RAE

1. TIPO DE DOCUMENTO: Trabajo de grado para optar por el título de INGENIERO ELECTRÓNICO

2. TÍTULO: EVALUACIÓN DEL PROTOCOLO 802.11P PARA LA TRANSMISIÓN DE VIDEO MULTIVISTA

3. AUTOR (ES): Juan Sebastián Díaz Chapetón

4. LUGAR: Bogotá, D.C

5. FECHA: Enero de 2018

6. PALABRAS CLAVES: Ancho de Banda, Codificación, Compiladores, Evalvid, Inalámbrico, Paquetes Perdidos, Pre-Procesamiento, Protocolo, Redes, Retardo, Simulación, Transmisión, Video Multivista.

7. DESCRIPCIÓN DEL TRABAJO: El objetivo principal de este proyecto es la simulación de la transmisión de un video multivista MVC, haciendo uso del protocolo 802.11p en el software NS2 (Network Simulator 2), con el fin de identificar y evaluar los parámetros de codificación multivista y analizar la efectividad del protocolo en simulaciones controladas por software.

8. LÍNEA DE INVESTIGACIÓN: Línea de Investigación de la USB: Tecnologías de Información y Comunicaciones (TIC) Campo Temático del Programa: Aplicaciones y servicios TIC

9. METODOLOGÍA: El presente proyecto sigue una metodología propia de una investigación aplicada con enfoque experimental y cuantitativo. Es experimental ya que se realizaron experimentos o simulaciones para evaluar la transmisión de flujos de video. También se abordó un análisis cuantitativo relacionado con los parámetros de calidad de la imagen. En el desarrollo del proyecto se utilizaron herramientas de simulación de redes de comunicaciones, así como software para la codificación de video Multivista y análisis de datos.

10.CONCLUSIONES: Después de la integración de Evalvid y NS2 y de evaluar la transmisión de video MVC sobre una red 802.11p. Se comparó el retardo y el flujo de datos sufrido por el tráfico de video. En cuanto al retardo se puede decir que la simulación indica que este está bajo unos límites admisibles para un tráfico de video. En lo referente al flujo de datos se observó la variación de este a medida que los nodos se mueven por el escenario, sin embargo, sabiendo que los escenarios simulados y los resultados fueron limitados, lo importante de este proyecto fue el lograr la integración de los conceptos de 802.11p y de la transmisión de video MVC. De tal manera que se aportaron las bases para desarrollar en el futuro proyectos más avanzados.

UNIVERSIDAD DE SAN BUENAVENTURA SEDE BOGOTÁ

EVALUACIÓN DEL PROTOCOLO 802.11P PARA LA TRANSMISIÓN DE VIDEO MULTIVISTA

JUAN SEBASTIAN DÍAZ CHAPETÓN

INGENIERÍA ELECTRÓNICA

2017

FACULTAD DE INGENIERÍA

UNIVERSIDAD DE
SAN BUENAVENTURA
SEDE BOGOTÁ

EVALUACIÓN DEL PROTOCOLO 802.11P PARA LA TRANSMISIÓN DE VIDEO MULTIVISTA

INGENIERÍA ELECTRÓNICA

PRESENTADO POR

JUAN SEBASTIAN DÍAZ CHAPETÓN

DIRECTOR

Ing. WILDER EDUARDO CASTELLANOS

DEDICATORIA

A la memoria de mis abuelas María Cecilia Uribe y Olga Hilda Guarín, gracias por regalarme sus alegrías, sus sonrisas, por el gran esfuerzo con el que de niño me enseñaron a no rendirme jamás, y porque siempre estaré agradecido con ustedes por su gran amor.

A mis padres, que son mis mejores amigos y que sin importar las circunstancias me han apoyado en los aspectos más importantes de mi vida.

A mi abuelo José Antonio Díaz, a quien le tengo gran admiración, respeto y le doy gracias por ser mi apoyo incondicional.

A mi tía Olga Cecilia Díaz, por darme el mejor ejemplo en el ámbito profesional y apoyarme en mi proyecto de vida.

A mis hermanos que son mi motor de vida y es por quienes lucho a diario buscando siempre darles el mejor ejemplo.

A mis amigos Cristian Rodríguez, Cristian Reyes, Nicolás Prieto y Diana Vargas por compartir conmigo cada una de mis metas y ser un ejemplo de lealtad y fraternidad.

A Angee Hueso quien me acompaño de principio a fin en cada una de las etapas más importantes de este maravilloso proceso de formación profesional.

RESUMEN

En este trabajo se presenta la simulación de la transmisión de un video multivista MVC, haciendo uso del protocolo 802.11p en el software NS2 (Network Simulator 2), con el fin de identificar y evaluar los parámetros de codificación multivista y analizar la efectividad del protocolo en simulaciones controladas por software.

Para el desarrollo de este proyecto se decidió utilizar la plataforma de software libre NS2, por sus siglas en inglés “Network Simulator 2”, JMCV/H264Extension, MP4Box y Evalvid.

Inicialmente se ha dividido el proceso en cuatro etapas establecidas como necesarias para completar el objetivo del proyecto. Primero se realizaron las adaptaciones necesarias al software de simulación de redes NS2, configurando los parámetros del protocolo e instalando los compiladores necesarios para el funcionamiento de la transmisión de video multivista con Evalvid. La segunda etapa fue la codificación de un video multivista para obtener el mismo en formato H.264. La tercera fase, se realizó un pre-procesamiento, configurando los parámetros del protocolo 802.11p en un entorno de red inalámbrico simulado, con el cual se evalúa la transmisión del video MVC.

Finalmente se hace una introducción a transmisión de video sobre redes inalámbricas 802.11p, y se muestran los resultados obtenidos al realizar la simulación y se evalúan las diferentes trazas a partir de las gráficas generadas durante el procedimiento, de paquetes perdidos, throughput y retardo, también se documenta el paso a paso del procedimiento llevado a cabo para lograr cumplir con cada uno de los hitos anterior mente mencionados, de forma que sea posible contribuir al desarrollo de nuevas investigaciones.

CONTENIDO

TABLA DE FIGURAS

TABLAS

INTRODUCCIÓN

Aunque la visualización de imágenes fijas en formato 3D es un tema que tuvo su auge principal en el siglo XX, el despliegue de sistemas de video 3D y la televisión han tomado fuerza recientemente, debido a los avances en visión estereoscópica que han sido de gran impacto en los últimos 5 años. En paralelo a esto, las múltiples generaciones de cámaras 3D y las técnicas implementadas a partir de estos novedosos elementos tecnológicos crean un entorno de competencia y persecución por la búsqueda cada vez más de métodos que permitan una sensación mucho más real, pero mitigando cada vez más la exigencia de elementos externos para poder disfrutar de este espectáculo.

El ser humano percibe el mundo en 3D, debido disparidad binocular de su sistema visual en el que dos imágenes, de la misma escena, se proyectan en las dos retinas. Esto produce la sensación de profundidad que permite que se perciba el mundo en tres dimensiones [1].

El mercado ya comercializa monitores 3D que reciben la forma clásica de sistema estéreo de dos vistas, donde es necesario el uso de gafas. Los monitores 3D auto-estereoscópicos tienen para el usuario, la ventaja de no necesitar gafas especiales para su visualización.

La transmisión de video multivista 3D/MVC hace referencia a la incesante e interactiva entrega continua de contenido multimedia 3D en múltiples vistas, a través de redes que interactúan sin una descarga previa. La importancia del desarrollo y la evolución de tecnologías de transmisión adaptable, puede explicarse como la gran contribución del internet en los últimos tiempos. Concentrar diferentes conceptos de tecnologías 3D o de vista múltiple en estrategias y métodos de transmisión se está convirtiendo en tendencia debido a la exigencia de la demanda actual, los nuevos mercados y directrices tecnológicas.

Este proyecto está centrado en la evaluación del protocolo de comunicación inalámbrico 802.11p, adaptado con el fin de analizar y comparar los diferentes resultados de la transmisión de un video multivista, aplicando codificación, simulación y análisis, para corroborar la veracidad de cada uno de los procedimientos puestos a prueba y concretados con éxito durante la ejecución total del proyecto.

El presente documento está organizado de la siguiente manera: en el capítulo 1 se describen los fundamentos, antecedentes y avances que se han adelantado a partir de la transmisión de video, video 3D y codificación de video multivista (MVC). Se da introducción a cada una de las temáticas con las cuales será posible comprender cada uno los ítems del proyecto y los resultados obtenidos, seguido de esto, en el capítulo dos se hace hincapié a la teoría fundamental, generalidades, aplicaciones y parámetros de configuración del protocolo 802.11p, donde también se documenta el comparativo entre la versión 802.11ª y 802.11p. Dada la explicación de los ítems anteriores, el capítulo 3 describe paso a paso la instalación del protocolo 802.11p, las configuraciones necesarias para que el software sea compatible con el simulador Network Simulator 2.33 (NS2), los errores corregidos durante la instalación y muestra un ejemplo de la codificación de un video prueba con el cual se fundamenta el uso de cada software utilizado. Para finalizar, entrando en materia practica del desarrollo del proyecto, en el capítulo cuatro, describe los fundamentos básicos de transmisión de video, el paso a paso del proceso de codificación, simulación y análisis de resultados del proyecto, y las

diferentes configuraciones necesarias para realizar la simulación final utilizando el protocolo 802.11p.

OBJETIVOS

OBJETIVO GENERAL

Evaluar el protocolo 802.11p mediante software de simulación para determinar las condiciones para la transmisión de video MVC

OBJETIVOS ESPECÍFICOS

- Identificar los parámetros básicos de la codificación de multivista (MVC) así como el funcionamiento del protocolo 802.11p

- Evaluar el funcionamiento del protocolo 802.11p mediante simulaciones en el software NS2.

- Simular la transmisión de video sobre el protocolo 802.11p con el fin de evaluar sus prestaciones durante la transmisión de este tipo de tráfico

CAPÍTULO 1

1 CODIFICACIÓN DE VIDEO MULTIVISTA (MVC)

En este capítulo se resumen y describen los conceptos de la codificación de video multivista (MVC) con el fin de mostrar las principales características y las aplicaciones de dicha codificación. Este marco conceptual servirá para entender los procesos descritos en los capítulos posteriores.

1.1 FUNDAMENTOS DE CODIFICACIÓN DE VIDEO

Para familiarizar los términos, se debe enfatizar en que una secuencia de vídeo es un conjunto de bytes que representa una escena, normalmente en movimiento. La forma más intuitiva para esta representación es acercar la escena en movimiento por una secuencia de imágenes estáticas, llamadas fotogramas (en inglés, frames). El número de fotogramas mostrados por segundo se denomina tasa de fotogramas (en inglés, frame rate).

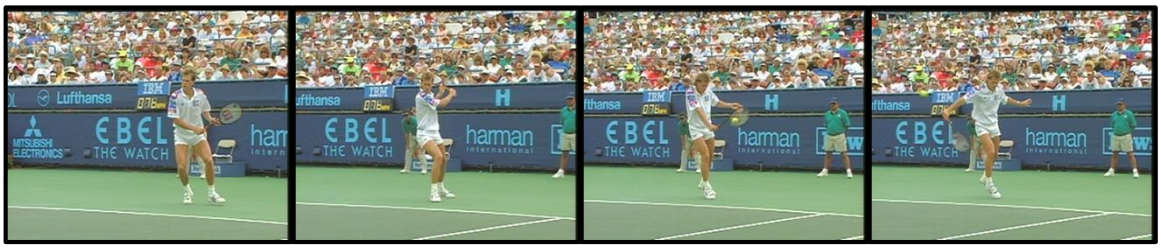

Figura 1. Secuencia de imágenes que conforman un video
Fuente: el Autor

Para obtener la sensación de movimiento, es necesario utilizar una velocidad de fotogramas apropiada ver Figura 1. Con una tasa muy baja, el ojo humano percibe el vídeo como una sucesión de imágenes estáticas, y no como una imagen en movimiento. Con una tasa más alta, la sensación de continuidad de los movimientos mejora, pero son necesarios más bytes. Las tasas adecuadas para reproducir video, están dentro del rango de 24 a 30 fps (frames por segundo) con los que una persona percibe continuidad en los movimientos. En la práctica los vídeos de alta calidad funcionan con 24, 25 o 30 fps. Los vídeos con tasas de fotogramas más pequeños se pueden usar, cuando es necesario reducir el volumen de datos en detrimento de la calidad. En este caso, si, además de reducir la velocidad de cuadro, también se puede utilizar una resolución más baja.

Por ejemplo, si un vídeo con resolución de 640x480 pixeles a 12 fps, consumirá un ancho de banda de casi 100 Mbits/s, si no se utiliza compresión.

Dependiendo de los diversos parámetros de calidad utilizados en la compresión, es posible obtener una reducción en el espacio del almacenamiento y la tasa de bits, del orden de cientos de veces. La codificación de vídeo no sólo pretende establecer un formato estándar, compatible para diferentes sistemas, sino principalmente busca la compresión de estos datos.

1.1.1 Proceso de codificación de video

En la codificación, se tiene que tener en cuenta principalmente que un archivo de vídeo no comprimido muestra una cantidad muy grande de datos, pero una cantidad no tan grande información, pues existe mucha redundancia. Áreas homogéneas de cada fotograma con valores igual o muy próximos, eso es lo que se llama redundancia espacial.

Otro concepto a tener en cuenta es la redundancia temporal, (ver Figura 3) donde también ocurren las similitudes en los valores de los píxeles de la misma posición, que ocurre, por ejemplo, en las áreas con ausencia de movimiento. Incluso cuando existe movimiento, ocurre todavía algún grado de redundancia temporal, ya que los fragmentos enteros de un fotograma pueden aparecer idénticos en los fotogramas siguientes, pero desplazados de posición.

Figura 2. Redundancia espacial.
Fuente [2]

La redundancia espacial tiene lugar dentro de cada fotograma. Ésta, viene asociada al hecho de que la naturaleza está llena de objetos sólidos con superficies y texturas uniformes; los decorados, los paisajes, e incluso los rostros no varían significativamente la información de pixel a pixel, sino que encontraremos generalmente grandes superficies sin variación. [2]. Por ejemplo, en la Figura 2, se puede observar que en el rostro de la mujer existen fotogramas con un porcentaje sustancialmente alto de similitud, en la pared que se observa justo detrás del rostro de la mujer, el caso es el mismo y en la pared aunque se aprecian a groso modo, dos tonalidades, en cada una se repiten secuencialmente los Frames.

Las diversas técnicas para compresión de vídeo exploran estas redundancias para lograr una disminución significativa en la cantidad de bits necesarios para la representación de la secuencia de vídeo.

Además de explorar la redundancia, las técnicas de compresión de vídeo ahorran aún más bits, cuantificando información que sea visualmente imperceptible. Es posible reducir bastante la tasa de bits, sin que sea percibida pérdida de calidad. A partir de cierto punto, se puede reducir más, pero la tasa de bits, pero con alguna pérdida de calidad. Es posible establecer un compromiso entre calidad y tasa de bits [2].

Figura 3. Redundancia temporal: cambio mínimo de fotograma en fotograma.
Fuente [2].

1.1.2 Tipos de fotogramas

Los fotogramas de una secuencia de vídeo, al ser codificados, pueden sufrir reducciones que activan las redundancias espaciales y temporales. En compresión de vídeo, la mayoría de los métodos que utilizan estas redundancias introducen distorsiones controladas para reducir la tasa de bits, estas distorsiones, aunque pequeñas, se acumulan, lo que podría provocaría errores intolerables. Después de la codificación de un video, este estará conformado por los siguientes tipos de fotogramas:

- **I (Intra frames**): utilizan únicamente información contenida en el propio fotograma y no dependen de la información de otros fotogramas, esto recibe el nombre de codificación intra-frame.
- **P (Predicted frame**): se obtiene mediante técnicas que exploran la redundancia temporal, y su información se refiere a la diferencia entre sus valores y los valores del fotograma I utilizados como referencia. Frames P ocupan menos bits que el fotograma I.
- **B (Bi-Predicted Frames):** en la generación de estos fotogramas, la estimación de movimiento se hace dos veces, utilizando como referencia un fotograma anterior y posterior. La predicción se hace sobre la base de la media de los errores, o del menor error. Los fotogramas B tienden a ser aún más pequeños que los fotogramas P.

En la Figura 4 se describe gráficamente las características de cada tipo de fotograma y la interdependencia que hay entre ellos.

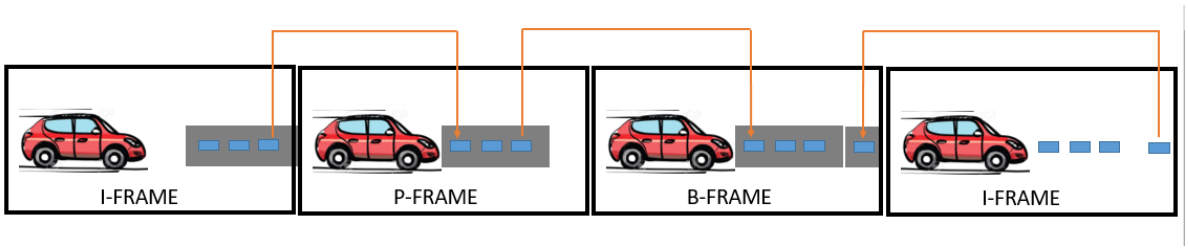

Figura 4. Dependencia entre los fotogramas
Fuente: el Autor.

Un grupo de fotogramas que comienza con uno tipo I, y este, va hasta el anterior, al siguiente de clase I se llamará GOP (Group of Pictures). El GOP es una secuencia de fotogramas, con longitud constante, que se repite en todo el vídeo. Restringimos el estudio al caso del GOP cerrado, que comienza siempre con uno de referencia I. También existe el GOP abierto, que puede utilizar como referencia, clase I o P pertenecientes a GOPs vecinos (ver Figura 5), se presentan tres ejemplos de posibles estructuras temporales:

IPP: En esta estructura, no se utilizan fotogramas B.

Ejemplo: IPPPPPPPPPPPP es un GOP con longitud 12 y estructura IPP

IBP: En esta estructura se inserta un fotograma B entre los fotogramas I y P

Ejemplo: IBPBPBPBPBPB es un GOP con longitud 12 y estructura IBP 37

IBBP: Se insertan 2 frames B entre I y P

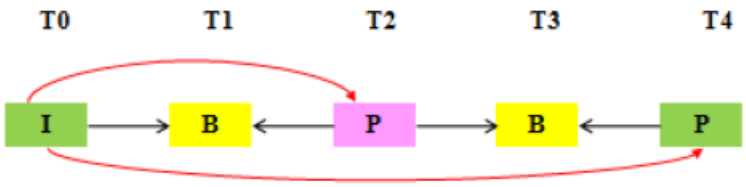

Figura 5. Estructura de codificación típica de H.264/AVC
Fuente [3]

El codificador siempre intentará representar los macrobloques de los frames P y B utilizando vectores de movimiento y valores residuales, para su uso, junto con el respectivo del fotograma de referencia, para poder recuperar el del vídeo original. En ciertas situaciones esto no es posible, es decir, se hace la búsqueda de uno en un área del fotograma de referencia, pero el mismo no es encontrado. Esto ocurre cuando hay mucho movimiento en la escena. En ese caso, el macrobloque puede ser representado sin la explotación de redundancia temporal, por ser la misma inexistente. Se trata entonces de uno de tipo I [4].

.

1.1.3 Fundamentos del estándar H.264/AVC

El estándar es publicado simultáneamente por ISO, como la parte 10 de la norma ISO/IEC 14496 (Advanced Video Coding o AVC) y con el nombre H.264, por parte de la ITU. Posteriormente, se han publicado cuatro nuevas versiones: una en el año 2005 y otras dos en el año 2007.

Al igual que todos los estándares mostrados anteriormente, la norma define una serie de perfiles y niveles. La relación de perfiles y niveles se ha ido modificando a lo largo de las versiones del estándar que se han publicado.

Posteriormente, en la segunda versión de la norma, se añadieron cuatro nuevos perfiles orientados fundamentalmente al desarrollo de aplicaciones profesionales, incluyendo características no soportadas por los perfiles anteriores veinte cinco.

La tercera versión del estándar ha incluido cinco nuevos perfiles orientados a aplicaciones profesionales.

La cuarta versión incluyó la codificación de vídeo escalable SVC dentro de la cual se definen tres nuevos perfiles (Scalable Baseline, Scalable High y Scalable High Intra). Finalmente, la última versión publicada para la codificación multivista en el que se definen dos nuevos perfiles (Stereo High Profile y Multiview High Profile).

La estructura básica del proceso de codificación con H.264 se puede apreciar en la Figura 6, donde se describen cuatro pasos claves (Pre-procesamiento, Codificación, Decodificación, Post- Procesamiento & Recuperación de Errores) que siempre se llevan a cabo para este fin.

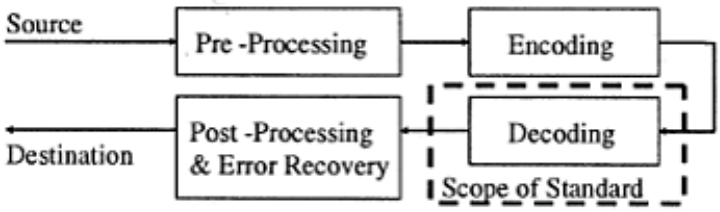

Figura 6. Estructura Básica de la codificación H.264.
Fuente [5]

Conceptualmente, la estructura de un codificador H.264 / MPEG-4 AVC consiste en una capa de codificación de video (VCL) y una capa de adaptación de red (NAL). Mientras que el VCL crea una representación codificada del contenido fuente, el NAL formatea estos datos y proporciona información de encabezado de una manera que permite una personalización efectiva del uso de datos de VCL para una amplia variedad de sistemas.

1) **Video Coding Layer (VCL):** Unidad que traduce la información de video en una secuencia de bits. Es la capa donde se hace la eliminación de la redundancia y se crean los distintos fotogramas (I, P, B) y tiene como objetivo representar de manera eficiente el contenido del video.

2) **Capa de Abstracción de Red (NAL, Network Abstracción Layer):** Aplica un formato a al flujo de bits proveniente de la capa VCL. Esto consiste en crear fragmentos de bits con su propia cabecera. Esta proporciona información sobre el fotograma que se está transmitiendo y que puede ser utilizada por los protocolos de transporte. En pocas palabras, esta capa, mapea y empaqueta el flujo de bits VCL en unidades antes de la transmisión o almacenamiento. Los paquetes de bits conformados en esta capa se llaman NALU (NAL Units). Las NALUs está conformadas asi:
   - El primer byte es del encabezado que contiene la indicación del tipo de datos.
   - El byte restante contiene datos de carga (también llamado RBSP).
   - Los datos de carga útil se entrelazan según sea necesario con los bytes de prevención de emulación, lo que impide que el prefijo de código de inicio se genere dentro de la carga útil.
3) **Estructura o trama H.264**

Básicamente es necesario tener el video fuente, el cual pasará por las tres capas fundamentales que son: capa de codificación de video, particionamiento de datos y la capa de abstracción de red, donde los datos de salida se agrupan en el byte del encabezado, identificando el tipo de datos del contenido, y el byte RBSP que contiene los datos de carga (Ver Figura 7).

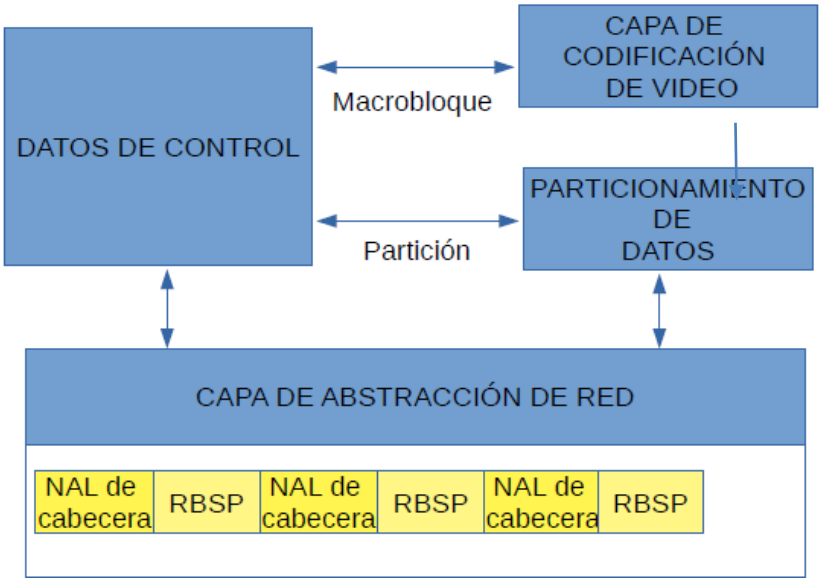

Figura 7. Estructura H264.
Fuente: el Autor.

