persiguen proporcionar mayor resistencia a errores al estándar.

Estamos, por supuesto, ante el proceso con mayor carga de trabajo del sistema al completo. La descodificación de vídeo MPEG-4 no es excesivamente costosa, pero sí significativa para equipos de baja potencia computacional, donde éste será el cuello de botella del sistema.

4.3. Visualización

Para la visualización de los datos descodificados se han utilizado bibliotecas que son capaces de aprovechar las características del microprocesador gráfico (aceleración 3D, instrucciones especiales, etc.), con lo que el rendimiento es óptimo y en ningún caso supone un cuello de botella o implica una disminución del rendimiento de las etapas anteriormente citadas. Además, los fotogramas se van presentando utilizando interrupciones del sistema, es decir, no se utilizan temporizadores que, normalmente, producen una latencia en el sistema completo.

Las bibliotecas elegidas han sido las del proyecto `XFree86`[6] y las `SDL`[7].

5. Evaluación

Para evaluar el rendimiento del sistema, se han utilizado distintas configuraciones de sistemas actuando como clientes, tal y como vemos en la Tabla 1.

Como vídeos de ejemplo se han utilizado secuencias clásicas dentro del panorama de la codificación y transmisión de vídeo (Akiyo, Flowergarden, Tempete, etc.), así como todo tipo de secuencias menos conocidas (películas, eventos deportivos, etc.). De esta forma, hemos obteniendo un grupo totalmente hetero-

Figura 5: Fotograma de la secuencia Foreman afectado por un fallo en la transmisión.

Figura 6: Fotograma de la secuencia Flowergarden afectado por un fallo en la transmisión .

géneo de vídeos, a fin de demostrar la funcionalidad del sistema.

Ya que nuestro sistema explota el conjunto de instrucciones especiales de los distintos procesadores, así como la aceleración 3D de los microprocesadores gráficos, hemos obtenido resultados satisfactorios para las 3 configuraciones, siendo quizás la última algo limitada al carecer de aceleración 3D en el microprocesador gráfico (el proceso de visualización actuaba como cuello de botella del sistema al completo, en lugar de la descodificación, como era de esperar).

El sistema responde correctamente ante posibles fallos de transmisión, llegando a detener la visualización en caso de caída del enlace, por ejemplo, y continuando el proceso en el momento adecuado cuando la situación se estabiliza. Además, gracias a las modificaciones del descodificador de MPEG-4, la resistencia a errores es mayor y los paquetes UDP perdidos/desordenados no alteran significativamente la calidad final de la reproducción.

En las Figuras 5 y 6 se pueden ver dos fotogramas afectados por errores en la transmisión. Se ha simulado una caída total del enlace y, como podemos ver, sólo se ha visto afectada una parte de los fotogramas (correspondiente a los paquetes perdidos). Estos es debido a la propia naturaleza del estándar MPEG-4[1] y a ciertas mejoras realizadas por nosotros.

Cuadro 1: Algunos de los equipos utilizados para evaluar el rendimiento.

	Frec. de reloj	Mem. RAM	Mem. de la GPU	C.I.E.
Centrino	1.6 GHz	512 MB	32 MB DDR	MMX y SSE2
Pent. 4	1.7 GHz	256 MB	64 MB DDR	MMX y SSE2
Pent. III	450 MHz	128 MB	32 MB	MMX y SSE
Cel. PIII	800 MHz	128 MB	32 MB (sin acel. 3D)	MMX y SSE

6. Conclusión

En este trabajo se ha pretendido dar una solución efectiva a la transmisión en tiempo real de vídeo, obteniendo un sistema que consuma la menor cantidad de recursos tanto por parte del cliente como por la del servidor.

Tal y como hemos destacado en la evaluación, consideramos que los objetivos iniciales han sido cumplidos y que el resultado obtenido es correcto. Hemos realizado el diseño e implementación de un sistema de transmisión de vídeo en tiempo real, con unos requisitos computacionales escasos y utilizando para ello un estándar de vídeo actual como es MPEG-4.

Referencias

[1] Fernando Pereira, and Touradj Ebrahimi, *The MPEG-4 Book*, Prentice Hall PTR, 2002.

[2] Douglas E. Comer, *Internetworking with TCP/IP. Principles, Protocols,and Architectures*, Prentice Hall, 2000.

[3] Steven Gringeri, Roman Egorov, Khaled Shuaib, Arianne Lewis, and Bert Basch, *Robust Compression and Transmission of MPEG-4 Video*, Technical report, GTE Laboratories Incorporated, 2003.

[4] Touradj Ebrahimi and Caspar Horne, *MPEG-4 Natural Video Coding - An overview*, Technical report, Swiss Federal Institute of Technology, 2003.

[5] FFMPEG Multimedia System, *http://ffmpeg.sourceforge.net.*

[6] XFree86, *http://xfree86.org.*

[7] Simple DirectMedia Layer (SDL), *http://www.libsdl.org.*