1.2 FUNDAMENTOS DE VIDEO 3D.

Para comenzar hablar de la importancia de reconocer el papel de lo que es el formato 3D para el proyecto, se hará hincapié en que el mayor desafío en la representación y transmisión de vídeo 3D es la elevada tasa de bits, resultante del hecho de que es necesario representar al menos dos vistas. El método más simple es transmitir dos flujos de vídeo, uno para la

visualización por el ojo izquierdo, otro por el derecho. Ambas imágenes son presentadas simultáneamente en la pantalla, en fotogramas alternos, y a través de gafas especiales, las imágenes para cada ojo son separadas y reconstruidas por el cerebro como una escena 3D.

Otra propuesta es utilizar el display auto-estereoscópico, que dispensa el uso de gafas. Sin embargo, para que funcione, es necesario que reciba varios flujos de cuarenta y tres vídeos, originarios de varias cámaras. Como se transmiten varias vistas, la tasa de bits es alta [6].

1.2.1 Percepción tridimensional del sistema visual humano.

Esta sección introduce al lector en el funcionamiento del sistema visual, y analiza en detalle los indicadores o claves que llevan al mismo a percibir el entorno de manera tridimensional.

El sistema visual de los humanos y animales permite a los individuos asimilar información del entorno.

El acto de ver comienza cuando la lente del ojo enfoca una imagen de los alrededores en una membrana de la parte posterior del ojo que es sensible a la luz, llamada retina. La retina es en realidad parte del cerebro, aislada para servir como transductor para la conversión de patrones de fotones de luz en señales neuronales [6].

La percepción de colores se basa en los distintos conos retínales situados en el ojo. Los conos son células especializadas, que contienen diferentes pigmentos, sensibles a diferentes espectros de color. En los humanos, existen tres conos sensitivos a tres espectros diferentes, resultando en una visión de color tricromática [6].

1.2.2 Percepción de profundidad de una escena

La percepción de profundidad es la habilidad visual de percibir el mundo en tres dimensiones, y la distancia respecto a objetos.

Surge a partir de una variedad de indicadores, clasificados típicamente en claves binoculares, es decir, que requieren de ambos ojos, y monoculares cuando únicamente requieren la visión de un ojo.

En la Figura 8 se muestran las partes del ojo humano que trabajan en conjunto y posibilitan la percepción y traducción de efectos visuales para la comprensión del formato 3D, así como se ejemplifica con dos líneas punteadas la forma como se percibe la imagen.

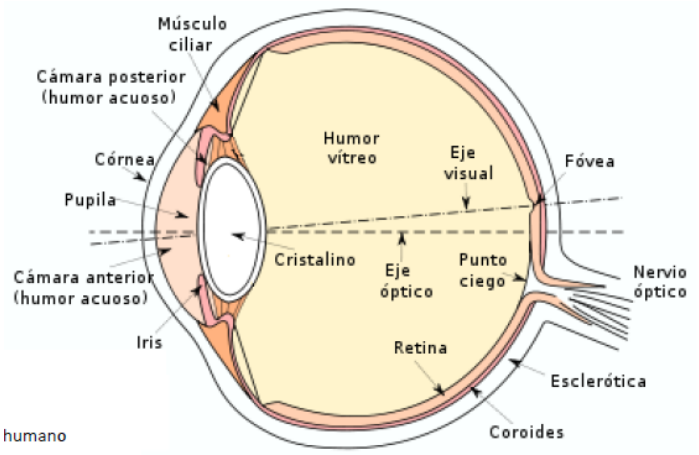

Figura 8. Sección de un ojo humano
Fuente [6]

1.2.3 Claves Monoculares

También llamadas claves monoscópicas, proveen información de profundidad al ver la escena con un único ojo.

- **Forma conocida**: tenemos recuerdo de la forma de los objetos que nos hemos encontrado previamente, y confirmamos dicha información cada vez que volvemos a ver dicho objeto. Aunque este indicador corresponde más a la percepción de forma, también contribuye a la percepción de profundidad. Por ejemplo, en la Figura 9 el objeto de la izquierda se interpreta como un balón de fútbol esférico, aunque fuese en realidad la imagen plana tumbada de la derecha.

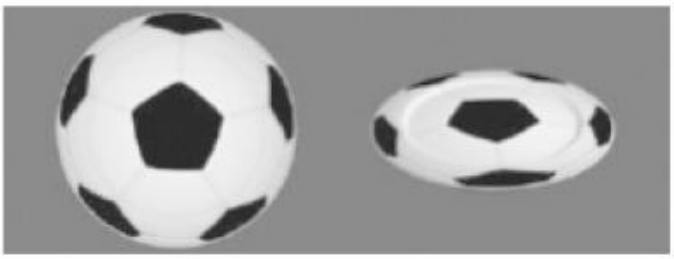

Figura 9. Ejemplo de forma conocida.
Fuente [6]

- **Tamaño relativo:** si se conoce que dos objetos son del mismo tamaño, por ejemplo, dos árboles, pero su tamaño absoluto es desconocido, el tamaño relativo puede proporcionar información sobre la profundidad relativa de dichos objetos. Si uno de los árboles subtiende un ángulo visual con la retina mayor que el otro, parecerá estar más cerca. Por ejemplo, en la Figura 10 se observa una persona y un edificio, los cuales aunque tienen el mismo tamaño, asumimos que el edificio debe estar más lejos.

Figura 10. Ejemplo de tamaño relativo.
Fuente [6]

- **Tamaño familiar:** dado que el ángulo proyectado por un objeto en la retina decrementa con la distancia, esta información se puede combinar con conocimientos previos del tamaño del objeto para determinar la profundidad absoluta del mismo. Por ejemplo, el tamaño del automóvil medio es bien conocido. Combinando dicha información con el ángulo que un coche subtiende con la retina, se determina la profundidad absoluta del mismo en una escena.

- **Color:** Los colores cálidos se perciben más cercanos, mientras que los colores fríos se perciben más lejanos (Ver Figura 11).

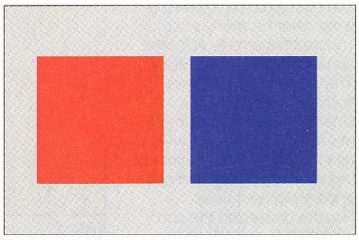

Figura 11. Los colores cálidos avanzan y los fríos retroceden en un fondo neutral.
Fuente [6]

1.2.4 Técnicas tradicionales

1) **Anáglifos**

Un anáglifo consiste en una estereografía en la que se hallan superpuestas dos imágenes tomadas o tratadas con filtros de colores distintos. Para la visualización de la Figura 12 se requieren de unas gafas anaglíficas, que, mediante filtros de colores distintos, hacen llegar a cada ojo únicamente la imagen que le corresponde, creando el efecto estereoscópico.

Las imágenes anaglíficas pueden ser creadas de diversas maneras: empleando los de negativos, colorearlas con ayuda de un ordenador, o proyectar diapositivas desde dos proyectores equipados con filtros.

Puede verse la superposición de un par estéreo de imágenes, una codificada en rojo, y la otra en cyan.

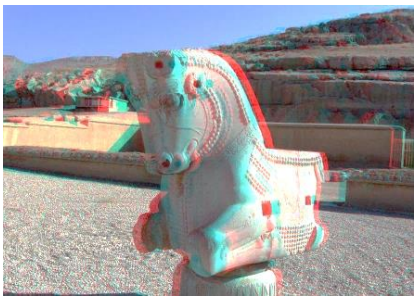

Figura 12. Imagen anaglífica.
Fuente [2]

2) **Estereoscopia tradicional**

La fotografía estereoscópica tradicional consiste en crear la ilusión de 3D a partir de dos imágenes 2D, es decir, un estereograma o par estéreo. La manera más fácil de realzar la percepción de profundidad en el cerebro es proveer a los ojos de dos imágenes distintas, representando dos perspectivas del mismo objeto con una desviación mínima, exactamente correspondientes a las perspectivas que ambos ojos reciben de forma natural en la visión binocular. A este método se lo conoce como Side-by-side (lado a lado) por la disposición de las imágenes, en la Figura 13 se puede apreciar un ejemplo de la perspectiva visual de la estereoscopia tradicional.

Las herramientas utilizadas para este fin, se pueden observar en la Figura 14. En resumen, es una estructura con dos cámaras sincronizadas que dependiendo los puntos a los que ambos focos apunten generando un efecto de visualización estéreo.

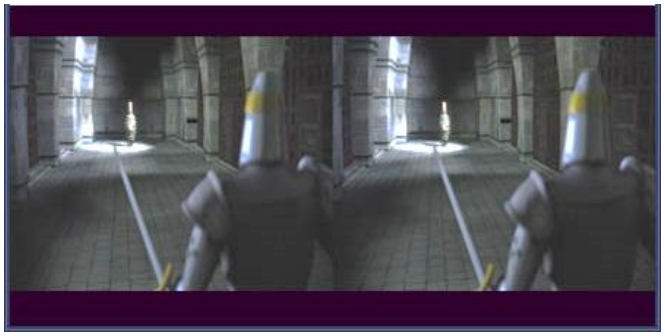

Figura 13. Vista izquierda & Vista derecha, Video 3D Estéreo.
Fuente [7]

Figura 14. Tipo de Cámaras Utilizadas En Video 3D Estéreo.

Fuente [8]

Para evitar la distorsión y el cansancio de la vista, cada una de las imágenes 2D debería ser presentada al ojo correspondiente del espectador, y de manera que cualquier objeto a una distancia infinita sea visto con los ojos completamente paralelos. Cuando la imagen no contiene objetos en la distancia infinita, como son el horizonte o una nube, las imágenes 2D deberían estar menos espaciadas la una de la otra.

1.2.5 Video Plus Depth (V + D)

En el video plus Depth (V + D), una escena en 3D está representada por un video en color y unos datos de mapas de profundidad asociados. El video a color es como cualquier vista de CSV, y el mapa de profundidad es un video monocromático, con un componente de luma solamente. En el mapa de profundidad, los objetos cercanos aparecen más brillantes, mientras que los objetos lejanos aparecen más oscuros. El V + D se ilustra en la Figura 15 [9].

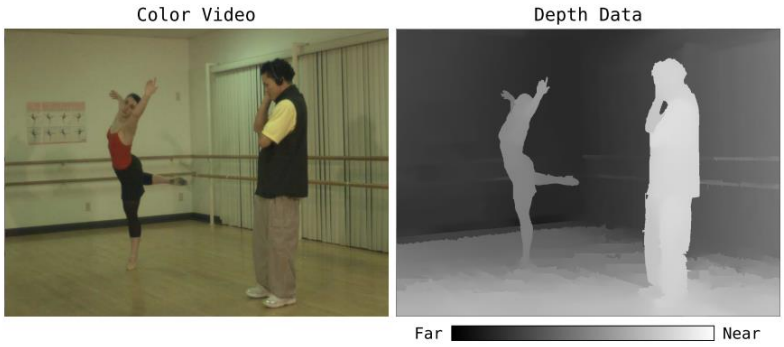

Figura 15. Representación V + D ("Ballet")
Fuente [9]

1.2.6 Multiview Video plus Depth (MVD)

La representación Multiview video plus depth (MVD) es una extensión de V + D. Cuenta con más de dos vistas, con un video en color y un mapa de profundidad asociado para cada uno de los puntos de vista. Por lo tanto, proporciona una serie de puntos de vista diferentes para el observador y para esta razón se usa principalmente para televisión 3D o aplicaciones de video de punto de vista libre (Ver Figura 16) [9].

1.3 FUNDAMENTOS DEL ESTÁNDAR H.264/MVC

Multiview Video Coding (MVC) "Codificación de Video Multivista", está concebido como una extensión del estándar de compresión de video H.264 Y MPEG 4 AVC, que popularmente es conocido como Advanced Video Coding, "Codificación Avanzada de Video", el cual se desarrolló gracias a los avances conjuntos de MPEG/VCEG (Grupo de expertos en codificación de video), que admite una eficiente codificación de secuencias capturadas simultáneamente desde varias cámaras de video con un único flujo de datos.

El MVC, está orientado a la codificación de vídeo estereoscópico, y también para la FTV (Televisión de punto de vista libre) y la televisión 3D. Éste sistema de codificación MVC, ha estado pensado para que también sea compatible con H.264/AVC, de forma que permita a los dispositivos antiguos decodificar el vídeo estereoscópico, ignorando la información adicional del segundo punto de vista y mostrando la imagen en 2D.La codificación de video multivista, es una extensión que se hizo para el estándar H.264 (MPEG-4), y en este se contempla la posibilidad de que desde la adquisición se obtengan múltiples vistas de una escena, y que estas sean comprimidas y transmitidas como un perfil multivista. Con la aparición de esta extensión a la norma, no solo se fortalece el concepto del 3D basado en la existencia de dos imágenes como en el sistema visual, sino también surge el concepto de FVV (Free View-point Video), en el cual se adquieren n vistas, y se presentan al espectador n vistas de las escenas. Para el caso particular en el que n=2 vistas corresponde al sistema usado en la actualidad en los sistemas Blu-Ray 3D [10].

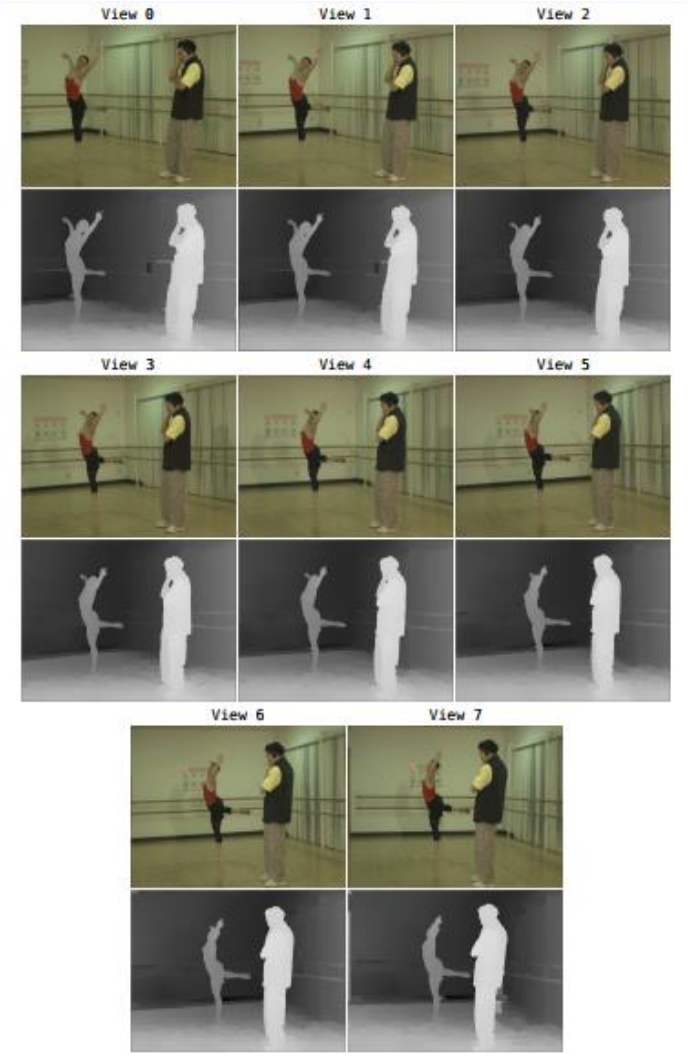

Figura 16. Representación Multiview Video Plus Depth ("Ballet")
Fuente [9]

La estructura general de MVC que define las etapas del modelo se ilustran en la Figura 17, el codificador recibe N secuencias de vídeo sincronizadas temporalmente y genera un flujo de bits. El decodificador recibe el flujo de bits, decodifica y emite N señales de vídeo que pueden utilizarse para diferentes propósitos: generar una o N vistas de una vista estéreo.

La predicción a través de vistas, como se muestra en la Figura 18, se utiliza para aprovechar la redundancia entre cámaras con la limitación de que la predicción entre vistas sólo se efectúa desde la misma instancia de tiempo y no puede exceder el número máximo de imágenes de referencia almacenadas [11].

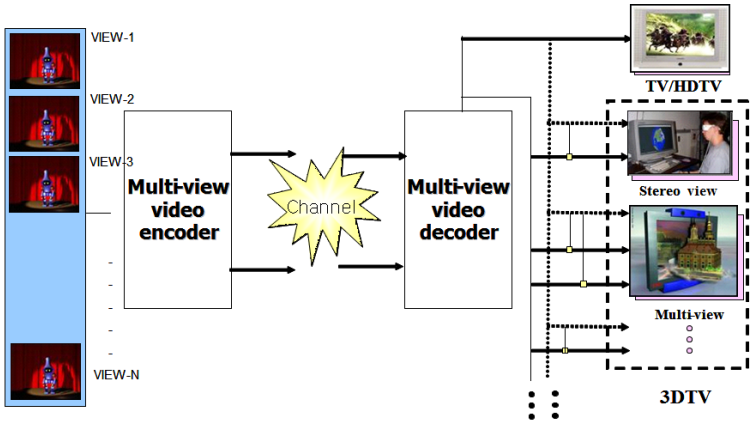

Figura 17. Modelo MVC
Fuente [12]

MVC, consiste en secuencias de vídeo capturadas por varias cámaras desde diferentes ángulos y lugares.

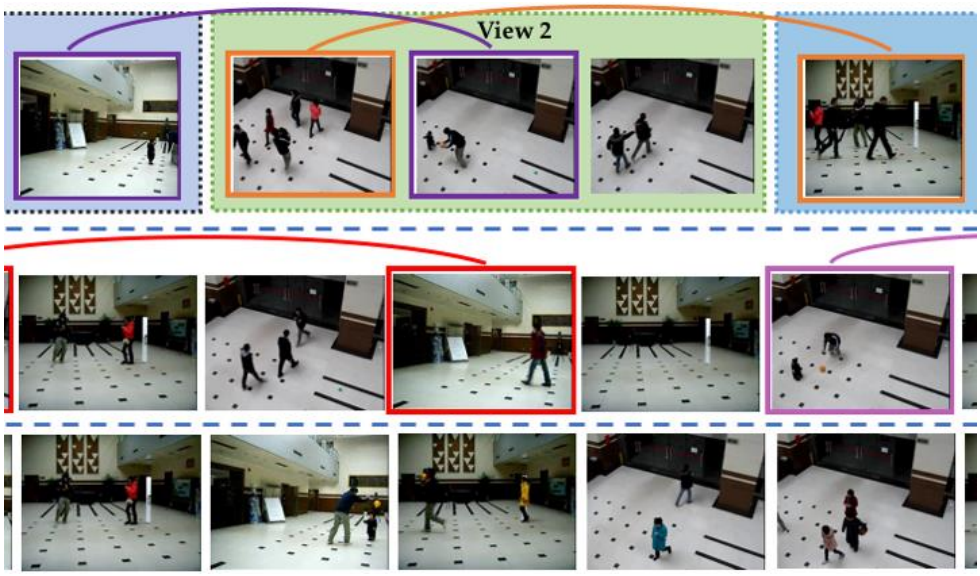

Figura 18. Diferentes tomas de varias escenas para MVC.
Fuente [13]

- La combinación temporal y la predicción de la vista temporal son las claves para hacer codificaciones eficientes en MVC.
- Esta predicción se basa en la suposición de que los fotogramas de vídeo desde diferentes puntos de vista, pueden ser libremente intercambiados o simultáneamente disponibles en el codificador.
- Así que, un fotograma de una cámara puede predecir fotogramas de la misma cámara, y también los fotogramas de las segundas cámaras.

1.3.1 Modalidades de predicción de MVC

- Predicción desde imágenes anteriores.
- Predicción desde imágenes posteriores.
- Predicción desde imágenes interpoladas.
- Predicción desde imágenes modificadas.

Las **imágenes I, P y B** son las imágenes que se codifican por sí mismas, se predicen por extrapolación y se predicen por interpolación respectivamente.

La elección de la estructura de predicción afecta en el retardo, las características de la memoria, o el acceso aleatorio a una determinada imagen.

1.3.2 Aplicaciones de MVC.

Las aplicaciones más comunes y que se han venido explorando con el fin de estudiar codificación de video multivisa son principalmente Vídeo 3D (3DV), y Vídeo con Punto de vista libre (FVV) son los nuevos tipos de medios visuales en los que se puede aplicar el MVC, y así ampliar la experiencia del usuario más allá de lo que ofrece el vídeo 2D.

Vídeo 3D:

- Transmite sensación de profundidad dentro de una escena 3D.
- Aplicaciones en videoconferencias, 3DTV.

Punto de vista libre (FVV):

- Proporciona la capacidad de selección interactiva del punto de vista y dirección dentro de un radio de acción determinado.
- Aplicación en vigilancia, deportes, conciertos.

En la Figura 19, un video multivista primero es capturado y luego procesado por un codificador de video (MVC). Un servidor transmite el código Bitstream (s) a diferentes clientes con diferentes capacidades, posiblemente a través de puertas de enlace de medios. El portal de medios es un dispositivo inteligente, también conocido como una red de medios de comunicación elemento (MANE), que está en el contexto de señalización y puede manipular los paquetes de video entrantes (en lugar de simplemente reenviar paquetes). En la etapa final, el video se decodifica y se representa con diferentes medios de acuerdo al escenario de la aplicación y las capacidades del receptor.

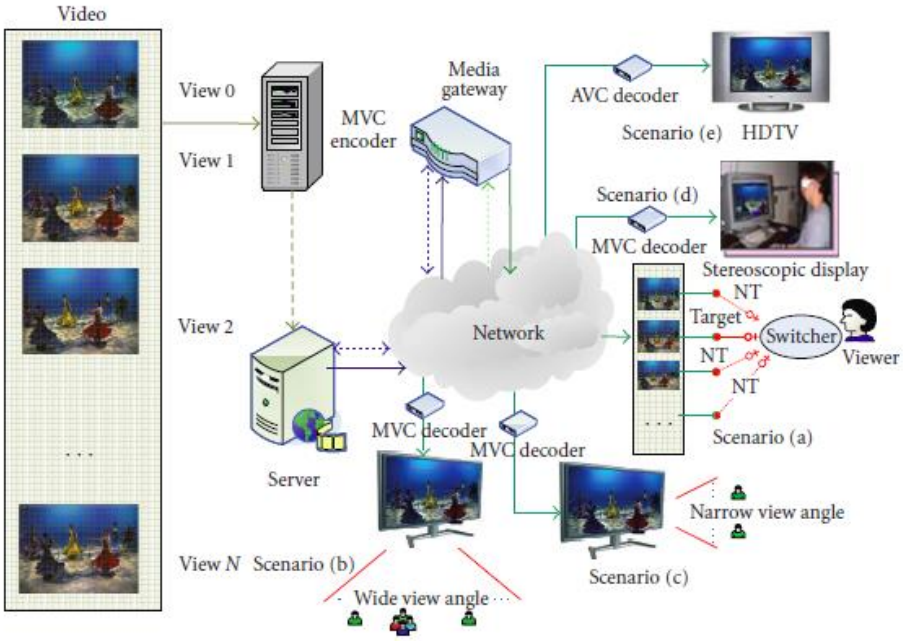

Figura 19. Arquitectura del sistema MVC.
Fuente [14]

1.4 CODIFICACIÓN DE VIDEO MVC

Para codificar un video en formato MVC es necesario la instalación y ejecución de un programa codificador. Existen varias herramientas software como: (WZC– Wyner-Ziv Coder), Slepian-Wolf (SWCSlepian Wolf Coder) y JMVC (Joint Multiview video coding). El software de referencia es el JMVC [15], el cual fue el utilizado para el desarrollo de este proyecto.

A continuación, se describe el proceso de desarrollado para codificar un video en formato MVC. El desarrollo práctico fue ejecutado en el sistema operativo *UBUNTU 16.04* y, por tal motivo, se instaló la versión compatible con el mismo.

También se describen los pasos de la instalación del codificador y la posterior codificación de un video de prueba.

1.4.1 Instalación del codificador

El primer paso es instalar los editores de texto necesarios para programar, compilar e identificar los procesos para ejecución del proyecto. A continuación, se describen los comandos y los pasos para realizar el mencionado procedimiento.

- Inicialmente se verificó la versión del compilador *"gcc"*, idealmente en el manual del software se sugiere la versión 4.0.0 del compilador; para el proyecto, utilizando la versión de Ubuntu 16.04, la versión del compilador utilizada fue la 5.4.0, ya que experimentalmente se comprobó que funcionaba en óptimas condiciones y no existía la necesidad de buscar una versión anterior.

Procedimiento: se abre una terminal en *Ubuntu 16.04* tecleando `(ctrl + alt + t)`, y en la terminal se ejecuta el código: `gcc -v`

El segundo paso es descargar el JMVC SOFTWARE de la página de github utilizando la siguiente dirección web o URL:

- `https://github.com/cmurphy/JMVC`

- Se extraen los archivos de la carpeta descargada (Ver Figura 20).

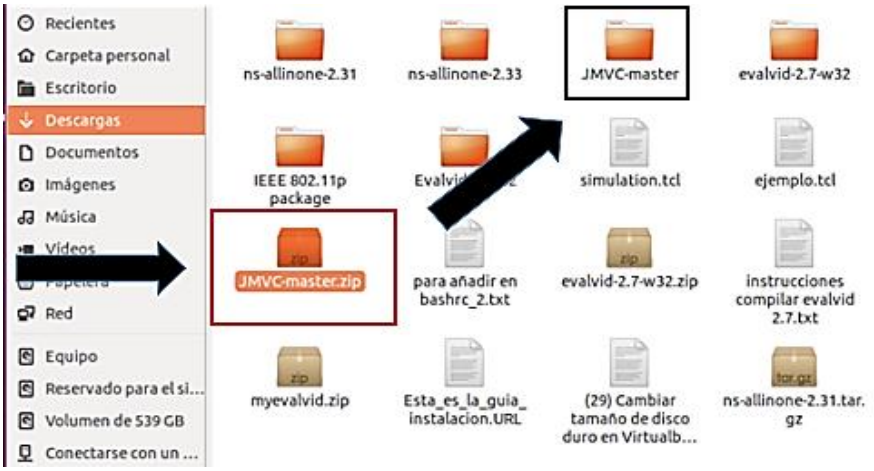

Figura 20. Visualización de la carpeta generada a partir de la extracción de archivos.
Fuente: el Autor.

- Se abre una terminal estando dentro de la carpeta extraída *"JMVC-master"* o digitando `cd JMVC/H264AVCExtension/build/Linux` y se digita el siguiente comando para instalar todos los paquetes, librerías y ejecutables del software:

```
make
```

- Luego de haber ejecutado el anterior comando, se verificó dentro de la carpeta JMVC-master, que se hallan creado las carpetas *"bin"* y *"lib"*, las cuales contienen las herramientas utilizadas para la codificación del video MVC (Ver Figura 21).

Figura 21. Visualización de las carpetas generadas a partir de la ejecución del comando make.
Fuente: el Autor.

- Dentro de la carpeta *"bin"*, se encuentran los siguientes archivos que serán los ejecutables de las diferentes herramientas de edición de video del software [16].

```
bin/DownConvertStatic
bin/DownConvertStaticd
bin/H264AVCDecoderLibTestStatic
bin/H264AVCDecoderLibTestStaticd
bin/H264AVCEncoderLibTestStatic
bin/H264AVCEncoderLibTestStaticd
bin/MVCBitStreamAssemblerStaticd
bin/MVCBitStreamAssemblerStatic
bin/MVCBitStreamExtractorStaticd
bin/MVCBitStreamExtractorStatic
bin/PSNRStatic
bin/PSNRStaticd
```

- Dentro de la carpeta *"lib"*, se encontrarán las siguientes librerías internas del software [16].

```
lib/libH264AVCCommonLibStatic.a
lib/libH264AVCCommonLibStaticd.a
lib/libH264AVCDecoderLibStatic.a
lib/libH264AVCDecoderLibStaticd.a
lib/libH264AVCEncoderLibStatic.a
lib/libH264AVCEncoderLibStaticd.a
lib/libH264AVCVideoIoLibStatic.a
lib/libH264AVCVideoIoLibStaticd.a
```

- Después de haber confirmado que se hallan creado correctamente las librerías y ejecutables anteriores, ya está preparado el software para codificar videos MVC.

1.4.2 Codificación de un video multivista

En primer lugar, se realizó la búsqueda de un video MVC en formato YUV para realizar posteriormente la codificación. El video seleccionado se llama *"BALLROOM"* y fue descargado desde: `http://www.merl.com/pub/avetro/mvc-testseq/orig-yuv/ballroom/`. Este video está conformado por 8 vistas, cada una almacenada en un archivo (. yuv). Cada una de las vistas debe ser codificada en por el programa JMVC. En la Figura 22 se muestran las 8 vistas descargadas.

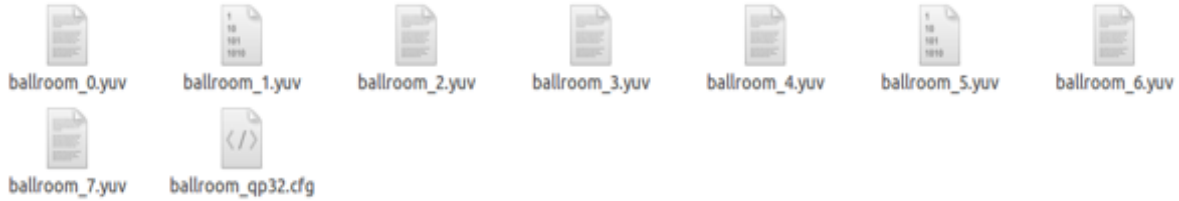

Figura 22. Visualización de los archivos formato (. yuv) descargados.
Fuente: el Autor.

En la Figura 23 se muestra un fotograma extraído de la vista 1 del video "*BALROOM"*

Figura 23. Frame tomado del video BALROOM
Fuente: el Autor

Luego de tener las vistas descargadas, se creó un archivo (.cfg). El cual fue llamado *"ballroomConfig.cfg"*, donde se configuraron los parámetros de codificación del video. Los parámetros básicos de configuración que contiene el archivo, se muestran en la Figura 24. El contenido completo de este archivo se muestra en el Anexo 7.3.

COMANDOS TESIS

```
# JMVC Main Configuration File
#====================== GENERAL ======================================
InputFile               /home/juancho/Documentos/ballroom/ballroom      # input file
OutputFile              stream      # bitstream file
ReconFile               rec         # reconstructed file
MotionFile              motion      # motion information file
SourceWidth             640         # input  frame width
SourceHeight            480         # input  frame height
FrameRate               25.0        # frame rate [Hz]
FramesToBeEncoded       250         # number of frames
#====================== CODING =======================================
```

Figura 24. Configuración de los parámetros del codificador
Fuente: el Autor.

- **InputFile** → `/home/Juancho/Documentos/balroom/balroom` : en el "*InputFile"* se ingresa la ruta donde se encuentra el video que va a ser modificado, para el caso de éste proyecto, la ruta es la que se puede ver anteriormente separada por el siguiente símbolo (/). Dichos archivos deben tener en el formato in_0.yuv, in_1.yuv según corresponda dependiendo de la cantidad de las vistas [16].
- **OutputFile** → (Stream): en este apartado, se especifica el nombre del archivo de salida o archivo ya codificado. Creará uno automáticamente para cada una de las vistas [16].

- **ReconFile** → (rec): Especifica el nombre del archivo de la secuencia de entrada codificada y reconstruida. Se genera automáticamente luego de la codificación [16].
- **MotionFile** → (motion): indica el modo salto de movimiento. Se creará automáticamente dependiendo las vistas [16].
- **SourceWidth** → (640): Especifica el ancho de las imágenes de entrada en muestras de luma. Éste parámetro, será distinto de cero y un múltiplo de 16 [16].
- **SourceHeight** → （480）: Especifica el largo de las imágenes de entrada en muestras de luma. Éste parámetro, será distinto de cero y un múltiplo de 16 [16].
- **FrameRate** → （25.0）: Especifica la velocidad de fotogramas de la secuencia de entrada en Hertz [16].
- **FramesToBeEncoded** → (250): Especifica el número de fotogramas de la secuencia de entrada que se codificará para una vista [16].

Los códigos ejecutados para codificar cada una de las vistas, fueron los siguientes:

```
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 0
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 2
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 1
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 4
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 3
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 6
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 5
./H264AVCEncoderLibTestStaticd -vf /home/Juancho/Documentos/ ballroomConfig.cfg 7
```

Como resultado, el codificador crea los archivos con el nombre *"stream"* para cada una de las vistas con comprimidas en el formato h.264 (Ver Figura 25).

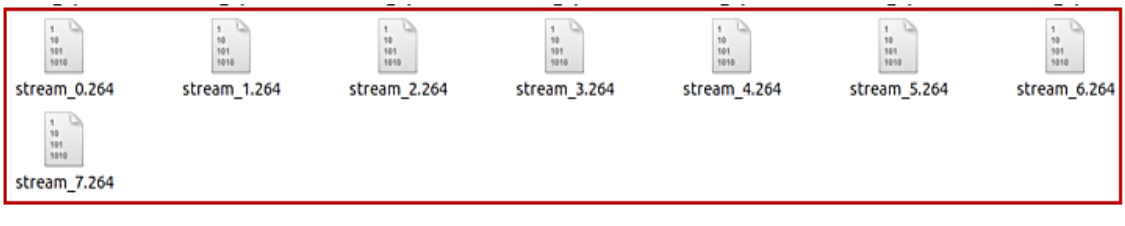

Figura 25. Visualización de los archivos de las vistas codificados en formato h.264
Fuente: el Autor.

CAPÍTULO 2

2 PROTOCOLO 802.11P

En este capítulo se presenta una descripción del protocolo 802.11p, en la cual, se muestran inicialmente los fundamentos necesarios para reconocer los aspectos más importantes del protocolo, y se citan algunos conceptos básicos que permitieron el entendimiento y posterior desarrollo de las configuraciones y parámetros insertados en "*NS2"* al momento de modificar el entorno de simulación.

2.1 FUNDAMENTOS DEL ESTÁNDAR IEEE 802.11P (ESTÁNDAR PARA REDES VEHICULARES VANET)

Las redes vehiculares dedicadas de corto alcance (DSRC) se rigen específicamente de acuerdo al estándar 802.11p. DSRC permite la transmisión y recepción de paquetes de datos entre vehículos y vehículos (Comunicación V2V) y entre vehículos e infraestructuras (comunicación V2I).

El protocolo IEEE 802.11p se encarga exclusivamente del control de las capas física y de acceso al medio para estas redes. Este estándar proporciona los mecanismos necesarios para para establecer comunicación entre vehículos. La disminución de la latencia en la comunicación es uno de los objetivos principales de este estándar. [17].

2.2 GENERALIDADES

El protocolo 802.11p opera dentro del espectro de frecuencias de 5,90 GHz y 6,20 GHz. Actualmente el ancho de banda medio requerido para las aplicaciones que utilizan estos protocolos es de 6Mbps y con una cobertura de 300 metros, con aspiraciones futuras de 27Mbps y una cobertura de 1km [18].

Las principales características del protocolo, están resumidas en la Tabla I. Además en la Tabla II, se describe otros parámetros técnicos, comparados con el estándar 802.11, con el fin de registrar los principales cambios de un estándar a otro.

Características
Define 7 canales de 10MHz cada uno. (1 canal de control CCH que intercambia mensajes de control de red & 6 canales de servicio SCH de red.)
Soporta transmisión de paquetes IP y mensajes cortos Wave (Wave Short Message)
El ancho de banda es dividido en ciclos de transmisión donde la duración de trama es de 50 milisegundos.
Soporta altas velocidades vehiculares a diferencia del estándar IEEE 802.11a que abarca movilidad baja en interiores en sistemas WLAN.

Define un nuevo tipo de BSS (Basic Service set) llamado WBSS (Wave Basic Service Set) que permite la transmisión de paquetes de datos.
Define las técnicas de señalización WAVE y funciones de interface que son controladas por la IEEE 802.11 MAC.
Utiliza modulación OFDM, sus tasas de transmisión son de 3, 4, 5 , 6, 9, 12, 24 y 27 Mbps.
Usa 52 subportadoras moduladas utilizando BPSK, QPSK, 16 QAM, 64 QAM, con codificaciones de ratios ½, 2/3 o ¾.

Tabla I. Características del protocolo 802.11p.
Fuente [19]

ESPECIFICACIONES	802.11p	802.11g
Velocidad de Transmisión (Tx) Mbps	3,4,5,6,9,12,18,24 y 27	6, 9, 12, 18, 24, 36, 48 y 54
Modulación	BPSK, QPSK, 16-QAM, 64-QAM	BPSK OFDM QPSK OFDM 16-QAM OFDM 64-QAM OFDM
Tasa de Codificación	1/2, 1/3, 3/4	1/2, 2/3, 3/4
Número de Subportadoras	52(= 48+4)	64
Longitud de Símbolo	8 µs	4 µs
Tiempo de Guarda	1,6 µs	1.2 µs
Frecuencia	5,9 GHz	2.4 GHz
Ancho de Banda Utilizado	70 MHz	20 MHz
Número de Canales	7	52
Ancho de Banda por Canal	10 MHz	16.6 MHz

Tabla II. Especificaciones del estándar IEEE 802.11p
Fuente: el Autor

2.3 COMPARACIÓN ENTRE 802.11A Y 802.11P

En general, las principales modificaciones en el estándar 802.11p comparándolo con el tradicional 802.11 se resumirán en la Tabla III.

- Principalmente se redujo la carga a la hora de establecer la comunicación, esto es debido al reducido tiempo de contacto entre los dispositivos.
- En el estándar IEEE 802.11p, se define un nuevo tipo de BSS (Basic Service Set) llamado WBSS (Wave Basic Service Set).
- WBSS tiene un identificador fijo y transmite tramas bajo demanda. Cada trama contiene la información esencial para establecer las comunicaciones, eliminándose el proceso de autenticación.
- Por otro lado, para evitar la exploración de los canales para encontrar la red deseada, las funciones de estos canales están ya fijadas.

- La capa física se basa en el estándar 802.11a y usa modulación OFDM (Orthogonal Frequency-Division Multiplexing).
- El ancho de banda se redujo de 20 Mhz a 10 Mhz, con esto se consigue reducir el retraso de difusión [20].

Presupuesto	***802.11a***	***802.11p***
Velocidad de datos	6, 9, 12, 18, 24, 36, 48, 54 Mbps	3, 4, 5, 6, 9, 12, 18, 24, 27 Mbps
Modulación	BPSK, QPSK, 16QAM, 64QAM	Igual que 802.11a
Tasa de codificación	1/2, 1/3, 3/4	Igual que 802.11a
Número de subportadoras	52	52
Duración del símbolo OFDM	4 µs	8 µs
Tiempo de guardia	0.8 µs	1,6 µs
Duración del preámbulo	16 µs	32 µs
Espaciado de la subportadora	0.3125 MHz	0.15625 MHz

Tabla III. Comparación entre 802.11ª y 802.11p.
Fuente [20]

2.4 APLICACIÓN

Una de las principales aplicaciones del estándar 802.11p son las redes vehiculares, conocidas como VANETs (Vehicular Ad-Hoc Networks). En estas redes los vehículos actúan como nodos de una red de comunicación, donde además hay estaciones fijas llamadas RSU (Road Side Unit). Estas estaciones permiten la propagación de los mensajes a más larga distancia, así como la interconexión con otro tipo de redes. En la Figura 26 se observa un escenario grafico de transmisión y recepción de redes vehiculares haciendo referencia a tres modalidades, inter vehicular, vehicular con las antenas presentes en el camino e inter caminos.

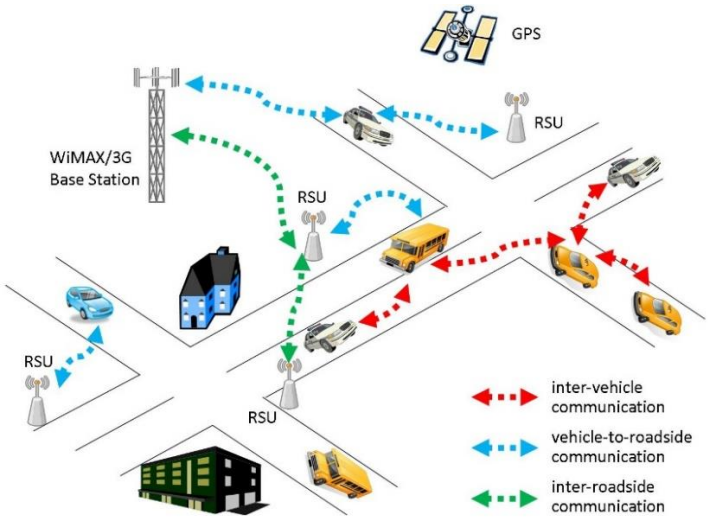

Figura 26. Transmisión de información a diferentes usuarios de una red VANET.
Fuente [19]

En este proyecto se pretende evaluar la transmisión de video sobre este protocolo. Esta evaluación se realizará utilizando el simulador de redes NS-2 (Network Simulator 2) [21]. En el siguiente capítulo se describirá como está conformada la plataforma completa utilizada para hacer la transmisión de un video sobre una red inalámbrica que utilice el estándar 802.11, también se presentan los resultados de una simulación básica de una red inalámbrica 802.11p. Esta simulación fue realizada sobre NS-2, por lo tanto, se describe a continuación como fue realizada la instalación del simulador. Posteriormente se describe la instalación del protocolo dentro del simulador y en la parte final del presente capítulo se presentan los resultados de la simulación.

2.5 INSTALACIÓN DEL SOFTWARE DE SIMULACIÓN NS-2

Inicialmente, antes de configurar el protocolo 802.11p para los efectos de simulación, se instaló el software de simulación NS-2 (Network Simulator 2), con el cual, se realizaron las adecuaciones necesarias para compilar en adelante los diferentes procesadores y protocolos con los que se trabajó para la transmisión de video.

En dicho software, se compilaron los ejecutables y las librerías del protocolo 802.11p, con el fin de asegurar el hecho de tener en un mismo simulador, todos los programas necesarios y así mismo trabajar en cada uno de los procedimientos para la pre transmisión, transmisión y post procesamiento.

El primer paso fue instalar las dependencias necesarias para que el software funcione correctamente, para ello, es necesario abrir una terminal en Ubuntu y escribir el siguiente código:

```
sudo apt-get install build-essential autoconf automake libxmu-dev.
```

El segundo paso fue descargar "*ns-allinone-2.33.tar.gz".* y guardarlo en la carpeta de usuario.

- El link de descarga utilizado para obtener las carpetas de instalación del software es el siguiente:

```
http://ufpr.dl.sourceforge.net/project/nsnam/allinone/ns-allinone-2.35/ns-allinone-2.35.tar.gz
```

En el tercer paso se descomprimió el paquete "*ns-allinone-2.33.tar.gz*" y seguido de esto se instaló en la Figura 27, se observa el comando para descomprimir el archivo y en la Figura 28, el comando para la instalacion.

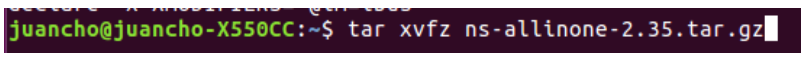

Figura 27. Comando para descomprimir archivo del instalador.
Fuente: el Autor

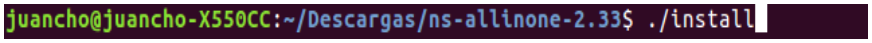

Figura 28. Comando para la instalación del software.
Fuente: el Autor

En el tercer paso, para añadir las variables de entorno que solicita el programa fue necesario abrir el archivo *"bashrc"*. El comando utilizado para entrar directamente en modo edición del archivo fue ingresado desde la terminal de Ubuntu y dicha instrucción es la siguiente (Ver Figura 29).

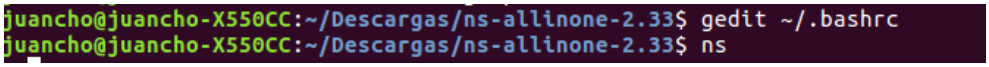

Figura 29. Comando para entrar en modo editor al archivo "bashrc".
Fuente: el Autor.

a) Las variables de entorno modificadas fueron las siguientes:

```
    # LD_LIBRARY_PATH
    OTCL_LIB=/home/programmer/ns-allinone-2.33/otcl-1.13
    NS2_LIB=/home/programmer/ns-allinone-2.33/lib
    X11_LIB=/usr/X11R6/lib
    USR_LOCAL_LIB=/usr/local/lib
    export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB
    :$X11_LIB:$USR_LOCAL_LIB

    # TCL_LIBRARY
    TCL_LIB=/home/programmer/ns-allinone-2.33/tcl8.4.18/library
    USR_LIB=/usr/lib
    export TCL_LIBRARY=$TCL_LIB:$USR_LIB

    # PATH
    XGRAPH=/home/programmer/ns-allinone-2.33/bin:/home/programmer/ns-allinone-
2.33/tcl8.4.18/unix:/home/programmer/ns-allinone-
2.33/tk8.4.18/unix:/home/programmer/ns-allinone-2.33/xgraph-12.1/
    NS=/home/programmer/ns-allinone-2.33/ns-2.33/
    NAM=/home/programmer/ns-allinone-2.33/nam-1.13/
    export PATH=$PATH:$XGRAPH:$NS:$NAM
```

Seguido de esto, se debe recargar el archivo *"bashrc"* ejecutando el comando (Ver Figura 30).

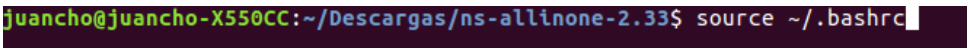

Figura 30. Comando para recargar el archivo "bashrc".
Fuente: el Autor.

Durante la instalación, algunos errores fueron detectados, estos errores, aunque no implicaron gravedad o falla definitiva del software, fue necesario acudir uno por uno durante el proceso, permitiendo así que quedara correctamente instalado NS2. Sin embargo, los principales errores serán documentados en imágenes a continuación, con el fin de esclarecerlos y dar suficiente soporte para que cualquier persona que intente instalar el software, pueda realizarlo en el menor tiempo posible.

- El primer error que surge durante la instalación se puede apreciar en la Figura 31. Para solucionar el error, es necesario cambiar la línea de código "6304" del archivo de texto *"configure"* de la sub carpeta *"otcl-1.13"* de *"ns-2.33"* en la cual aparece la siguiente instrucción: `-SHLIB_LD= "id -shared"` y deberá ser reemplazada por `+SHLIB_LD=" gcc -shared"`

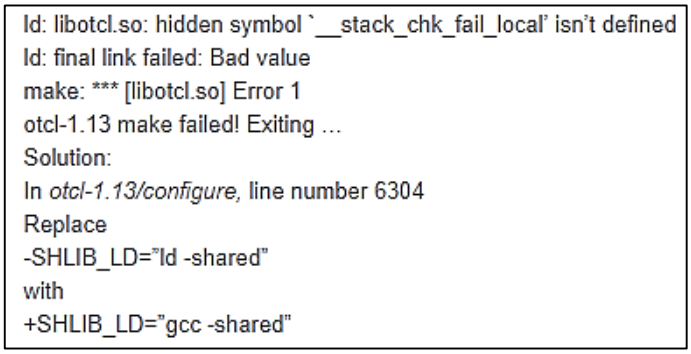

Figura 31. Error #1 instalación NS2.
Fuente [22].

- El segundo error y su eventual solución se puede visualizar en la Figura 32.
  La solución al error consiste en reemplazar la línea 219 del archivo de texto *"ranvar.cc"* de la sub carpeta *"tools"* de "ns-2.33" en la que aparece la siguiente línea de código: `-return GammaRandomVariable::GammaRandomVariable(1.0 + alpha_, beta_).value()* pow (u, 1.0 / alpha_);` que será reemplazada por la siguiente: `-return GammaRandomVariable(1.0 + alpha , beta),value() * pow(u, 1.0/alpha);`

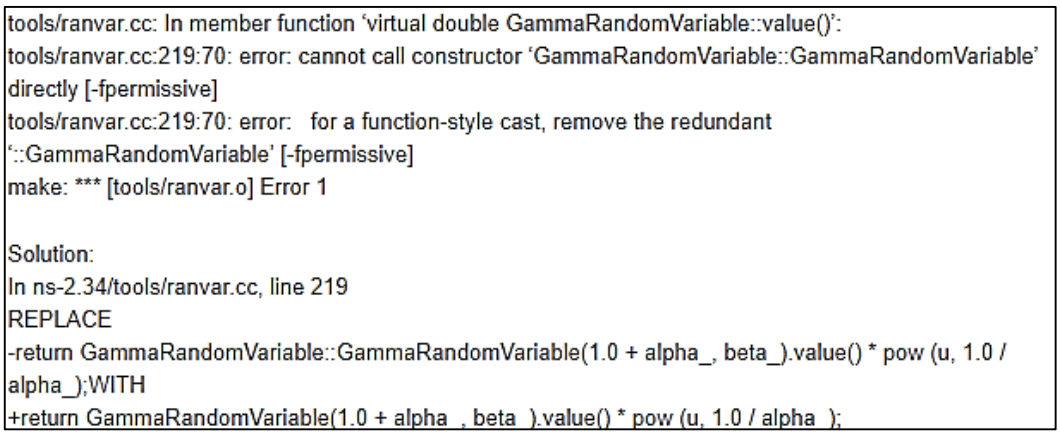

Figura 32. Error #2 instalación NS2.
Fuente [22].

- El tercer error tiene que ver con la capa *"mac"* dentro de NS2 que hace referencia al protocolo 802.11p el cual, para la versión utilizada del software, esta referenciada como *"802_11Ext"* (Ver Figura 33).

El procedimiento para solucionar la falla presente, es modificar en la línea 65 del archivo de configuración *"mac-802_Ext.h"* de la carpeta *"mac"* por la siguente línea de código: `+#include<cstddef>`

```
In file included from mac/mac-802_11Ext.cc:66:0:
mac/mac-802_11Ext.h: In member function 'u_int32_t PHY_MIBExt::getHdrLen11()':
mac/mac-802_11Ext.h:175:19: error: expected primary-expression before 'struct'
mac/mac-802_11Ext.h:175:41: error: 'dh_body' was not declared in this scope
mac/mac-802_11Ext.h:175:51: error: 'offsetof' was not declared in this scope
mac/mac-802_11Ext.h:177:3: warning: control reaches end of non-void function [-Wreturn-type]
make: *** [mac/mac-802_11Ext.o] Error 1
Ns make failed!

Solution:
In mac/mac-802_Ext.h, line 65

+#include<cstddef>
```

Figura 33. Error #3 instalación NS2.
Fuente [22].

- El cuarto error y su oportuna solución estará en la siguiente captura de pantalla, tomada de un fragmento de la página de errores del soporte web para la instalación de NS2 (Ver Figura 34).

```
mobile/nakagami.cc: In member function 'virtual double Nakagami::Pr(PacketStamp*, PacketStamp*,
WirelessPhy*)':
mobile/nakagami.cc:183:73: error: cannot call constructor
'ErlangRandomVariable::ErlangRandomVariable' directly [-fpermissive]
mobile/nakagami.cc:183:73: error:   for a function-style cast, remove the redundant
'::ErlangRandomVariable' [-fpermissive]
mobile/nakagami.cc:185:67: error: cannot call constructor
'GammaRandomVariable::GammaRandomVariable' directly [-fpermissive]
mobile/nakagami.cc:185:67: error:   for a function-style cast, remove the redundant
'::GammaRandomVariable' [-fpermissive]
make: *** [mobile/nakagami.o] Error 1

Solution:
In ns-2.34/mobile/nakagami.cc ,

if (int_m == m) {
resultPower = ErlangRandomVariable::ErlangRandomVariable(Pr/m, int_m).value();
} else {
resultPower = GammaRandomVariable::GammaRandomVariable(m, Pr/m).value();
}
return resultPower;  }
Replace the above with :  if (int_m == m) {
resultPower = ErlangRandomVariable(Pr/m, int_m).value();
} else {
resultPower = GammaRandomVariable(m, Pr/m).value();
}
return resultPower;
}
```

Figura 34. Error #4 instalación NS2.
Fuente [22].

b) Después de haber solucionado cada uno de los errores e instalado "*NS2"*, se debe reiniciar el PC. Para comprobar si quedó bien el procedimiento, se puede ejecutar el comando `ns` en una ventana de terminal

Una vez instalado y comprobado el software NS2, se realizó la descarga del paquete de instalación del protocolo 802.11p en "*Ns2*". Para este procedimiento, se documentaron los pasos claves para la descarga, instalación y configuraciones definidas con el objetivo de poder simular las redes teniendo la configuración precisa del protocolo.

2.5.1 Pasos para instalar 802.11P en Ns2

a. Primero se descargó el paquete con el contenido 802.11p para "*NS2*", en este caso se hizo la descarga directamente desde el portal web *"interlab"*. A continuación, se deja el link de descarga.

```
http://www.gel.usherbrooke.ca/interlab/index.php?page=NS2
```

b. El archivo de descarga, se descomprime y se extrae para poder visualizar y utilizar los archivos internos (Ver Figura 35).

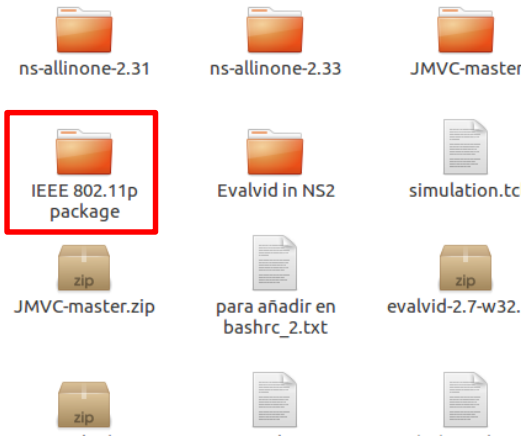

Figura 35. Carpeta descargada IEEE 802.11P package.
Fuente: el Autor.

c. Luego de tener la carpeta, internamente se seleccionaron todos los archivos que contiene y se copiaron en la carpeta mac del directorio ns-2.33 (Ver Figura 36, Figura 37 y Figura 38).

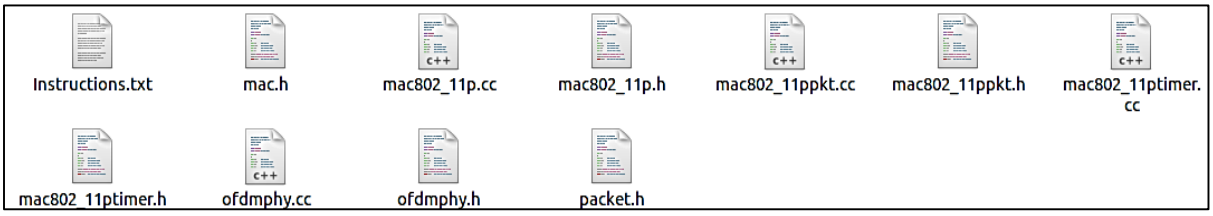

Figura 36. Contenido carpeta IEEE 802.11P package.
Fuente: el Autor.

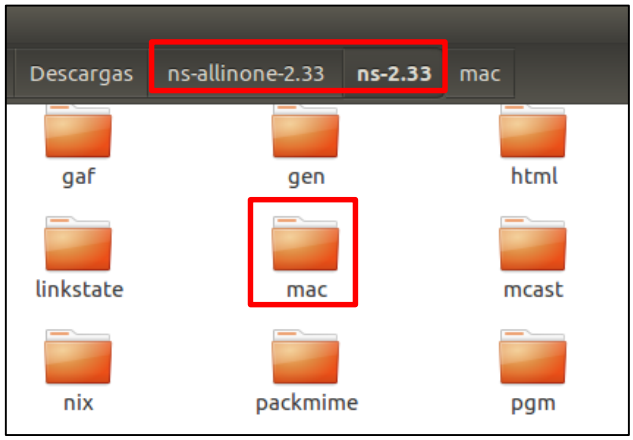

Figura 37. Carpeta mac en directorio ns-2.33.
Fuente: el Autor.

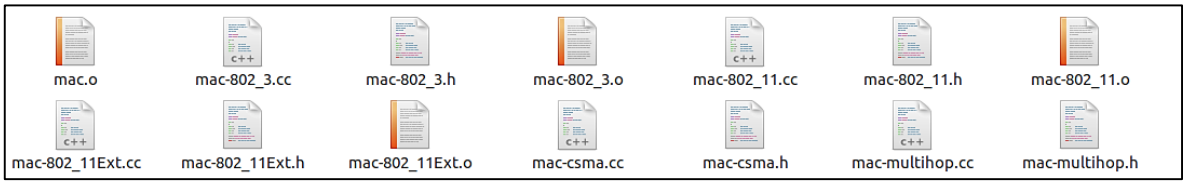

Figura 38. Carpeta mac con 802.11p en directorio ns-2.33.
Fuente: el Autor.

d. Luego de haber tenido la carpeta "*mac"* con los archivos de 802.11p, se agregaron las siguientes líneas en el archivo *"Makefile.in"* en el directorio ns-2.33 (Ver Figura 39) :

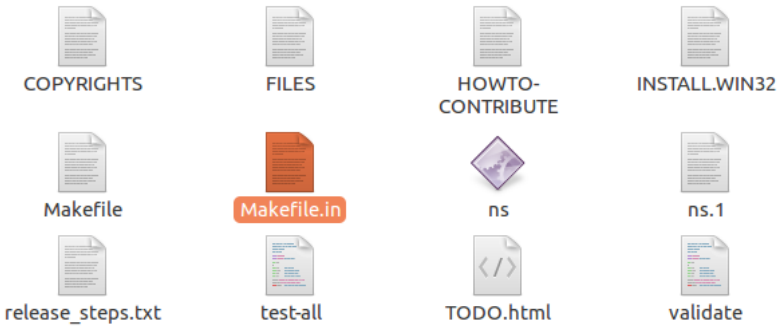

Figura 39. Carpeta "Makefile.in" en Directorio "ns-2.33".
Fuente: el Autor.

```
    mac/ofdmphy.o \
         mac/mac802_11ppkt.o \
         mac/mac802_11ptimer.o \
         mac/mac802_11p.o \
```

Nota: Agregue los archivos de objeto antes de la última línea (@ V_STLOBJ@) de la lista OBJ_CC (Ver Figura 40).

```
wpan/p802_15_4trace.o wpan/p802_15_4transac.o \
mac/ofdmphy.o \
 mac/mac802_11ppkt.o \
 mac/mac802_11ptimer.o \
 mac/mac802_11p.o \
myevalvid/myudp.o \
myevalvid/myevalvid_sink.o \
myevalvid/myevalvid.o \
@V_STLOBJ@
```

Figura 40. Líneas modificadas en archivo Makefile.in del directorio ns-2.33.
Fuente: el Autor.

e. Se reemplazó el archivo "*mac.h"* y "*packet.h"* en la carpeta mac de la carpeta "*ns-2.33"* con la que se encuentra dentro de la carpeta "IEEE 802.11p package" (Ver Figura 41).

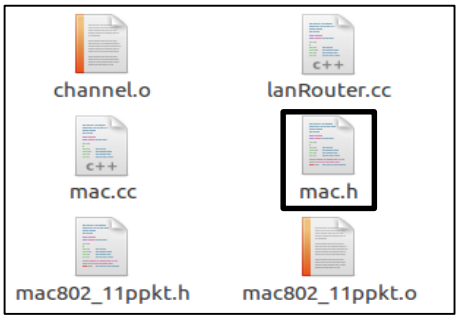

Figura 41. Archivo mac.h reemplazado.
Fuente: el Autor.

f. Se editó el archivo *"common / packet.h"* y se agregó un nuevo paquete tipo "*PT_WAVEBS"* en la enumeración "*packet_t"* de nombre "nombre_ [PT_WAVEBS] =" waveCtrl ";a la clase "*p_info"* (Ver Figura 42). La forma como se debe escribir correctamente dicha instrucción es: `name_[PT_waveBS] = "waveCtrl";`

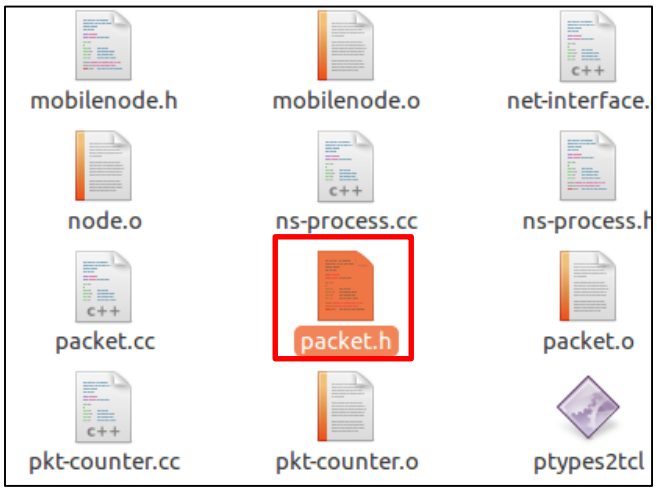

Figura 42. Archivo packet.h en directorio common.
Fuente: el Autor.

g. Se editó el archivo `tcl / lib / "ns-packet.tcl"` y se agregó `"802_11p"` a la lista definida en la lista de paquetes (sección del código llamada *"foreach prot"*).

h. Finalmente, es necesario recompilar el simulador para que la integración se realice efectivamente. Para recompilar el código fuente del simulador, es necesario abir una terminal desde la carpeta "*ns-2.31"* y se ejecutar el siguiente comando:

```
"./configure; make clean; make".
```

2.6 EVALUACIÓN DEL PROTOCOLO 802.11P

Para evaluar las características básicas del protocolo 802.11p se realizó una simulación de un escenario de red inalámbrica. Este escenario se utilizó con el protocolo 802.11 y 802.11p, con el fin de comprar el desempeño de ambos.

2.6.1 Descripción del escenario de simulación.

El escenario de red que está conformado por un nodo móvil (nodo 0) y cinco nodos inalámbricos estáticos (nodos 1,2,3,4,5). El nodo 0, está configurado como el nodo fuente y es el encargado de la transmisión de la información. El nodo 1, está configurado como el nodo destino, en este caso es quien recibe dicha información proporcionada por el nodo 0. La velocidad de transmisión es de 5Mbps, pero la capacidad del canal inalámbrico es de 6Mbps (una de los parámetros típicos del protocolo 802.11p, tal describió en la Tabla III). Los nodos están separados 250 m entre ellos, y a partir del segundo 10 de simulación, el nodo 0 empieza a moverse en dirección del nodo (Ver Figura 43).

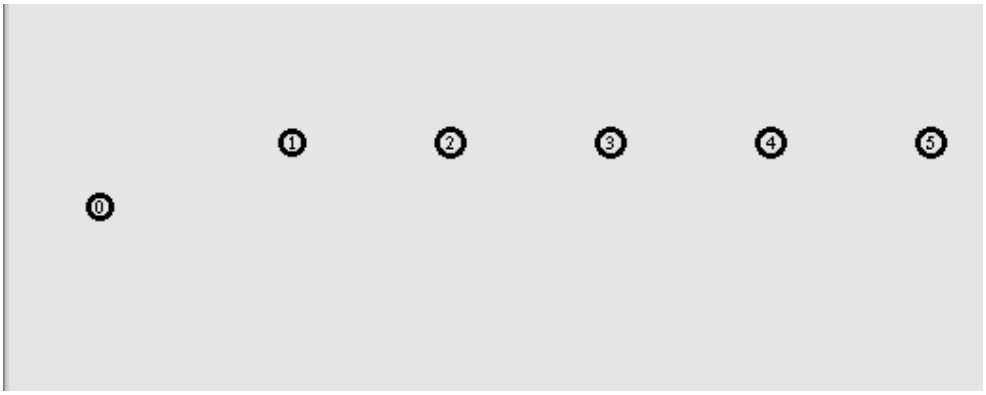

Figura 43. Captura de Pantalla del escenario de simulación para 802.11 y 802.11p
Fuente: el Autor

2.6.2 Simulación de los Protocolos 802.11 y 802.11p.

Las configuraciones de la red realizadas para la simulación del estándar 802.11, se pueden consultar en el Anexo 7.1. Cuando se realizó la simulación para el protocolo 802.11 (Ver Figura 44), se pudo observar que, a medida que el nodo 0 se mueve hacia el nodo 5, el número de saltos presentados en el tráfico entre el nodo 0 y el 1 se va incrementando. Por ejemplo, cuando inicia la simulación, el nodo 0 le puede transmitir directamente al nodo 1 sin

necesidad de nodos intermedios, ver Figura 45. Sin embargo, cuando el nodo 0 sale de la cobertura del nodo 1, el tráfico debe enrutarse a través de los otros nodos de la red. En la Figura 46, el nodo 0 está más cerca del nodo 4 y por lo tanto los paquetes deben ser retransmitidos a través del nodo 4, nodo 3, nodo 2, hasta el nodo 1.

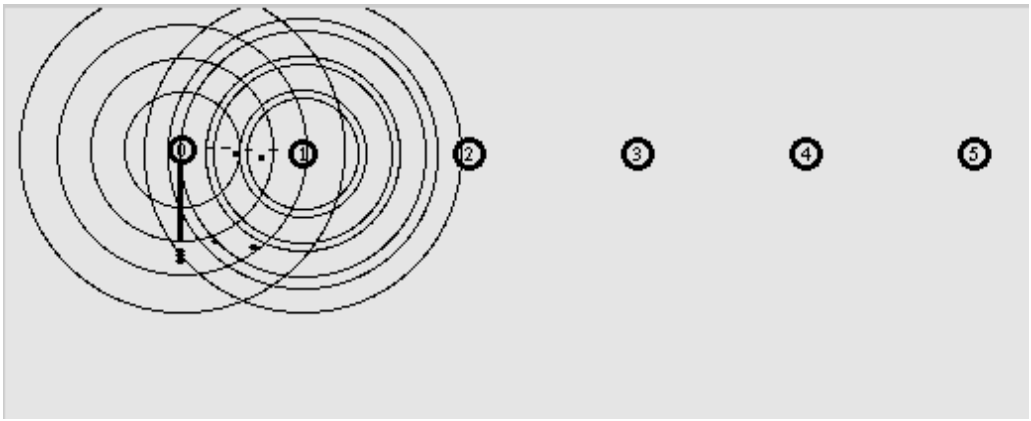

Figura 44. Escenario de Simulación (protocolo 802.11).
Fuente: el Autor

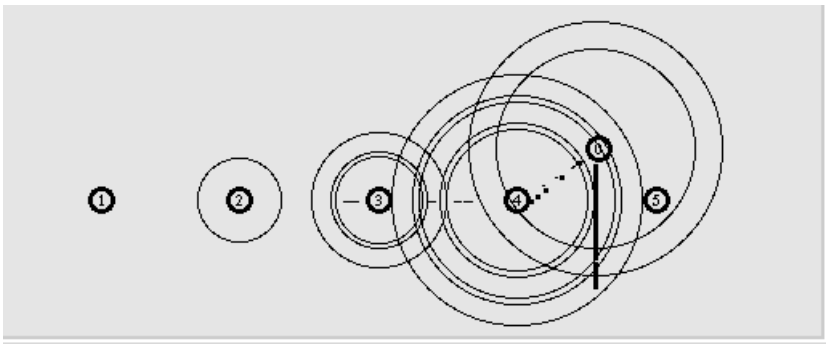

Figura 45. Número de saltos durante la transmisión entre el nodo (0) y nodo (1).
Fuente: el Autor

Este aumento de los saltos afecta el flujo de datos que puede recibir el nodo 1, tal como se muestra en la Figura 46. En dicha figura se puede observar que cuando la ruta está conformada por un solo salto, el nodo 1 puede recibir datos a una tasa de hasta 3.2 Mbps. Sin embargo, a medida que aumentan los saltos que dar los paquetes para llegar al destino, el flujo de datos disminuye. Por el contrario, con el protocolo 802.11p el flujo de datos se mantiene ya que puede establecer enlaces con mayor cobertura. Esto permite que el flujo de datos se mantenga constante ya que siempre el nodo 0 puede transmitirle directamente al nodo 1 (ver Figura 47).

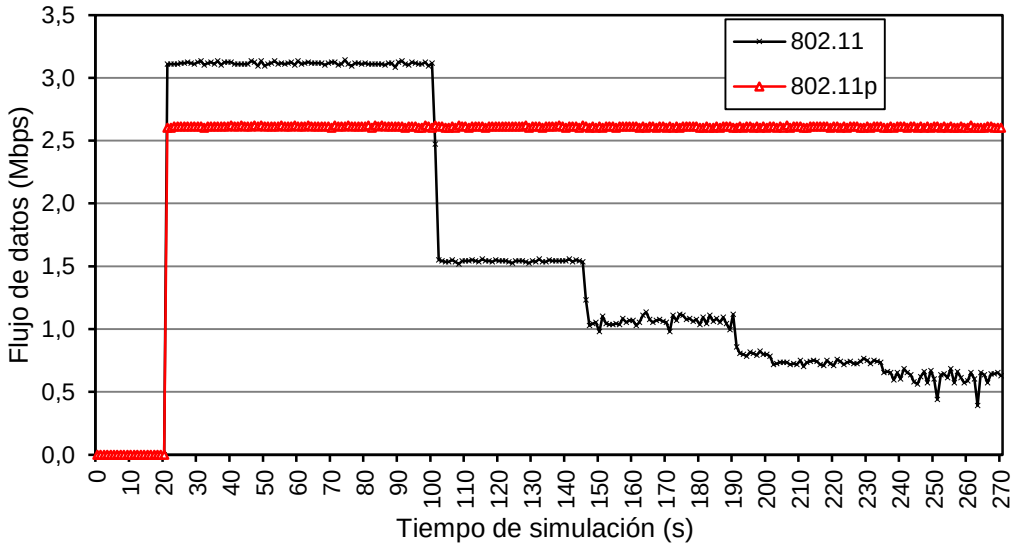

Figura 46. Gráfica del flujo de datos para 802.11.
Fuente: el Autor

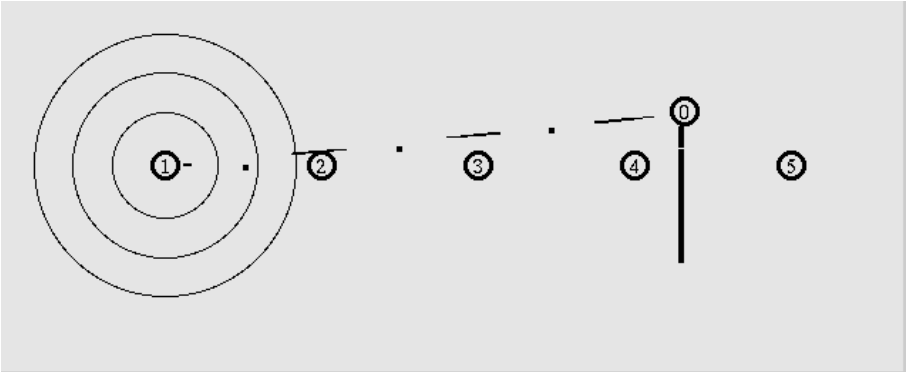

Figura 47. Escenario de Simulación con protocolo 802.11p.
Fuente: el Autor

El impacto del comportamiento descrito anteriormente tiene una consecuencia importante si se analizan las pérdidas de paquetes. Tal como se observa en Figura 48, para el protocolo 802.11, a medida que aumentan los saltos, también aumenta el porcentaje de paquetes que se pierden a lo largo de la ruta. Aunque para el estándar 802.11p la pérdida de paquetes no varía, ya que se obtuvo unas pérdidas del 50%.

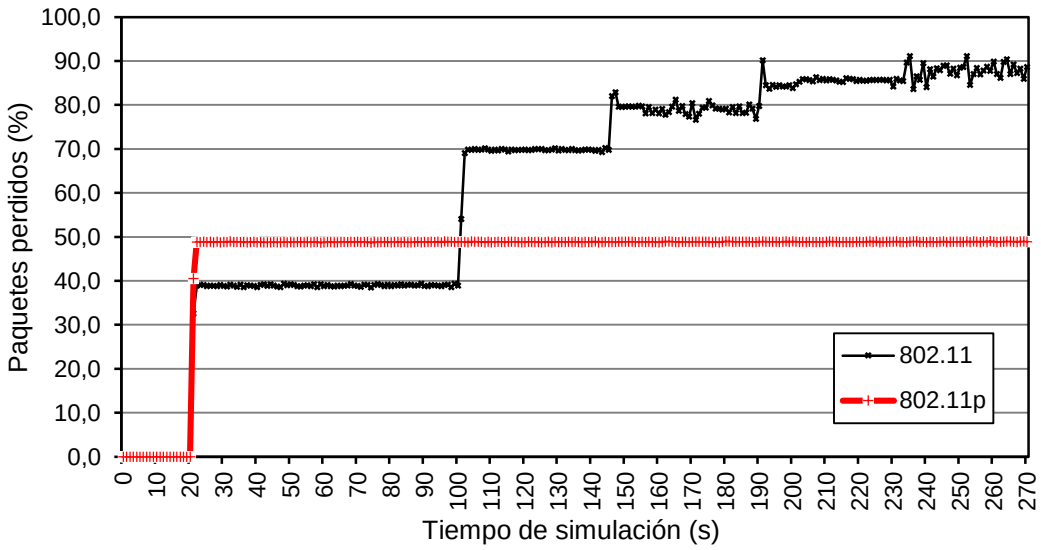

Figura 48. Porcentaje de paquetes perdidos durante la simulación.
Fuente: el Autor

Y a manera de comprobación también se obtuvo directamente de la simulación, el porcentaje de paquetes recibidos por el nodo 1 (ver Figura 49). Como se observa, el comportamiento de estas curvas es coherente con las del porcentaje de paquetes perdidos.

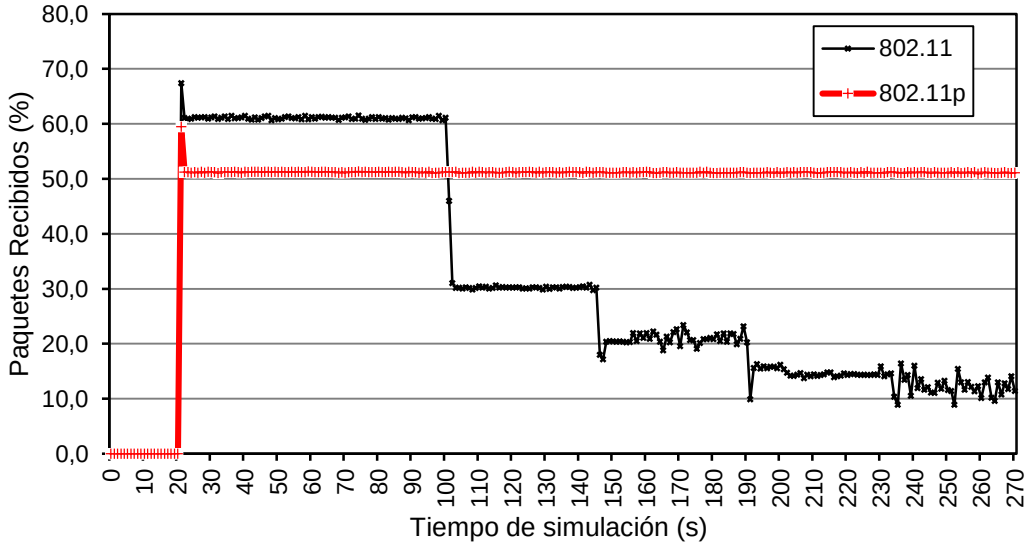

Figura 49. Porcentaje de paquetes recibidos durante la simulación.
Fuente: el Autor

Finalmente, es importante resaltar que esta evaluación nos permite establecer que para hacer una transmisión de video sobre el estándar 802.11p, es necesario reducir el bit rate del video de manera adecuada de tal manera que esta no se aproxime a la capacidad de transmisión del canal inalámbrico. Por ejemplo, si se configura la data rate del estándar

802.11p en 6Mbps, tal como se hizo en la simulación que se presentó en este capítulo, el video debe estar codificado de tal manera que consuma menos de 5Mbps, y así evitar la alta pérdida de paquetes.

La configuración completa del archivo de simulación se puede consultar en los Anexos 7.1 y 7.2, los cuales se resumen en los siguientes diagramas de flujo (Ver Figura 50 y Figura 51):

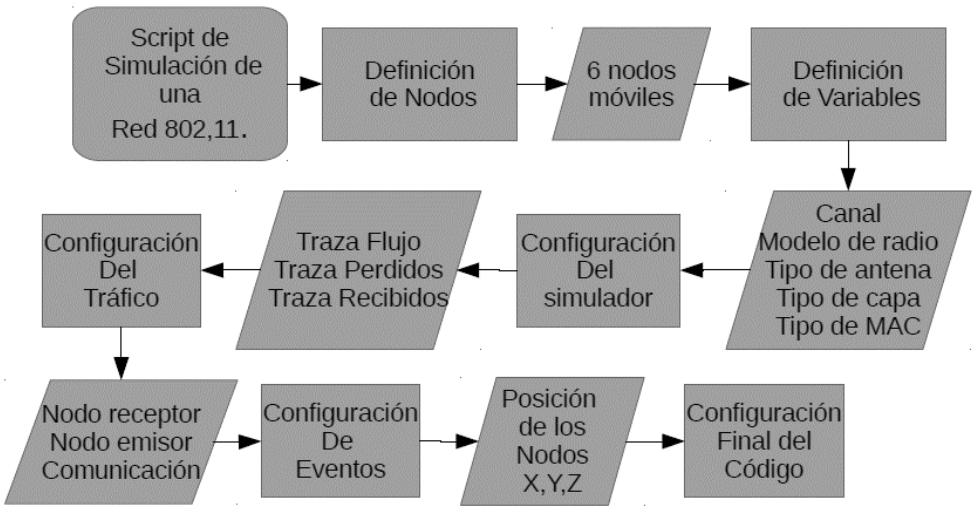

Figura 50. Resumen Anexo 7.1
Fuente: el Autor

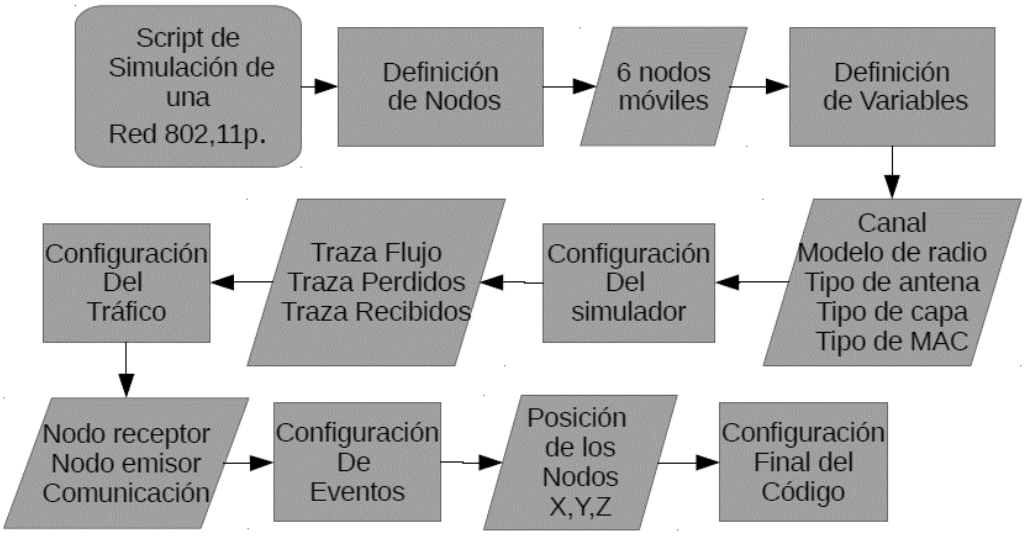

Figura 51. Resumen Anexo 7.2
Fuente: el Autor

CAPÍTULO 3

3 TRANSMISION DE VIDEO SOBRE REDES

3.1 FUNDAMENTOS DE TRANSMISIÓN DE VIDEO SOBRE REDES

En esta oportunidad se hablará principalmente de la problemática de la transmisión de la información multimedia desde un servidor hacia diferentes clientes, usando como red de transporte Internet, que como se ha venido trabajando con el tiempo, se sabe que está basada en la arquitectura TCP/IP.

Existen alternativas para transmitir la información multimedia basadas en diferentes técnicas y con las cuales se daría solución a esta problemática, con base a esto, se mencionarán tres, de las cuales cada una servirá de fundamento para el desarrollo del procedimiento final del proyecto.

3.2 ALTERNATIVAS PARA LA TRANSMISIÓN DE INFORMACIÓN MULTIMEDIA.

La primera alternativa hace referencia a la hora de transmitir la información multimedia con el objetivo de manejar esta como el resto de la información, usando así aplicaciones y servicios estándar de Internet como por ejemplo *"ftp" y "http"*. Una de las ventajas principales de estos servicios, es el hecho de que permiten una vista del medio justo cuando éste se haya descargado por completo [23].

La Segunda alternativa, característicamente indica que la información se descarga usando el máximo ancho de banda dispuesto para "Cliente y Servidor". No existe ningún control para evitar interrupciones o cortes en la reproducción, en resumen, el medio se va almacenando en disco conforme se descarga. Si el ancho de banda es más reducido que el necesario para la reproducción, la información se reproduce "a saltos", ya que se va reproduciendo conforme llega [23].

La tercera alternativa, hace hincapié al tema del *"Streaming"* (Transmisión), en el que se utilizan protocolos para la transmisión de información multimedia en tiempo real (tal y como el protocolo estándar RTP) con un control de sesión dinámico (como el que permite RTSP). El ancho de banda utilizado para este caso en particular, es el estrictamente necesario para transmitir y visualizar el medio en tiempo real. Otra característica importante es que no se produce una descarga completa del medio, sino que conforme se descarga se va descartando una vez ha sido utilizado para la reproducción [23].

3.2.1 Transmisión de información multimedia.

La transmisión de información multimedia para múltiples aplicaciones requiere prestaciones de tiempo real. Estas prestaciones implican la necesidad de recibir a tiempo la información para que ésta pueda ser útil para la capa de aplicación. La recuperación de pérdidas a través de la retransmisión de los datos perdidos es una alternativa que puede introducir retardos inaceptables. Por esta razón, muchos Autores proponen transmitir la información mediante capas de transporte sin función de recuperación de errores y, en su

lugar, se usan códigos de error FEC y técnicas similares para maximizar la recuperación de datos en el receptor [24].

El proceso de transmisión consiste en la entrega de uno o varios medios multiplexados hacia un cliente en tiempo real, y usando una red con un determinado ancho de banda considerable según sea la necesidad. En el proceso de transmisión se debe considerar lo siguiente:

No hay ningún fichero que se descarga al ordenador del cliente, sino que el medio se reproduce conforme se está recibiendo.

A su vez el medio se recibe a la velocidad adecuada para su reproducción.

Esto contrasta con las descargas progresivas, en las que el fichero sí queda descargado en disco y además se recibe a la mayor velocidad posible, con el fin de terminar el proceso de descarga lo antes posible [25]

3.2.2 Proceso de transmisión estándar de audio y video sincronizado

Las peticiones de servicio por parte de los clientes se pueden manejar utilizando el protocolo RTSP (RealTime Streaming Protocol) en castellano (Protocolo de Transmisión en Tiempo Real).

El “stream” que contiene típicamente audio y vídeo sincronizados, se pueden transportar usando el protocolo estándar RTP (Real-Time Transport Protocol).

RTP, es un protocolo de transporte que permite la transmisión de información multimedia en tiempo real sobre cualquier tipo de red (aunque su uso más habitual es sobre redes usando el protocolo UDP) [24].

Como se muestra a continuación en la Figura 52, el reproductor y el servidor intercambian una serie de mensajes RTSP. El reproductor envía una solicitud RTSP SETUP y el servidor responde con un mensaje RTSP OK. El reproductor envía una solicitud RTSP de reproducción PLA de, digamos, audio de baja fidelidad y el servidor le responde con un mensaje RTSP OK. En este punto, el servidor de flujos coloca el audio de baja fidelidad en su propio canal en banda. A continuación, el reproductor multimedia envía una solicitud RTSP de pausa con el mensaje PAUSE y el servidor responde de nuevo con un mensaje RTSP OK. Cuando el usuario ha terminado, el reproductor multimedia envía una solicitud RTSP TEARDOWN para terminar la sesión y el servidor confirma con un mensaje de respuesta RTSP OK [24].

3.2.3 Compresión de Audio y Video

Para poder transmitir a una red de computadoras el audio y el video, se debe tener en cuenta que previo a esto, debe realizarse un proceso de digitalización y compresión. La digitalización, se lleva a cabo ya que las redes de computadoras transmiten en bits, de modo que toda la información transmitida tiene que representarse como una secuencia de bits. La compresión es importante porque el audio y el vídeo descomprimidos consumen enormes cantidades de almacenamiento y de ancho de banda: eliminando las redundancias inherentes mediante la compresión de las señales de audio y vídeo digitalizadas puede reducirse la cantidad de datos que habrá que almacenar y transmitir en varios órdenes de magnitud.

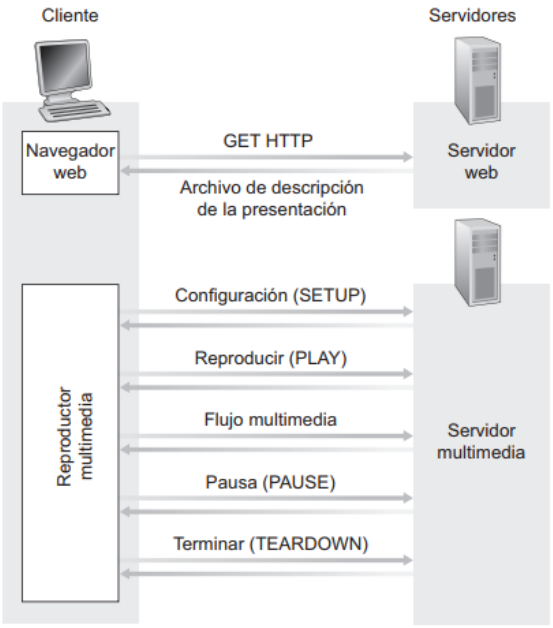

Figura 52. Interacción entre un cliente y un servidor utilizando RSTP
Fuente [25]

Por ejemplo, una única imagen de 1024 píxeles, con cada píxel codificado en RGB con 24 bits (8 bits por cada color rojo, verde y azul) requiere 3 Mbytes de almacenamiento sin compresión. Se tardarían siete minutos en enviar esta imagen a través de un enlace de 64 kbps. Si la imagen se comprime con una relación de comprensión modesta de 10:1, el requisito de almacenamiento se reduce a 300 kbytes y el tiempo de transmisión se reduce también en un factor de 10:1 [25].

3.2.4 Acceso al audio y al video a través de un servidor web

En muchas implementaciones de transmisión de flujos de audio/vídeo sobre HTTP, la funcionalidad del lado del cliente se divide en dos partes.

La tarea del navegador es solicitar un archivo que proporciona información (por ejemplo, un URL y un tipo de codificación, de modo que pueda ser identificado el reproductor multimedia apropiado) acerca del archivo multimedia que va a ser transmitido mediante HTTP. Este archivo se pasa entonces desde el navegador al reproductor multimedia, cuyo trabajo consiste en contactar al servidor HTTP, el cual a continuación envía el archivo multimedia al reproductor vía HTTP. Estos pasos se ilustran en la Figura 53.

El usuario hace clic en el hipervínculo correspondiente a un archivo de audio/vídeo. El hipervínculo no apunta directamente a dicho archivo, sino a un metarchivo.

El metarchivo contiene el URL al archivo de audio/vídeo real. El mensaje de respuesta HTTP que encapsula el metarchivo incluye una línea de cabecera con el tipo de contenido que indica una aplicación de audio/vídeo específica.

El navegador del cliente examina la línea de cabecera correspondiente al tipo de contenido del mensaje de respuesta, ejecuta el reproductor multimedia asociado y pasa el cuerpo del mensaje de respuesta (es decir, el metarchivo) al reproductor.

El reproductor multimedia establece una conexión TCP directamente con el servidor HTTP. El reproductor multimedia envía un mensaje de respuesta HTTP para el archivo de audio/vídeo a través de la conexión TCP. El archivo de audio/vídeo se envía dentro de un mensaje de respuesta HTTP hacia el reproductor, el cual reproduce el archivo.

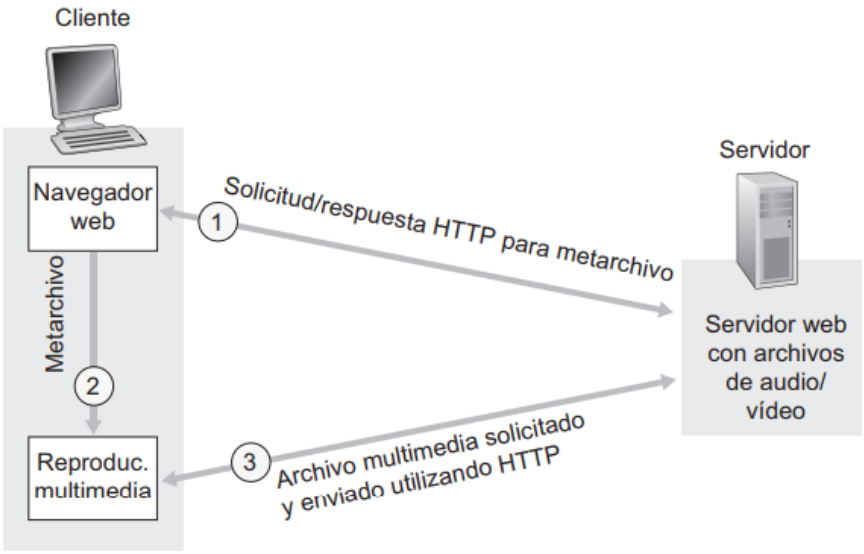

Figura 53. Flujo de audio y video desde el servidor al reproductor multimedia.
Fuente [25]

3.2.5 Tipos de transmisión (Streaming)

El proceso de “Streaming” se puede dividir en dos categorías, en función de cómo se obtiene la información a difundir: “Streaming” en directo o bajo demanda.

3.2.6 El streaming en directo

Es aquel tipo de transmisión en la que se transmiten eventos que están sucediendo justo en el momento de la difusión. Por ejemplo: la transmisión de conciertos o eventos deportivos, la transmisión de radio y televisión por Internet tiene estas características, aunque en ocasiones parte de la información que se difunde no hace parte de un evento en directo. Por ejemplo, cuando se transmite un programa que ha sido grabado previamente, pero que se va a difundir en un momento determinado.

En este tipo de transmisión se emplea el término difusión (broadcast) porque realmente se está transmitiendo “en vivo” a todos los clientes la misma información, que no es más que el evento que se está produciendo en ese momento.

3.2.7 Flujo multimedia bajo demanda

La transmisión del medio empieza con el inicio del evento a ser reproducido para cada uno de los clientes. El medio a transmitir puede estar ya preparado desde el comienzo del proceso en un fichero comprimido. En este caso no representa una ventaja adicional el disponer de posibilidad del realizar “multicast” en la red, ya que cada cliente recibe una parte distinta del “stream” y por lo tanto un paquete de datos diferente [23].

3.2.8 Flujo multimedia adaptativo

Los métodos descritos anteriormente se conocen como las técnicas tradicionales de transmisión de video. En dichas técnicas, el video es enviado como un flujo de paquetes a usando un protocolo encargado de establecer una sesión entre el proveedor del video y el cliente. Sin embargo, este esquema tradicional de transmisión multimedia carece de la flexibilidad y la adaptabilidad necesaria ya que el video es transmitido al usuario final como un flujo de bits constante través de una conexión que en la mayoría de los casos es fluctuante. Por lo tanto, si el canal de transmisión experimenta congestión, la calidad del video puede degradarse significativamente [26].

La transmisión adaptativa de video es una técnica que estima el ancho de banda disponible de la red, con el fin de adaptar el flujo de video a dichas condiciones. Información más detallada sobre algunas técnicas que podrían ser utilizadas para implementar este tipo transmisión de video, puede ser consultada en las referencias [27] y [28].

Cualquier método de transmisión de video, requiere ser evaluada antes entrar en operación en una red real, por lo tanto, es importante establecer cuáles son las alternativas para evaluar la transmisión de video sobre redes de datos. A continuación, se hace una breve descripción de una técnica que permite evaluar tanto los protocolos como las codificaciones, involucrados en la transmisión de video sobre redes de comunicaciones, especialmente sobre redes de paquetes.

3.3 EVALUACIÓN DE LA TRANSMISIÓN DE VIDEO POR REDES DE COMUNICACIONES

Una de las técnicas más utilizadas para medir la calidad de una transmisión de contenido multimedia es por medio de plataformas de simulación. La principal ventaja se este método es la significativa reducción de costos y tiempo de implementación, en comparación con la implementación en equipos reales. La principal plataforma de simulación de este tipo de transmisiones es la conformada por EVALVID [29] y NS-2 (NetworK Simulator II).

3.3.1 Evalvid

Evalvid es una plataforma y conjunto de herramientas para la evaluación de la calidad del video transmitido a través de una red de comunicación real o simulada. Está dirigido a investigadores que desean evaluar sus diseños o configuraciones de red en términos de calidad de video percibida por el usuario.

En la Figura 54 , se muestra un ejemplo general del procedimiento de procesamiento de un video con Evalvid.

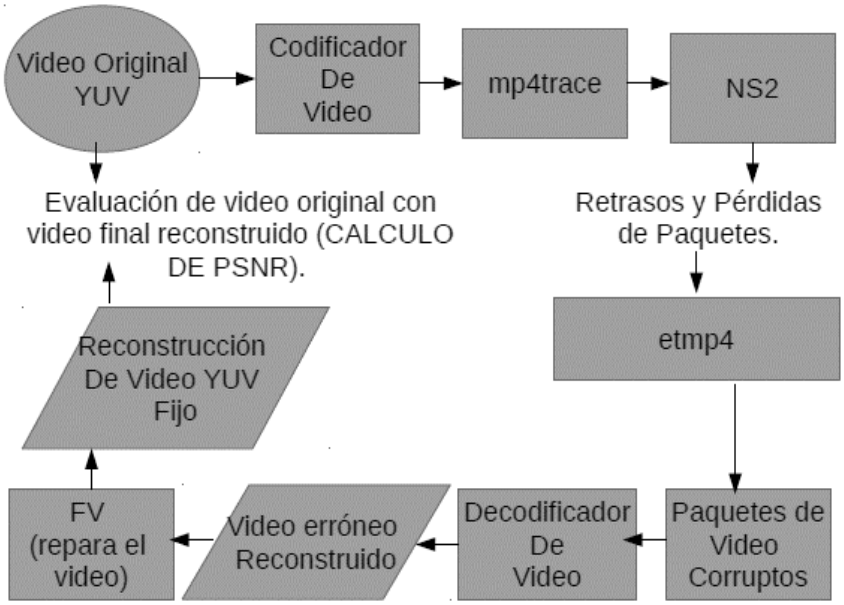

Figura 54. Metodología para la codificación y transmisión de un video usando Evalvid y NS2.
Fuente: el Autor

El diagrama de la Figura 55, contiene los pasos más detallados para la transmisión del video y la participación de Evalvid en el proceso. Como se puede observar, se inicia con un video formato YUV (RAW, sin codificación). Este video primero se pasa por un codificador, se generan un archivo descriptor del video (también conocida como traza de video). Este archivo o traza contiene la información de cómo se debe empaquetar y enviar el video (*"Videotrace.tr"*). Esta traza se adecua al formato de NS2 por medio del script "maketrace.tcl".

Finalmente, la traza *"trace_video_ns2.tr"* se pasa al simulador para que este genere los paquetes y los envié de acuerdo a lo indicado en la traza. Durante la simulación se generan unos archivos que contienen los resultados de la misma. Con el contenido del archivo de salida *"eventos.tr"*, se pueden calcular parámetros como el flujo de datos, los paquetes perdidos, el retardo, entre otros parámetros.

Durante la simulación también se genera la traza *"traza_recibida.tr"* que contiene la información de los paquetes de video recibidos. Esta traza es enviada al programa *"etmp4t"* para construir un video a partir de la información contenida en dicha traza.

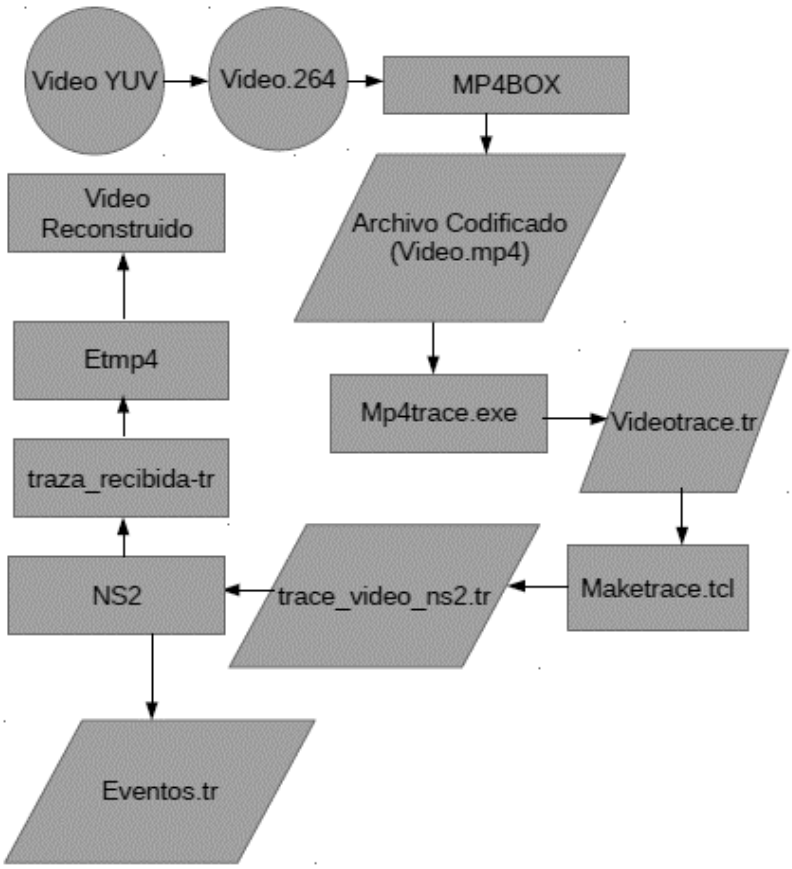

Figura 55. Estructura de Evalvid.
Fuente: el Autor

3.4 INTEGRACIÓN DE EVALVID EN NETWORK SIMULATOR 2 (NS2)

3.4.1 Pasos para instalar Evalvid en NS2

Inicialmente se descarga la carpeta de archivos de "*evalvid"* para *"ns allinone 2.33"*, llamada "*myevalvid",* la cual contiene los archivos fuente y compiladores que serán luego instalados y compilados por medio de NS2 (ver Figura 56) [30].

Figura 56. Carpeta "myevalvid" descargada y descomprimida.
Fuente: el Autor.

Teniendo la carpeta descargada, se deben copiar los archivos internos que contiene ésta, y se ingresa dentro de la ruta de la carpeta del software ns2.33, la cual tiene la ***ruta*** `home/carpeta personal/Descargas/ns-allinone-2.33/ns-2.33` y se crea una carpeta con el mismo nombre ***"myevalvid"***, donde serán almacenados los archivos que permitirán establecer una nueva fuente de tráfico de video (Ver Figura 57).

Figura 57. Contenido de la carpeta "myevalvid".
Fuente: el Autor.

3.4.2 Configuraciones para instalación de myevalvid en NS2

Para la correcta integración de *evalvid* en *ns2* se deben modificar algunos archivos del código fuente del simulador.

3.4.2.1 ***Modificación del archivo "packet.h":*** se introduce un campo "frametype_" y "sendtime_" en el encabezado "hdr_cmn". El campo "frametype_" es para indicar a qué tipo de cuadro pertenece el paquete. Para llevar a cabo este paso, se debe modificar el archivo "packet.h" en la carpeta "common" dentro de "ns-2.33" como se muestra a continuación las líneas escritas en negrilla dentro del archivo.

```
struct hdr_cmn {
  enum dir_t { DOWN= -1, NONE= 0, UP= 1 };
  packet_t ptype_;      // packet type (see above)
  int     size_;               // simulated packet size
  int     uid_;         // unique id
  int     error_;              // error flag
  int     errbitcnt_;     // # of corrupted bits jahn
  int     fecsize_;
  double      ts_;           // timestamp: for q-delay measurement
  int     iface_;              // receiving interface (label)
  dir_t direction_;        // direction: 0=none, 1=up, -1=down
  // source routing
        char src_rt_valid;
  double ts_arr_; // Required by Marker of JOBS
 //add the following three lines
  int frametype_;
  double sendtime_;
  unsigned long int frame_pkt_id_;
```

3.4.2.2 ***Modificación del archivo "agent.h":*** se modifica el archivo "agent.h" en la carpeta "common" dentro de "ns-2.33", como se muestra a continuación las líneas en negrilla:

```
class Agent : public Connector {
public:
Agent(packet_t pktType);
virtual ~Agent();
void recv(Packet*, Handler*);
. . . . . .
inline packet_t get_pkttype() { return type_; }
```

```
// add the following two lines
inline void set_frametype(int type) { frametype_ = type; }
inline void set_prio(int prio) { prio_ = prio; }
protected:
int command(int argc, const char*const* argv);
 . . . . . .
int defttl_;                // default ttl for outgoing pkts
// add the following line
int frametype_;              // frame type for MPEG video transmission
 . . . . . .
private:
void flushAVar(TracedVar *v);
};
```

3.4.2.3 Modificación del archivo "agent.cc": en la carpeta "common" dentro de "ns-2.33", como se muestra a continuación las líneas en negrilla:

```
Agent::Agent(packet_t pkttype) :
  size_(0), type_(pkttype), frametype_(0),
  channel_(0), traceName_(NULL),
  oldValueList_(NULL), app_(0), et_(0)
{
}

. . . . . .
Agent::initpkt(Packet* p) const
{
  hdr_cmn* ch = hdr_cmn::access(p);
  ch->uid() = uidcnt_++;
  ch->ptype() = type_;
  ch->size() = size_;
  ch->timestamp() = Scheduler::instance().clock();
  ch->iface() = UNKN_IFACE.value(); // from packet.h (agent is local)
  ch->direction() = hdr_cmn::NONE;

  ch->error() = 0;        /* pkt not corrupt to start with */
 // add the following line
  ch->frametype_= frametype_;
```

3.4.2.4 Modificación del archivo "ns-default.tcl": en la ruta `ns-allinone-2.33/ns-2.33/tcl/lib/ns-default.tcl` como se muestra a continuación las líneas:

```
    Agent/myUDP set packetSize_ 1000
    Tracefile set debug_ 0
```

3.4.2.5 *Modificación del archivo "Makefile.in":* en la ruta `ns-allinone-2.33/ns-2.33/Makefile.in`, como se muestra a continuación las líneas.

Poner `myevalvid/myudp.o, myevalvid/myevalvid_sink.o y myevalvid/myevalvid.` en la lista **OBJ_CC**.

3.4.2.6 *Recompilar "ns-2.33":* se recompila NS2 con los siguientes comandos, para que queden las configuraciones anteriores grabadas y se instale Evalvid:

```
./configure; make clean; make
```

CAPÍTULO 4

4 EVALUACIÓN DE LA TRANSMISIÓN DE VIDEO MVC SOBRE REDES 802.11P

A continuación, se escribe paso a paso de la metodología llevada a cabo para la ejecución de la simulación final, con la cual se evidencia la transmisión de video MVC por medio de una red simulada, configurada con los parámetros estandarizados para 802.11p.

A nivel general los pasos para realizar una simulación son los descritos en el diagrama de flujo de la Figura 58.

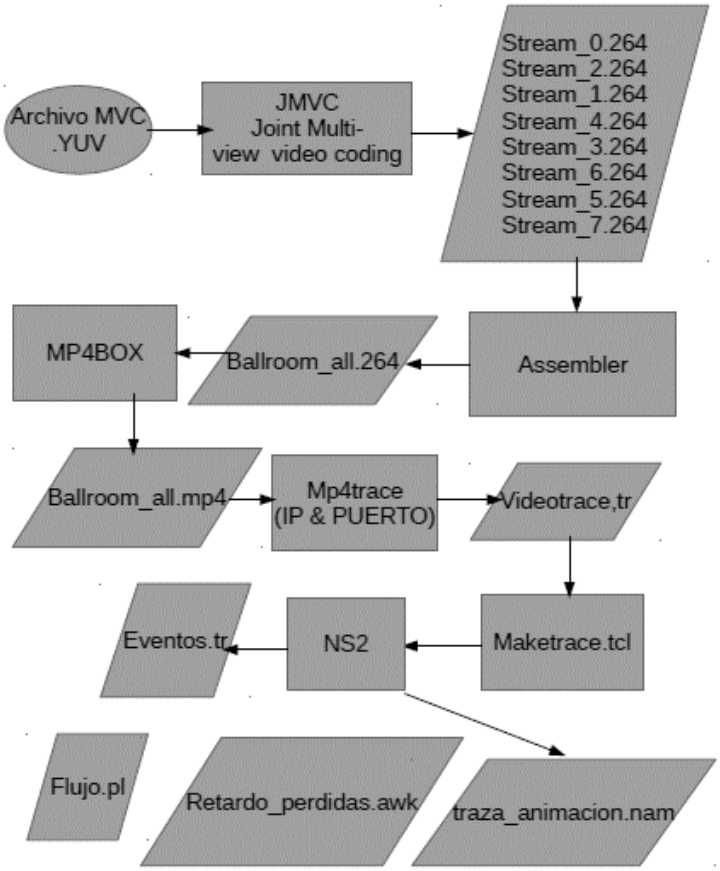

Figura 58. Diagrama de flujo de los pasos de la simulación.
Fuente: el Autor.

4.1 PRE-PROCESAMIENTO

4.1.1 Codificación y transmisión del video MVC sobre NS2

Una vez ya instalado el software de codificación *JMVC*, y habiendo descargado las vistas del video *MVC* en formato yuv (Ver Figura 59), se procede a crear el archivo de codificación,

el cual será nombrado ***"ballroomConfig.cfg",*** la forma correcta de hacerlo, estará descrita a continuación, los pasos a seguir se ejecutaron con ayuda del manual de usuario del software proporcionado por los desarrolladores [31].

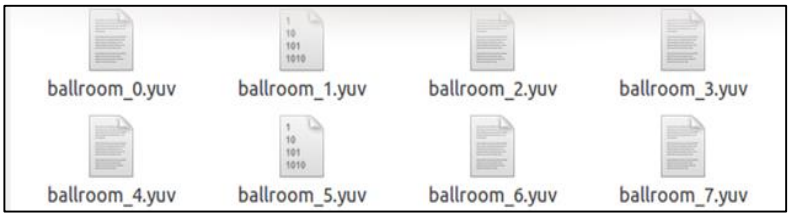

Figura 59. visualización de las 8 vistas MVC en formato YUV.
Fuente: el Autor.

En la sección de anexos, se dejará el formato del contenido del archivo "*ballroomConfig.cfg"* y las configuraciones realizadas para el proyecto, al mismo tiempo, estarán las configuraciones por defecto estipuladas por el *JMVC* software. Los parámetros en negrilla, son aquellos que fueron modificados para llevar a cabo la simulación propuesta en el proyecto, el resto serán los que vienen ya configurados (Ver Anexo 7.3).

Una vez codificado el anterior archivo (.cfg), se proceden a codificar cada una de las ocho vistas del video original, siguiendo para cada una de ellas el orden sugerido por el manual de usuario [31].

El orden especificado para este procedimiento es el siguiente: vistas (0-2-1-4-3-6-5-7).

Los códigos implementados con cada una de las vistas son los siguientes:

```
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 0
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 2
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 1
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 4
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 3
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 6
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 5
./H264AVCEncoderLibTestStatic-vf/home/juancho/Documentos/ballroomConfig.cfg 7
```

Luego de ser codificadas las vistas, para cada una de ellas se crea un archivo con el formato *H.264*, el cual tendrá el nombre que se configuro anteriormente en el archivo (.cfg), exactamente en el output file cuyo nombre configurado fue ***"stream"*** (Ver Figura 60).

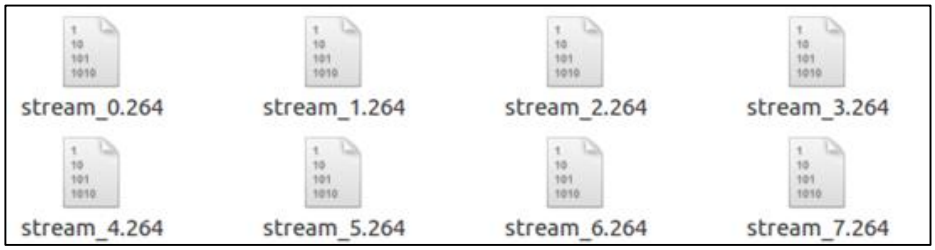

Figura 60. Archivos generados para cada vista debido a codificación.
Fuente: el Autor.

El siguiente paso, es crear el archivo ensamblador, con el que se unen las vistas para crear una sola, dicho archivo se llamará ***"ballroom_all.cfg"*** y el resultado de la unión de las vistas, se llamará ***"ballroom_all.264".***

"ballroom_all.cfg"

```
    #==========Assembler: View Encode ==================
    OutputFile          /home/juancho/Documentos/JMVC-
master/bin/ballroom_all.264
    NumberOfViews       8
    InputFile0          /home/juancho/Documentos/JMVC-master/bin/stream_0.264
    InputFile1          /home/juancho/Documentos/JMVC-master/bin/stream_2.264
    InputFile2          /home/juancho/Documentos/JMVC-master/bin/stream_1.264
    InputFile3          /home/juancho/Documentos/JMVC-master/bin/stream_4.264
    InputFile4          /home/juancho/Documentos/JMVC-master/bin/stream_3.264
    InputFile5          /home/juancho/Documentos/JMVC-master/bin/stream_6.264
    InputFile6          /home/juancho/Documentos/JMVC-master/bin/stream_5.264
    InputFile7          /home/juancho/Documentos/JMVC-master/bin/stream_7.264
```

Para ejecutar el ensamblador, fue necesario escribir lo siguiente:

```
    ./MVCBitStreamAssemblerStatic -vf assembler.cfg
```

En la Figura 61 se observa una toma de pantalla del ingreso del código en la respectiva terminal.

Figura 61. Visualización código del ensamblador ejecutado en terminal.
Fuente: el Autor.

En la Figura 62 se puede observar el archivo ensamblado "ballroom_all" con su extensión H.264.

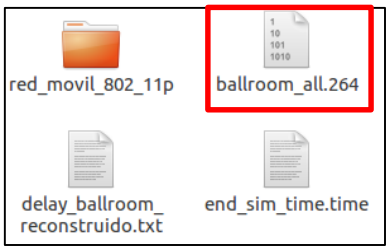

Figura 62. Visualización de la carpeta creada por el ensamblador "ballroom_all.264"
Fuente: el Autor.

El siguiente paso es agrupar los videos en una sola carpeta de simulación y ejecutar el siguiente comando desde una terminal abierta directamente desde la carpeta mencionada (Ver Figura 63).

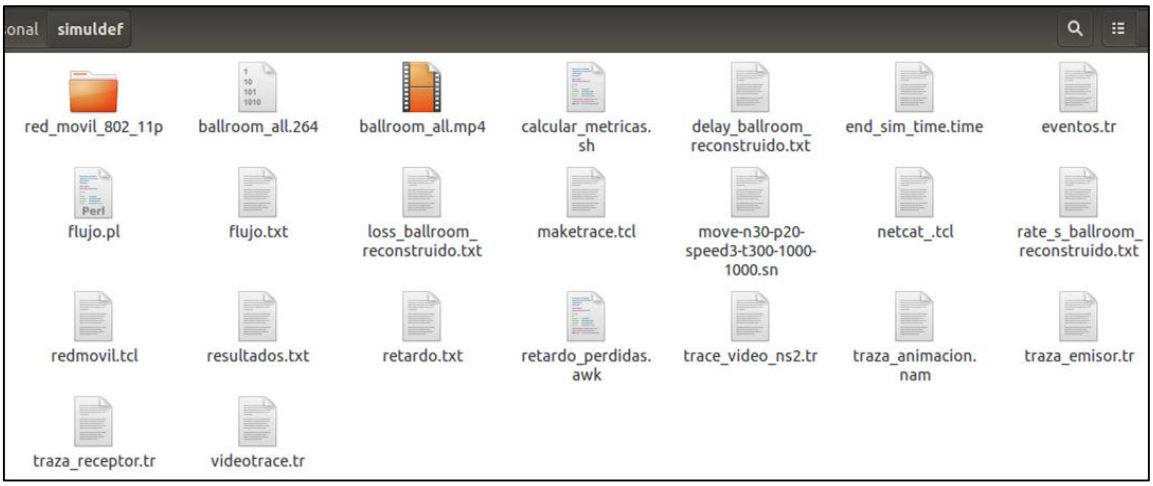

Figura 63. Carpeta con todos los archivos agrupados.
Fuente: el Autor.

4.1.2 Paquetización

En la Figura 64 se muestra una captura de la pantalla del ingreso del código utilizado para el cambio de formato del video en la terminal.

```
juancho@juancho-X550CC:~/simuldef$ MP4Box -hint -mtu 1024 -add ballroom_all.264
-fps 30 ballroom_all.mp4
```

Figura 64. Visualización del comando digitado en la terminal para crear archivo "mp4"
Fuente: el Autor.

En la Figura 65 se puede apreciar el archivo en formato (.mp4).

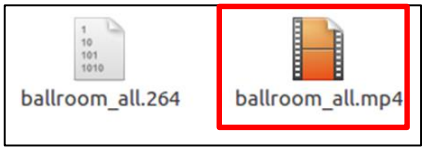

Figura 65. Archivo "mp4" creado a partir del H.264.
Fuente: el Autor.

Al tener el archivo de video en formato mp4, se ejecuta `netcat_.tcl`, el cual también es un archivo que se encuentra dentro de la carpeta de simulación.

```
ns netcat_.tcl
```

4.1.3 Generación del archivo descriptor del video (traza video)

Seguido de ejecutar el "*netcat*", es necesario re direccionar el archivo en formato mp4, asignando una dirección ip y almacenando estos datos en una traza llamada *"videotrace.tr".* El comando empleado para este procedimiento fue el siguiente (Ver Figura 66):

```
juancho@juancho-X550CC:~/simuldef$ wine ~/bin/mp4trace.exe  -f -s 192.168.1.2 12
346 ballroom_all.mp4 > videotrace.tr
```

Figura 66. Re direccionamiento del archivo del video en "mp4"
Fuente: el Autor.

Para dar fin al procedimiento se ejecuta en la misma terminal el siguiente comando: `(killall nc)`, el cual sirve para dejar inactivo "nc", activado anteriormente con "*netcat"*.

Hasta el momento se ha preparado el archivo de video para poder generar la traza ideal para trabajar con "*ns2"* y realizar la simulación final. Para generar la traza se ejecuta el siguiente comando (Ver Figura 67):

```
move-n30-p20-speed3-t300-1000-1000.sn  videotrace.tr
netcat_.tcl
juancho@juancho-X550CC:~/simuldef$ ns maketrace.tcl videotrace.tr trace_video_ns
2.tr
```

Figura 67. Visualización de la terminal con el comando para generar las trazas.
Fuente: el Autor.

En la Figura 68 se puede observar un fragmento del archivo *"videotrace.tr"* , la información está distribuida de la siguiente forma: en la primera columna se imprime el *número del frame*, en la segunda columna se imprime el *tipo de frame* , en la tercera columna se imprime el *tamaño del frame,* en la cuarta columna se imprime el *número total de paquetes utilizados para la transmisión* y en la quinta columna se imprime el *tiempo de envío.*

1	H	17513	19	0.067
2	P	4674	6	0.198
3	P	1201	3	0.199
4	P	732	2	0.199
5	P	719	2	0.199
6	P	4738	6	0.331
7	P	1201	3	0.331
8	P	689	2	0.331
9	P	773	2	0.331
10	H	15992	17	0.462
11	P	1431	3	0.462
12	P	820	2	0.462
13	P	918	2	0.462
14	P	5439	7	0.595

Figura 68. Fragmento del archivo "videotrace.tr"
Fuente: el Autor.

En la Figura 69 se puede observar un fragmento del archivo *"trace_video_ns2.tr"*, la información está distribuida de la siguiente forma: en la primera columna se imprime el *Identificador del archivo*, en la segunda columna se imprime el *tiempo de la traza* , en la tercera columna se imprime el *longitud de la traza,* en la cuarta columna se imprime el *tipo de traza* y en la quinta columna está el *tamaño del fragmentado* (es decir, el tamaño máximo que tendrá el paquete de video).

```
67000 17513 1 0 1024
131000 4674 2 0 1024
1000 1201 2 0 1024
0 732 2 0 1024
0 719 2 0 1024
132000 4738 2 0 1024
0 1201 2 0 1024
0 689 2 0 1024
0 773 2 0 1024
131000 15992 1 0 1024
0 1431 2 0 1024
0 820 2 0 1024
0 918 2 0 1024
132999 5439 2 0 1024
0 1389 2 0 1024
0 952 2 0 1024
0 826 2 0 1024
```

Figura 69. Contenido archivo "trace_video_ns2.tr"
Fuente: el Autor.

4.2 SIMULACIÓN EN NS2

Después de tener las configuraciones anteriormente realizadas, solo queda simular la transmisión del video, y analizar los resultados. El flujo de datos y el retardo, son los aspectos más comúnmente analizados en este tipo de trabajos, donde únicamente se está evaluando el protocolo de transmisión.

Para simular la red, se ha creado el archivo de simulación llamado ***"redmovil.tcl"*** *(Ver* Figura 70***),*** el cual contiene toda la información ya trabajada anteriormente y recopila los datos necesarios para realizar la transmisión definitiva de datos. (Ver Anexo 7.4)

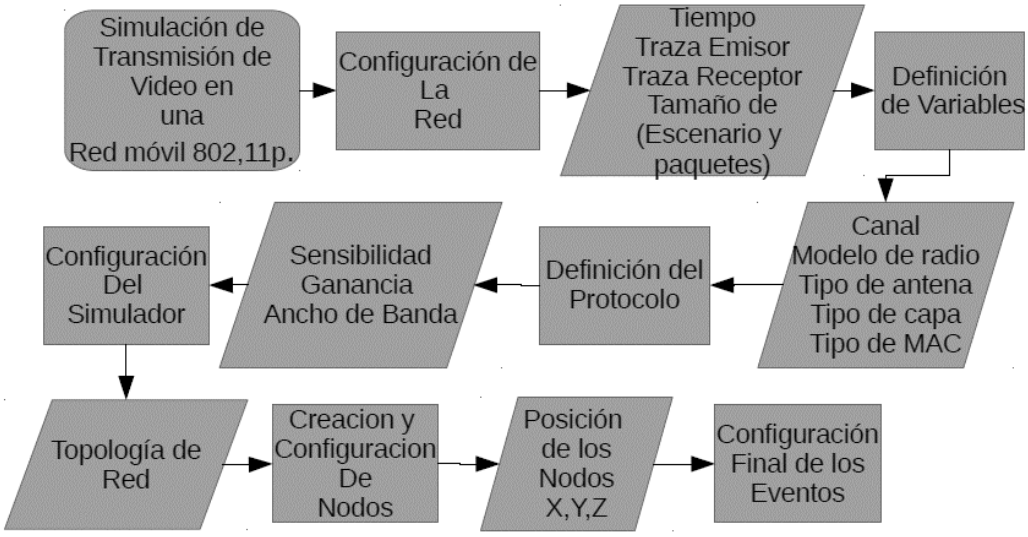

Figura 70. Resumen del Anexo 7.4 archivo "redmovil.tcl"

4.2.1 DESCRIPCIÓN DEL ESCENARIO A SIMULAR

Se ha modelado una red móvil ad hoc, con 30 nodos que poseen movimiento aleatorio, generado a partir de la herramienta *"setdest"* simulando el modelo Random Waypoint Model, el estándar inalámbrico utilizado es el IEEE 802.11p y de enrutamiento AODV, con una capacidad de canal de 11 Mbps, en un rango de transmisión de 1000 metros cuadrados con un tiempo de simulación de 300 segundos (Ver Tabla IV). El software para visualizar la simulación seleccionado es NAM, con el cual se analiza visualmente el comportamiento de la transmisión de datos de video entre el nodo 11 y 17, los cuales a partir de los 100 segundos comienzan a transmitir datos entre sí.

PARAMETROS	DESCRIPCION (VALOR)
Estándar Inalámbrico	802.11p
Capacidad del Canal Inalámbrico	11Mbps
Rango de Transmisión	1000m
Número Total de Nodos	30
Modelo de Movilidad	Random Waypoint Model
Protocolo de Enrutamiento	AODV
Tiempo de simulación	300 (segundos)

Tabla IV. Parámetros relevantes de simulación.

Para compilar y hacer funcionar el archivo de simulación definitiva ***"redmovil.tcl"***, es necesario abrir una terminal e introducir el siguiente comando (Ver Figura 71):

```
juancho@juancho-X550CC:~/simuldef$ ns redmovil.tcl
```

Figura 71. Comando de compilación archivo "redmovil.tcl"
Fuente: el Autor.

Con el paso anterior, ya se ha simulado la red, y lo siguiente que se hace es analizar el retardo y el flujo de datos. Para cada uno de estos procedimientos, existe un algoritmo que será utilizado a continuación, el cual también deberá estar guardado en la carpeta donde se encuentran todos los archivos de simulación y de las trazas.

Para poner a funcionar el programa que calcula el retardo se ejecuta el siguiente código en una terminal desde la carpeta de simulación (Ver Figura 72) el programa implementado para esta operación se encuentra en el Anexo 7.4.2.

```
juancho@juancho-X550CC:~/Documentos/JMVC-master/bin$ cd
juancho@juancho-X550CC:~$ cd ~ simuldef/
juancho@juancho-X550CC:~$ cd ~/simuldef/
juancho@juancho-X550CC:~/simuldef$ awk -f retardo_perdidas.awk eventos.tr > resultados.txt
```

Figura 72 Código para calcular el retardo
Fuente: el Autor.

El resumen del anexo 7.4.2 estará descrito en el siguiente diagrama de flujo de la **¡Error! No se encuentra el origen de la referencia.**:

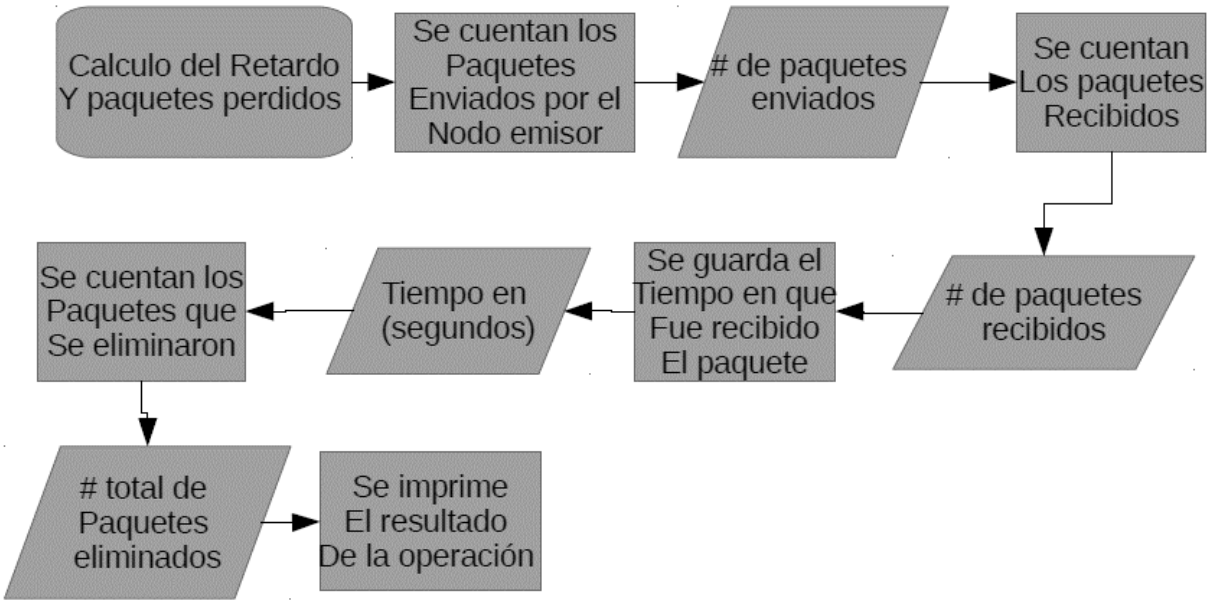

Figura 73. Resumen del funcionamiento del algoritmo para calcular el retardo.

Ahora para calcular el flujo de datos, también se utiliza un algoritmo que ya está formulado y simplemente es compilarlo con los datos de las trazas que se generaron anteriormente con base al proyecto. (Ver Anexo 7.4.1)

El siguiente diagrama de flujo contiene un breve resumen del algoritmo del cálculo del flujo de datos (Ver Figura 74).

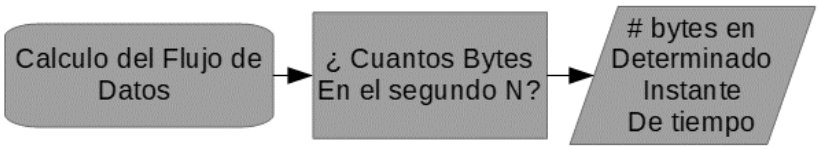

Figura 74. Resumen del funcionamiento del cálculo del flujo de datos.

Para poner a funcionar este programa y que obtengamos el flujo de datos, se ejecuta el siguiente comando desde la terminal:

```
perl flujo.pl eventos.tr 11 0.5 > flujo.txt
```

Por último, se ejecuta el archivo con extensión (. nam) el cual muestra la simulación en tiempo real y donde se puede divisar el flujo de datos entre los nodos. La forma correcta de poner a correr la simulación es utilizando el siguiente comando:

```
sudo nam traza_animacion.nam
```

A continuación, se abrirá una ventana gráfica del entorno de simulación, hacia los 100 segundos, se puede observar como el nodo 11 recibe los datos multimedia correspondientes al video MVC.

4.2.2 Resultados de la simulación

Al compilar el código para calcular el retardo se obtiene el siguiente gráfico del resultado de los mismos (Ver Figura 75) donde se evidencia la fluctuación del retardo debido al movimiento de los nodos. En general durante la simulación no se superaron los 160 ms de retardo, lo cual es favorable para una transmisión de video.

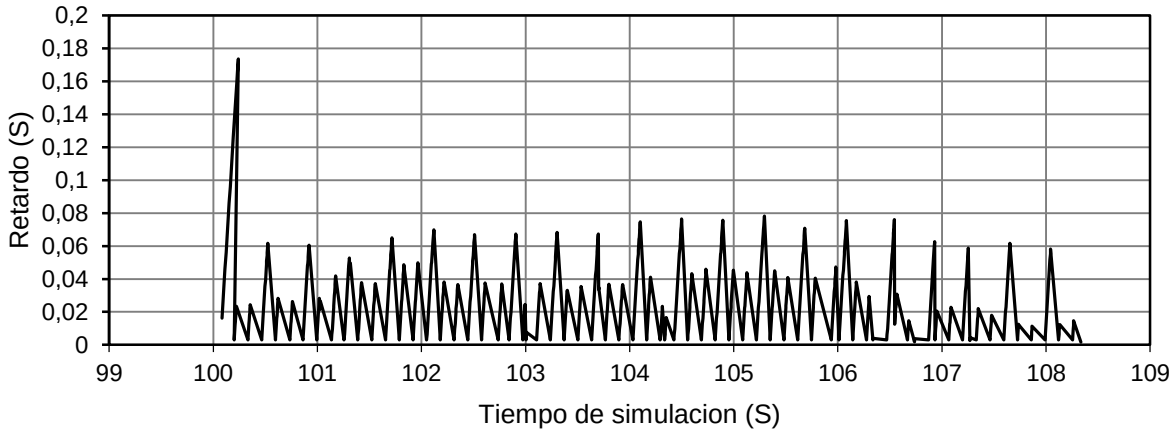

Figura 75. Gráfica del retardo.
Fuente: el Autor.

En cuanto al flujo de datos, se obtiene el siguiente gráfico de la Figura 76 , donde se evidencia que se comienzan a transmitir datos hacia los 100 segundos y se tiene un tiempo aproximado de transmisión de 15 segundos en total (que es el tiempo que tarda el video en transmitirse). La curva del flujo de datos muestra la variación de la cantidad de datos recibidos por el nodo destino por unidad de tiempo. Estas fluctuaciones de deben a la interferencia que ejercen los nodos vecinos y al establecimiento de ruta a través de nodos intermedios. Sin embargo, el flujo de datos obtenido es suficiente para transmitir todos los paquetes del video sin que haya pérdidas.

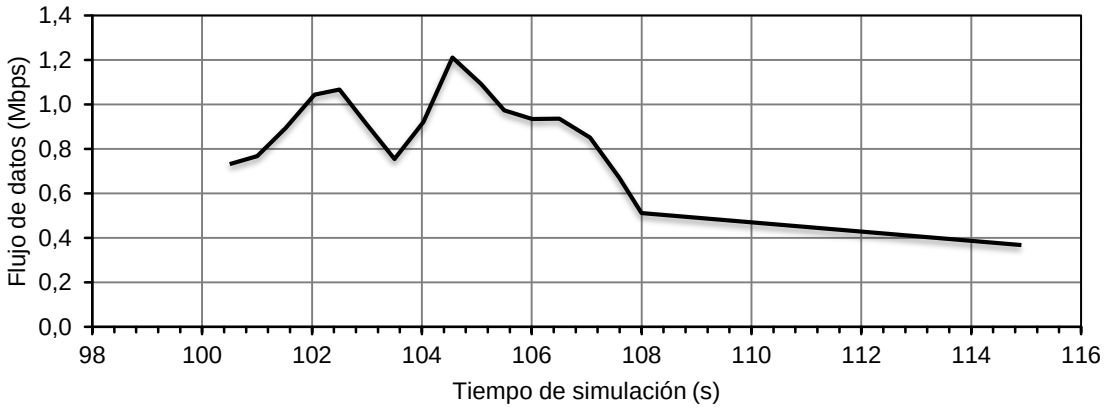

Figura 76. Gráfica del flujo de datos.
Fuente: el Autor.

Finalmente, en la Figura 77 se visualiza el escenario simulado mediante la aplicación NAM. El instante visualizado corresponde aproximadamente a los 133 segundos de simulación y en la figura se pueden apreciar la posición que tenían lo nodos en ese instante.

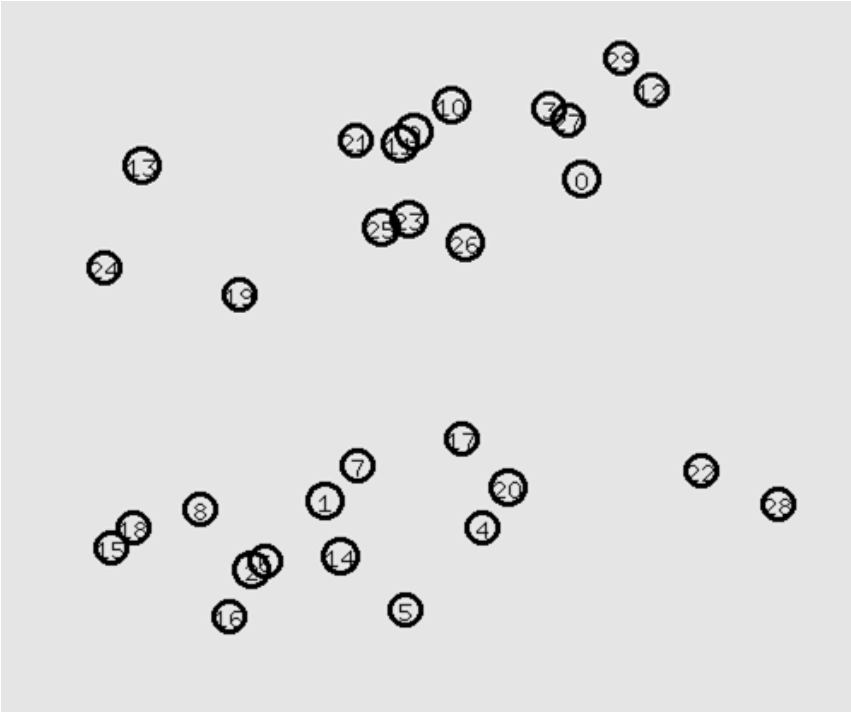

Figura 77. Escenario de simulación de la red para 802.11p.
Fuente: el Autor.

CAPÍTULO 5

5 CONCLUSIONES

En este proyecto, se estudiaron los conceptos de video 3D, principalmente el formato multivista. Este formato ofrece diferentes ventajas respecto a las otras presentaciones 3D ya que existen técnicas como la predicción a través de vistas, que permite aprovechar directamente el hecho de tener redundancia entre cámaras sin exceder el número máximo de imágenes de referencia. Es decir, al momento de codificar el video se tienen en cuenta los fotogramas de otras vistas para calcular la redundancia, esto permite reducir la cantidad de fotogramas de referencias (fotogramas tipo I) que son los de mayor tamaño (en bytes). En comparación con las otras técnicas de video 3D, la codificación MVC, permite una amplia gama de técnicas como lo son la predicción de imágenes modificadas e imágenes interpoladas, lo cual significa una reducción notoria en el procesamiento de las imágenes obtenidas al final y un ahorro de fotogramas al momento de realizar algún tipo de transmisión. Las ventajas y las aplicaciones de la codificación multivista sigue siendo evaluada en la actualidad, principalmente en lo relacionado con las aplicaciones de ultra alta definición. Las aplicaciones recientes han estado enfocadas en escenas de películas con sensaciones de profundidad reales, videoconferencias en vivo con efectos 3D de luminosidad, en televisión inmersiva, en aplicaciones de circuitos cerrados de televisión (punto de vista libre, FVV), filmación de conciertos y deportes de impacto internacional como lo es el fútbol.

Dentro de los diferentes procesos llevados a cabo en el proyecto, se experimentó con la codificación de videos multivista usando el software de referencia JMVC (Joint Multiview Video Coding). En particular, se desarrollaron experimentos a partir un video compuesto por ocho vistas que originalmente están en formato YUV, es decir, sin compresión. El programa utilizado permite trabajar vista por vista y su respectivo ensamblaje en un solo archivo (.264). Es importante resaltar que para este trabajo es esencial tener en cuenta la secuencia en que se codificaron las diferentes vistas. En el desarrollo experimental se descubrió que al codificar de forma aleatoria las vistas, el ensamblador del software no codifica ni crea el archivo final a pesar de que ningún error es mostrado durante el proceso. Se debe tener en cuenta que, con este software, el orden que se utilizó fue el siguiente: primero la vista0- luego la vista2, vista1, vista4, vista3, vista6, vista5, vista7.

En este proyecto también se realizó una búsqueda específica en los fundamentos del protocolo IEEE 802.11p. Las características de este protocolo se compararon el 802.11, encontrándose mejoras de rendimiento en el establecimiento de la comunicación, donde se reducen los tiempos de contacto entre los dispositivos involucrados. Otro aspecto importante a resaltar es la reducción de archivos de caché o archivos indeseados en las tramas transmitidas, ya que contiene únicamente la información esencial, transmite las tramas bajo demanda y se elimina el proceso de autenticación.

El estándar 802.11p fue evaluado mediante la simulación de una red inalámbrica en el software NS2. Para el desarrollo de esta simulación fue necesaria la instalación del protocolo dentro del simulador. Para lograr dicha instalación se debe resaltar como procedimiento fundamental la modificación del núcleo del simulador y la recompilación del mismo. Específicamente se modificaron los archivos “*mac.h”*, “*packet.h”* y “*makefile.in”*. Los resultados

de la simulación se compararon con la simulación del estándar 802.11 y se evidenció las ventajas del protocolo 802.11p en cuanto a la mejora que se obtiene en el flujo de datos y las pérdidas de paquetes. Esto permitió identificar cómo se comporta el protocolo 802.11p frente a la transmisión sobre las redes multisalto. La principal conclusión del estudio del estándar 802.11p fue que, a pesar de la reducción del porcentaje de paquetes perdidos, en comparación con 802.11, cada paquete sigue siendo parte fundamental y únicamente en el proceso se trabaja con los que permiten concluir el procedimiento. Para evitar estas pérdidas la fuente de tráfico no debería transmitir a la máxima tasa que le permita el estándar. Por ejemplo, si se está utilizando el estándar 802.11p con una data rate de 6Mbps, el dispositivo transmisor debería transmitir por debajo de los 5Mbps para garantizar una pérdida de paquetes inferior al 50%.

Después de evaluar el comportamiento del protocolo 802.11p, se estudiaron los diferentes métodos para evaluar la transmisión de video sobre redes de comunicaciones. La estrategia elegida fue la integración de las herramientas de la plataforma Evalvid y el simulador NS2. Fue elegida Evalvid ya que se ha convertido en la herramienta generalmente utilizada para este tipo de estudios. Sin embargo, Evalvid no incluye mecanismos para reproducir los efectos de transmitir el video sobre una red. Por esto, es necesario integrarla con un simulador y así obtener estudios realistas. La integración con NS2 presentó varios errores los cuales fueron superados a través de múltiples ensayos. También se obtuvieron dificultades al momento de buscar que Evalvid y el estándar 802.11p quedaran instalados en el mismo simulador. Esto era esencial con el fin de buscar la transmisión de video MVC sobre el estándar 802.11p.

Después de la integración de Evalvid y NS2 se hace la evaluación de la transmisión de video MVC sobre una red 802.11p. Se evaluó el retardo y el flujo de datos sufrido por el tráfico de video. En cuanto al retardo se puede decir que la simulación indica que este está bajo unos límites admisibles para un tráfico de video. En lo referente al flujo de datos se observó la variación de este a medida que los nodos se mueven por el escenario.

Aunque los escenarios simulados y los resultados fueron limitados, lo importante de este proyecto fue el lograr la integración de los conceptos de 802.11p y de la transmisión de video MVC. De tal manera que se aportaron las bases para desarrollar en el futuro proyectos más avanzados.

A nivel general, las principales dificultades del proyecto estuvieron relacionadas directamente con la versión del sistema operativo utilizada, debido a que al intentar compilar los programas implicados como *NS2*, *Evalvid*, entre otros, surgieron una serie de errores que sugerían el cambio oportuno de la versión de Linux, específicamente la versión de Ubuntu utilizada, Sin embargo, se investigaron a fondo los errores y fue posible modificar internamente los códigos fuente para que funcionaran correctamente cada uno de los programas.

El uso de nuevos lenguajes de programación también fue un reto especial en este proyecto, lenguajes como *“tcl”*, utilizado directamente en la simulación, “*c++”*, con él que fue posible modificar el código fuente de NS2.

6 REFERENCIAS

[1] E. de P. de I. de C. Jolimar Rivas (06-40174), Abel J. Francisco (06-39546), «Percepción Visual de Figuras 3D (Mar 2011)», 2014.

[2] U. de C.-L. Mancha, «Compresión de la información de vídeo 3. Compresión de la información de vídeo 3.1. Introducción», 2014.

[3] R. H. Cuenca, «CLASIFICACIÓN DE LOS MODELOS DE FUENTES DE VIDEO SOBRE INTERNET», 2009.

[4] D. de electronica UTFSM, «Analise da Transmissao de Video 3D em Redes De ComunicaÇoes», pp. 1-2, 2012.

[5] T. Wiegand, «Overview of the H. 264/AVC video coding standard», *… Syst. Video …*, vol. 13, n.º 7, pp. 560-576, 2003.

[6] E. En y S. E. Imagen, «Escuela técnica superior de ingenieros industriales y de telecomunicación», pp. 1-50, 2013.

[7] E. Seoane, «Home Theater 3D: Reproductor de cine 3D casero de bajo presupuesto - Efimeroteca», 2010. [En línea]. Disponible en: http://www.efimeroteca.com/blog/creatividad/home-theater-3d-tu-cine-3d-casero-de-bajo-presupuesto/. [Accedido: 09-nov-2017].

[8] P. Hanhart, «3D Video Quality Assessment | MMSPG», 2015. [En línea]. Disponible en: https://mmspg.epfl.ch/3dvqa. [Accedido: 09-nov-2017].

[9] C. Bal, «Three-dimensional Video Coding on Mobile Platforms», n.º September, p. 103, 2009.

[10] P. La, T. De, V. José, V. Lumpié, J. Ramón, y C. Bueno, «Análisis y Evaluacion De Las Tecnicas Utilizadas Para La Transmision De 3D», 2014.

[11] F. Díaz De María y M. De, «El est ándar de codificación de vídeo H.264/ AVC Claves para una codificación híbrida m ás pot ent e (I I)», 2012.

[12] R. the media Bits, «MPEG-4 Inside – Advanced Video Coding (AVC) – Riding the Media Bits», 2013. [En línea]. Disponible en: http://ride.chiariglione.org/mpeg-4-inside-advanced-video-coding-avc/. [Accedido: 12-oct-2017].

[13] R. Panda, A. Das, y A. K. Roy-Chowdhury, «Video Summarization in a Multi-View Camera Network», 2014.

[14] Y. Chen, Y. K. Wang, K. Ugur, M. M. Hannuksela, J. Lainema, y M. Gabbouj, «The emerging MVC standard for 3D video services», *EURASIP J. Adv. Signal Process.*, vol. 2009, 2009.

[15] V. C. E. G. (VCEG), «JMVC - Joint Multiview Video Coding», 2012. [En línea]. Disponible en: https://github.com/cmurphy/JMVC#jmvc---joint-multiview-video-coding.

[16] J. Multiview *et al.*, «MVC Software Manual», vol. 1, pp. 1-26, 2013.

[17] J. A. Guerrero-ibáñez, C. Flores-cortés, y A. B. Marti, «De Handoff Dentro De Un Entorno De Redes Vehiculares», n.º May, pp. 148-154, 2014.

[18] G. Jiménez Pinto, D. A. López Sarmiento, y L. F. Pedraza Martínez, «Simulación

y análisis de desempeño de protocolos unicast para Redes VANET», *Tecnura Tecnol. y Cult. Afirmando el Conoc. ISSN-e 0123-921X, Vol. 15, Nº. 31, 2011, págs. 66-75*, vol. 15, n.º 31, pp. 66-75, 2011.

[19] E. Paola, P. Sanchez, y D. I. J. Bravo, «Análisis de Simulacion De la Diseminacion De Mensajes De Emergencia En Redes Vehiculares AD-HOC Mediante Software Libre», vol. 2013, 2013.

[20] R. W. World, «WLAN 802.11a vs 802.11p-Difference between 802.11a,802.11p», 2011. [En línea]. Disponible en: http://www.rfwireless-world.com/Terminology/WLAN-802-11a-versus-802-11p.html. [Accedido: 21-oct-2017].

[21] D. de electronica UTFSM, «Analise da Transmissao de Video 3D em Redes De ComunicaÇoes», 2013.

[22] S. Hussain, «All About Ad hoc Networks: NS-2.33 Installation on Ubuntu 10.04», 2013. [En línea]. Disponible en: http://hackingstuforbeginners.blogspot.com.co/2011/10/ns-233-installation-on-ubuntu-1004.html. [Accedido: 22-oct-2017].

[23] GRC Universidad Politecnica De Valencia, «Transmision En Internet: Streaming de Audio y Video», 2013.

[24] J. F. M. Puga, G. Maciá-fernández, A. Grilo, y N. M. C. Tiglao, «Transmisión eficiente de datos multimedia en redes inalámbricas de sensores», pp. 1-8, 2010.

[25] J. F. Kurose y Ì. Ê. Ê, *Redes de computadoras (un enfoque descendente)*. 2009.

[26] W. E. Castellanos, J. C. Guerri, y P. Arce, «A QoS-aware routing protocol with adaptive feedback scheme for video streaming for mobile networks», *Comput. Commun.*, vol. 77, n.º C, pp. 10-25, mar. 2016.

[27] W. Castellanos, J. C. Guerri, y P. Arce, «Performance Evaluation of Scalable Video Streaming in Mobile Ad hoc Networks», *IEEE Lat. Am. Trans.*, vol. 14, n.º 1, pp. 122-129, 2016.

[28] W. E. Castellanos, J. C. Guerri, y P. Arce, «SVCEval-RA: an evaluation framework for adaptive scalable video streaming», *Multimed. Tools Appl.*, vol. 76, n.º 1, pp. 437-461, 2017.

[29] C. H. Ke, C. K. Shieh, W. S. Hwang, y A. Ziviani, «An evaluation framework for more realistic simulations of MPEG video transmission», *J. Inf. Sci. Eng.*, vol. 24, n.º 2, pp. 425-440, 2008.

[30] K. (柯志亨) Hih-Heng, «myEvalvid-NT (myEvalvid Network Trace)», 2010. [En línea]. Disponible en: http://csie.nqu.edu.tw/smallko/ns2_old/myEvalvidNT.htm. [Accedido: 06-nov-2017].

[31] V. C. E. G. (VCEG), «MVC Sofrware Manual», 2011.

ANEXOS

7 ANEXOS

7.1 SCRIPT TCL DE SIMULACIÓN DE UNA RED 802.11.

```
set rate_Mbps 5000000

# Simulacion de 6 nodos
# (nodos móviles) n1 a n5 son nodos inalámbricos fijos, n0 un nodo móvil
# ···················································

#
#      n1     n2     n3     n4     n5
#   n0 ----------------------------->n0
#      <-----------------------------
#
# n0 envia al n5
#·························································


# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# DEFINICION VARIABLES
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
set val(chan)         Channel/WirelessChannel   ;# Tipo de canal
set val(prop)         Propagation/TwoRayGround ;# Modelo de radio
set val(ant)          Antenna/OmniAntenna      ;# Tipo de Antena
set val(ll)                   LL               ;# Tipo de capa de enlace
set val(ifq)          Queue/DropTail/PriQueue  ;# Cola de interfaz
set val(ifqlen)       50                       ;# paquete máximo en ifq
set val(netif)        Phy/WirelessPhy          ;# tipo de interfaz de red
set val(mac)          Mac/802_11               ;# Tipo de MAC
set val(nn)                   6                ;# Numero de nodos móviles
set val(rp)                   AODV             ;# protocolo de enrutamiento
set val(x)                    1500
set val(y)                    1400
set val(stop)         270                      ;#finaliza la simulación


Mac/802_11 set dataRate_ 6Mb
Mac/802_11 set basicRate_ 6Mb

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DEL SIMULADOR
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
set ns [new Simulator]

#Configuración de la traza de eventos
set Traza_eventos [open eventos.tr w]


set TrazaFlujo [open traza_flujo.tr w]
set TrazaPerdidos [open traza_perdidos.tr w]
set TrazaRecibidos [open traza_recibidos.tr w]


$ns trace-all $Traza_eventos
#$ns trace-all $TrazaFlujo
#$ns trace-all $TrazaPerdidos
```

```
#$ns trace-all $TrazaRecibidos


# Configuración de la traza con los datos de la visualización de la topologia
set Traza_nam [open traza_animacion.nam w]
$ns namtrace-all-wireless $Traza_nam $val(x) $val(y)

#Se configura el tamaño de la topologia
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
#
create-god $val(nn)
set god [God instance]
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CREACION DE LOS NODOS Y CONFIGURACION
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# Se asignan las configuraciones a todos los nodos que se van a crear
$ns node-config -adhocRouting $val(rp) \
                                               -llType $val(ll) \
                                               -macType $val(mac) \
                                               -ifqType $val(ifq) \
                                               -ifqLen $val(ifqlen) \
                                               -antType $val(ant) \
                                               -propType $val(prop) \
                                               -phyType $val(netif) \
                                               -channelType $val(chan) \
                                               -topoInstance $topo \
                                               -agentTrace ON \
                                               -routerTrace ON \
                                               -macTrace ON \
                                               -movementTrace ON \

#Creación de los nodos
for {set i 0} {$i < $val(nn)} {incr i} {
     set node_($i) [$ns node]
}

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# Tamaño inicial de los nodos (para la visualización de la topologia en nam)
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
for {set i 0} {$i < $val(nn)} {incr i} {
     $ns initial_node_pos $node_($i) 40
}


# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DEL TRAFICO
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~

#Se configura el nodo receptor (nodo 5)
set sink5 [new Agent/LossMonitor]
$ns attach-agent $node_(1) $sink5

# Se configura el nodo emisor (nodo 0)

set udp_emisor [new Agent/UDP]
$ns attach-agent $node_(0) $udp_emisor

# Se configura la comunicacion nodo 0 -> nodo 5
$ns connect $udp_emisor $sink5
```

```
    #Se crea la fuente de trafico
    set fuente_cbr [new Application/Traffic/CBR]
    $fuente_cbr set packetSize_ 1000
    $fuente_cbr set rate_ $rate_Mbps
    $fuente_cbr attach-agent $udp_emisor



    # =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
    # CONFIGURACION DE EVENTOS
    # =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
    $ns at 0.0 "grabar"
    $ns at 20.0 "$fuente_cbr start"
    $ns at val(stop).0 "$fuente_cbr stop"


    $node_(0) set X_ 50.0
    $node_(0) set Y_ 100.0
    $node_(0) set Z_ 0.0

    $node_(1) set X_ 300.0
    $node_(1) set Y_ 100.0
    $node_(1) set Z_ 0.0

    $node_(2) set X_ 550.0
    $node_(2) set Y_ 100.0
    $node_(2) set Z_ 0.0

    $node_(3) set X_ 800.0
    $node_(3) set Y_ 100.0
    $node_(3) set Z_ 0.0

    $node_(4) set X_ 1050.0
    $node_(4) set Y_ 100.0
    $node_(4) set Z_ 0.0

    $node_(5) set X_ 1300.0
    $node_(5) set Y_ 100.0
    $node_(5) set Z_ 0.0

    # - Restablecer todos los nodos cuando se alcanza el tiempo de simulación for
{set i 0} {$i < $val(nn)} {incr i} {
         $ns at $val(stop).0 "$node_($i) reset"
    }



    $ns at 0.0 "$node_(1) setdest 300.0 100.0 20.0"
    $ns at 0.0 "$node_(2) setdest 550.0 100.0 20.0"
    $ns at 0.0 "$node_(3) setdest 800.0 100.0 20.0"
    $ns at 0.0 "$node_(4) setdest 1050.0 100.0 20.0"
    $ns at 0.0 "$node_(5) setdest 1300.0 100.0 20.0"

    $ns at 10.0 "$node_(0) setdest 1300.0 200.0 5.5"

    # - Finalizando la simulación y nam

    $ns at $val(stop).0 "$ns nam-end-wireless $val(stop)"
    $ns at $val(stop).0002 "finalizar"
    $ns at $val(stop).1000 "puts \"ns-2 ENDS simulation\" ; $ns halt"
```

```
   # Se crea el procedimiento "final"
   proc finalizar {} {
        global ns Traza_eventos Traza_nam TrazaFlujo TrazaPerdidos TrazaRecibidos
        $ns flush-trace
        close $Traza_nam
        close $Traza_eventos

        close $TrazaFlujo
        close $TrazaPerdidos
        close $TrazaRecibidos
        exit 0
   }

   proc grabar {} {
      global sink5 TrazaFlujo TrazaPerdidos TrazaRecibidos
   #se crea una instancia del planificador
      set ns [Simulator instance]
   #periodo de tiempo después del cual se volverá a llamar el proc grabar
      set tiempo 1.0
   #se cuentan cuantos bytes llegan al sumidero
      set BYTES [$sink5 set bytes_]
   #Se cuentan los paquetes perdidos
      set paq_perdidos [$sink5 set nlost_]
   #Se cuentan los paquetes recibidos
      set paq_recibidos [$sink5 set npkts_]
   #almacenar el tiempo actual
      set ahora [$ns now]
   #Se calcula el flujo de datos (throughput) [Mbits/seg] y se guarda en el archivo
respectivo
        set flujo [expr ($BYTES/$tiempo)*8/1000000]
      puts $TrazaFlujo "$ahora $flujo"

      if {$paq_recibidos == 0} {
        puts $TrazaPerdidos "$ahora 0"
        puts $TrazaRecibidos "$ahora 0"
      }
      if {$paq_recibidos !=0} {
                set total_paquetes [expr $paq_perdidos.0 + $paq_recibidos.0]
        set porcentaje_paq_perdidos [expr ($paq_perdidos.0/$total_paquetes )*100]
        puts $TrazaPerdidos "$ahora $porcentaje_paq_perdidos"

        set porcentaje_paq_recibidos [expr ($paq_recibidos.0/$total_paquetes )*100]
        puts $TrazaRecibidos "$ahora $porcentaje_paq_recibidos"
      }

   #Ponemos a cero los valores de los sumideros
      $sink5 set bytes_ 0 ;
      $sink5 set nlost_ 0 ;
      $sink5 set npkts_ 0 ;
   #Se programa la próxima llamada del procedimiento “grabar”
      $ns at [expr $ahora+$tiempo] "grabar"
   }
   puts "Inicio de la Simulacion.."
   $ns run
```

7.2 ARCHIVO DE SIMULACIÓN DE UNA RED 802.11P.

```
    set rate_Mbps                                    5000000

    # Simulacion de 6 nodos
    #el nodo 0 es un nodo movil, mientras que los nodos 1,2,3,4y5 con nodos
inalambricos fijos

    # ...................................................

    #
    #      n1     n2     n3     n4     n5
    #   n0 ----------------------------->n0
    #
    #
    # n0 transmite a n1
    #.......................................................


    # =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
    # DEFINICION VARIABLES
    # =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
    set val(chan)         Channel/WirelessChannel   ;# tipo de canal
    set val(prop)         Propagation/TwoRayGround ;# modelo de radio propagacion
    set val(ant)          Antenna/OmniAntenna      ;# tipo de antena
    set val(ll)                   LL               ;# tipo de capa de enlace de
datos
    set val(ifq)          Queue/DropTail/PriQueue ;# tipo de cola
    set val(ifqlen)       50                       ;# tamaño de la cola
    set val(netif)        Phy/WirelessPhyExt               ;# interface de red
    set val(mac)          Mac/802_11Ext            ;# MAC type
    set val(nn)                   6                ;# number of mobile nodes
    set val(rp)                   AODV             ;# protocolo de enrutamiento
    set val(y)                    1400
    set val(stop)         270                      ;#tiempo de fin de la simulación


    # =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
    # DEFINICION PROTOCOLO 802.11P
    # =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
    Phy/WirelessPhyExt set CSThresh_               3.9810717055349694e-13;# -94 dBm
wireless interface sensitivity
    Phy/WirelessPhyExt set Pt_                     0.1                    ;
    Phy/WirelessPhyExt set freq_                   5.9e+9
    Phy/WirelessPhyExt set noise_floor_       1.26e-13 ;
    Phy/WirelessPhyExt set L_                      1.0     ;
    Phy/WirelessPhyExt set PowerMonitorThresh_     3.981071705534985e-18  ;
    Phy/WirelessPhyExt set HeaderDuration_         0.000040  ;
    Phy/WirelessPhyExt set BasicModulationScheme_  0 ;
    Phy/WirelessPhyExt set PreambleCaptureSwitch_  1 ;
    Phy/WirelessPhyExt set DataCaptureSwitch_      1
    Phy/WirelessPhyExt set SINR_PreambleCapture_   3.1623;                ;
    Phy/WirelessPhyExt set SINR_DataCapture_       10.0;                  ;
    Phy/WirelessPhyExt set trace_dist_             1e6                    ;

    Phy/WirelessPhyExt set PHY_DBG_                0

    Mac/802_11Ext set CWMin_                       15
    Mac/802_11Ext set CWMax_                       1023
    Mac/802_11Ext set SlotTime_                    0.000013
```

```
Mac/802_11Ext set SIFS_                       0.000032
Mac/802_11Ext set ShortRetryLimit_            7
Mac/802_11Ext set LongRetryLimit_             4
Mac/802_11Ext set HeaderDuration_             0.000040
Mac/802_11Ext set SymbolDuration_             0.000008
Mac/802_11Ext set BasicModulationScheme_      0
Mac/802_11Ext set use_802_11a_flag_           true
Mac/802_11Ext set RTSThreshold_               2346
Mac/802_11Ext set MAC_DBG                     0

Mac/802_11Ext set dataRate_ 6Mb
Mac/802_11Ext set basicRate_ 6Mb

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DEL SIMULADOR
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
set ns [new Simulator]

#Configuración de la traza de eventos
set Traza_eventos [open eventos.tr w]


set TrazaFlujo [open traza_flujo.tr w]
set TrazaPerdidos [open traza_perdidos.tr w]
set TrazaRecibidos [open traza_recibidos.tr w]


$ns trace-all $Traza_eventos
#$ns trace-all $TrazaFlujo
#$ns trace-all $TrazaPerdidos
#$ns trace-all $TrazaRecibidos


# Configuración de la traza con los datos de la visualización de la topologia
set Traza_nam [open traza_animacion.nam w]
$ns namtrace-all-wireless $Traza_nam $val(x) $val(y)

#Se configura el tamaño de la topologia
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
#
create-god $val(nn)
set god [God instance]
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CREACION DE LOS NODOS Y CONFIGURACION
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# Se asignan las configuraciones a todos los nodos que se van a crear
$ns node-config -adhocRouting $val(rp) \
                                               -llType $val(ll) \
                                               -macType $val(mac) \
                                               -ifqType $val(ifq) \
                                               -ifqLen $val(ifqlen) \
                                               -antType $val(ant) \
                                               -propType $val(prop) \
                                               -phyType $val(netif) \
                                               -channelType $val(chan) \
                                               -topoInstance $topo \
                                               -agentTrace ON \
                                               -routerTrace ON \
                                               -macTrace ON \
                                               -movementTrace ON \
```

```
#Creación de los nodos
for {set i 0} {$i < $val(nn)} {incr i} {
     set node_($i) [$ns node]
}

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# Tamaño inicial de los nodos (para la visualización de la topologia en nam)
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
for {set i 0} {$i < $val(nn)} {incr i} {
     $ns initial_node_pos $node_($i) 40
}


# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DEL TRAFICO
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~

#Se configura el nodo receptor (nodo 5)
set sink5 [new Agent/LossMonitor]
$ns attach-agent $node_(1) $sink5

# Se configura el nodo emisor (nodo 0)

set udp_emisor [new Agent/UDP]
$ns attach-agent $node_(0) $udp_emisor

# Se configura la comunicacion nodo 0 -> nodo 5
$ns connect $udp_emisor $sink5

#Se crea la fuente de trafico
set fuente_cbr [new Application/Traffic/CBR]
$fuente_cbr set packetSize_ 1000
$fuente_cbr set rate_ $rate_Mbps
$fuente_cbr attach-agent $udp_emisor



# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DE EVENTOS
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
$ns at 0.0 "grabar"
$ns at 20.0 "$fuente_cbr start"
#$ns at 165.0 "$fuente_cbr stop"

#$ns at 188.0 "$fuente_cbr start"
#$ns at 268.0 "$fuente_cbr stop"

#$ns at 299.0 "$fuente_cbr start"
#$ns at 370.0 "$fuente_cbr stop"

#$ns at 397.0 "$fuente_cbr start"
#$ns at 467.0 "$fuente_cbr stop"

#$ns at 494.0 "$fuente_cbr start"
$ns at val(stop).0 "$fuente_cbr stop"


$node_(0) set X_ 50.0
$node_(0) set Y_ 100.0
$node_(0) set Z_ 0.0
```

```
$node_(1) set X_ 300.0
$node_(1) set Y_ 100.0
$node_(1) set Z_ 0.0

$node_(2) set X_ 550.0
$node_(2) set Y_ 100.0
$node_(2) set Z_ 0.0

$node_(3) set X_ 800.0
$node_(3) set Y_ 100.0
$node_(3) set Z_ 0.0

$node_(4) set X_ 1050.0
$node_(4) set Y_ 100.0
$node_(4) set Z_ 0.0

$node_(5) set X_ 1300.0
$node_(5) set Y_ 100.0
$node_(5) set Z_ 0.0

# -Reset all nodes when simulation time is reached
for {set i 0} {$i < $val(nn)} {incr i} {
     $ns at $val(stop).0 "$node_($i) reset"
}



$ns at 0.0 "$node_(1) setdest 300.0 100.0 20.0"
$ns at 0.0 "$node_(2) setdest 550.0 100.0 20.0"
$ns at 0.0 "$node_(3) setdest 800.0 100.0 20.0"
$ns at 0.0 "$node_(4) setdest 1050.0 100.0 20.0"
$ns at 0.0 "$node_(5) setdest 1300.0 100.0 20.0"

$ns at 10.0 "$node_(0) setdest 1300.0 200.0 5.5"
#$ns at 690.0 "$node_(0) setdest 800.0 200.0 0.8"
#$ns at 710.0 "$node_(0) setdest 550.0 200.0 0.8"
#$ns at 501.0 "$node_(0) setdest 100.0 200.0 2.5"
#$ns at 20.0 "$node_(5) setdest 1490.0 100.0 15.0"


# -Ending the simulation and nam

$ns at $val(stop).0 "$ns nam-end-wireless $val(stop)"
$ns at $val(stop).0002 "finalizar"
$ns at $val(stop).1000 "puts \"ns-2 ENDS simulation\" ; $ns halt"


# Se crea el procedimiento "final"
proc finalizar {} {
     global ns Traza_eventos Traza_nam TrazaFlujo TrazaPerdidos TrazaRecibidos
     $ns flush-trace
     close $Traza_nam
     close $Traza_eventos

     close $TrazaFlujo
     close $TrazaPerdidos
     close $TrazaRecibidos
     exit 0
}
```

```
    proc grabar {} {
       global sink5 TrazaFlujo TrazaPerdidos TrazaRecibidos
    #se crea una instancia del planificador
       set ns [Simulator instance]
    #periodo de tiempo después del cual se volverá a llamar el proc grabar
       set tiempo 1.0
    #se cuentan cuantos bytes llegan al sumidero
       set BYTES [$sink5 set bytes_]
    #Se cuentan los paquetes perdidos
       set paq_perdidos [$sink5 set nlost_]
    #Se cuentan los paquetes recibidos
       set paq_recibidos [$sink5 set npkts_]
    #almacenar el tiempo actual
       set ahora [$ns now]
    #Se calcula el flujo de datos (throughput) [Mbits/seg] y se guarda en el archivo
respectivo
        set flujo [expr ($BYTES/$tiempo)*8/1000000]
       puts $TrazaFlujo "$ahora $flujo"

       if {$paq_recibidos == 0} {
         puts $TrazaPerdidos "$ahora 0"
         puts $TrazaRecibidos "$ahora 0"
       }
       if {$paq_recibidos !=0} {
                 set total_paquetes [expr $paq_perdidos.0 + $paq_recibidos.0]
         set porcentaje_paq_perdidos [expr ($paq_perdidos.0/$total_paquetes )*100]
         puts $TrazaPerdidos "$ahora $porcentaje_paq_perdidos"

         set porcentaje_paq_recibidos [expr ($paq_recibidos.0/$total_paquetes )*100]
         puts $TrazaRecibidos "$ahora $porcentaje_paq_recibidos"
       }

    #Ponemos a cero los valores de los sumideros
       $sink5 set bytes_ 0 ;
       $sink5 set nlost_ 0 ;
       $sink5 set npkts_ 0 ;
    #Se programa la próxima llamada del procedimiento “grabar”
       $ns at [expr $ahora+$tiempo] "grabar"
    }
    puts "Inicio de la Simulacion.."
    $ns run
```

7.3 ARCHIVO DE CONFIGURACIÓN PARA LA CODIFICACIÓN DEL VIDEO BALLROOM

“ballroomConfig.cfg”

```
    # JMVC Archivo de configuración principal
    #======================GENERAL================================
    InputFile               /home/juancho/Documentos/ballroom/ballroom
    OutputFile              stream     # Archivo de Flujo de Bits
    ReconFile               rec        # Archivo Reconstruido
    MotionFile              motion     # Archivo de información de movimiento
    SourceWidth             640        # Ancho del marco de entrada
    SourceHeight            480        # Altura del marco de entrada
    FrameRate               25.0       # velocidad de fotogramas [Hz]
    FramesToBeEncoded       250        # Número de frames
    #======================CODING ================================
    SymbolMode              1          # 0=CAVLC, 1=CABAC
```

```
FRExt                   1          # 8x8 transform (0:off, 1:on)
BasisQP                 31         # Quantization parameters
#======================INTERLACED============================
MbAff                   0          # 0=frameMb, 1=MbAff
PAff                    0          # 0=frame, 1=field, 2=frame/field
#======================STRUCTURE=============================
GOPSize                 4          # GOP Size (at maximum frame rate)
IntraPeriod             12          # Período de anclaje
NumberReferenceFrames   3          # Número de imágenes de referencia
InterPredPicsFirst      1          # 1 Inter Pics; 0 Inter-view Pics
Log2MaxFrameNum         11   # Especifica max. valor por frame_num (4.16)
Log2MaxPocLsb           7          # Especifica la codificación de POC's (4..15)
DeltaLayer0Quant        0          # Diferencial QP para la capa 0
DeltaLayer1Quant        3          # Diferencial QP para la capa 1
DeltaLayer2Quant        4          # Diferencial QP para la capa 2
DeltaLayer3Quant        5          # Diferencial QP para la capa 3
DeltaLayer4Quant        6          # Diferencial QP para la capa 4
DeltaLayer5Quant        7          # Diferencial QP para la capa 5
MaxRefIdxActiveBL0      2    # active entries in ref list 0 for B slices
MaxRefIdxActiveBL1      2    # active entries in ref list 1 for B slices
MaxRefIdxActiveP        1    # active entries in ref list for P slices
#======================MOTIONSEARCH=======================
SearchMode              4    # Search mode (0:BlockSearch, 4:FastSearch)
SearchFuncFullPel       3    # Search function full pel
                             #   (0:SAD, 1:SSE, 2:HADAMARD, 3:SAD-YUV)
SearchFuncSubPel        2          # Search function sub pel
                                   #   (0:SAD, 1:SSE, 2:HADAMARD)
SearchRange             32         # Search range (Full Pel)
BiPredIter              4          # Max iterations for bi-pred search
IterSearchRange         8    # Search range for iterations (0: normal)
#======================LOOPFILTER ==========================
LoopFilterDisable       0          # Loop filter idc (0: on, 1: off, 2:
                                   #   on except for slice boundaries)
LoopFilterAlphaC0Offset 0          # AlphaOffset(-6..+6): valid range
LoopFilterBetaOffset    0          # BetaOffset (-6..+6): valid range
#=========================WEIGHTEDPREDICTION================
WeightedPrediction      0  # Weighting IP Slice (0:disable, 1:enable)
WeightedBiprediction    0 # Weighting B Slice (0:disable,:explicit, 2:implicit)
#======PARALLELDECODING INFORMATION SEI Message ==============
PDISEIMessage           0  # PDI SEI Habilitar mensaje (0:disable, 1:enable)
PDIInitialDelayAnc      2  # PDI retraso inicial para imágenes de anclaje
PDIInitialDelayNonAnc   2  # PDI para imágenes que no son de anclaje
#==============NESTINGSEIMESSAGE============================
NestingSEI              0   #(0: NestingSEI off, 1: NestingSEI on)
SnapShot                0   #(0: SnapShot off, 1: SnapShot on)
#====================ACTIVEVIEWINFOSEIMESSAGE===============
ActiveViewSEI           0   #(0: ActiveViewSEI off, 1: ActiveViewSEI on)
#==============VIEWSCALABILITYINFOMATIONSEIMESSAGE =========
ViewScalInfoSEI         0   #(0: ViewScalSEI off, 1: ViewScalSEI on)
#============ Level conformance checking of the DPB size ==============
DPBConformanceCheck      1      # (0: disable, 1: enable, 1:default)
#================MULTIVIEWCODINGPARAMETERS ===============
NumViewsMinusOne     7          # (Number of view to be coded minus 1)
ViewOrder        0-2-1-4-3-6-5-7   # (Order in which view_ids are coded)
View_ID     0                      # view_id (0..1024): valid range
Fwd_NumAnchorRefs    0       # (number of list_0 references for anchor)
Bwd_NumAnchorRefs    0      # (number of list 1 references for anchor)
Fwd_NumNonAnchorRefs 0   # (number of list 0 references for non-anchor)
Bwd_NumNonAnchorRefs 0   # (number of list 1 references for non-anchor)
View_ID               1          # view_id (0..1024): valid range
Fwd_NumAnchorRefs      1      # (number of list_0 references for anchor)
```

```
Bwd_NumAnchorRefs      1      # (number of list 1 references for anchor)
Fwd_NumNonAnchorRefs      1 #(number of list 0 references for non-anchor)
Bwd_NumNonAnchorRefs     1  #(number of list 1 references for non-anchor)
Fwd_AnchorRefs      0 0      #ref_idx view_id combination
Bwd_AnchorRefs      0 2      #ref_idx view_id combination
Fwd_NonAnchorRefs      0 0      #ref_idx view_id combination
Bwd_NonAnchorRefs      0 2      #ref_idx view_id combination
View_ID                  2
Fwd_NumAnchorRefs      1
Bwd_NumAnchorRefs      0
Fwd_NumNonAnchorRefs     0
Bwd_NumNonAnchorRefs     0
Fwd_AnchorRefs      0 0
View_ID                  3
Fwd_NumAnchorRefs      1
Bwd_NumAnchorRefs      1
Fwd_NumNonAnchorRefs     1
Bwd_NumNonAnchorRefs     1
Fwd_AnchorRefs      0 2
Bwd_AnchorRefs      0 4
Fwd_NonAnchorRefs      0 2
Bwd_NonAnchorRefs      0 4
View_ID                 4
Fwd_NumAnchorRefs      1
Bwd_NumAnchorRefs      0
Fwd_NumNonAnchorRefs     0
Bwd_NumNonAnchorRefs     0
Fwd_AnchorRefs      0 2
View_ID                 5
Fwd_NumAnchorRefs      1
Bwd_NumAnchorRefs      1
Fwd_NumNonAnchorRefs     1
Bwd_NumNonAnchorRefs     1
Fwd_AnchorRefs      0 4
Bwd_AnchorRefs      0 6
Fwd_NonAnchorRefs      0 4
Bwd_NonAnchorRefs      0 6
View_ID                 6
Fwd_NumAnchorRefs      1
Bwd_NumAnchorRefs      0
Fwd_NumNonAnchorRefs     0
Bwd_NumNonAnchorRefs     0
Fwd_AnchorRefs       0 4
View_ID                 7
Fwd_NumAnchorRefs      1
Bwd_NumAnchorRefs      1
Fwd_NumNonAnchorRefs     0
Bwd_NumNonAnchorRefs     0
Fwd_AnchorRefs      0 6
Bwd_AnchorRefs       0 6
```

7.4 SCRIPT .TCL DE SIMULACIÓN DE UNA TRANSMISIÓN DE VIDEO SOBRE UNA RED MÓVIL 802.11P

"redmovil.tcl"

```
set tiempo_simulacion             300
set traza_original_video  trace_video_ns2.tr
set Traza_emisor                  traza_emisor.tr
set Traza_receptor                traza_receptor.tr
set Tamaño_max_frag               1024
set packetSize                    1052


#=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·
# ns simulacion.tcl >& salida.txt
#=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·=·

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# DEFINICION VARIABLES
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
set val(chan)            Channel/WirelessChannel  ;# tipo de canal
set val(prop)            Propagation/TwoRayGround ;# modelo de radio propagaciónset
val(ant)         Antenna/OmniAntenna              ;# Tipo de Antena
set val(ll)                       LL              ;# Tipo de capa de enlace
set val(ifq)             Queue/DropTail/PriQueue ;# Tipo de cola de interfaz
set val(ifqlen)  50                               ;# paquete máximo en ifq
set val(netif)   Phy/WirelessPhyExt               ;# tipo de interfaz de red
set val(mac)             Mac/802_11Ext                    ;# Tipo de MAC
set val(nn)                       30              ;# número de nodos móviles
set val(rp)                       AODV            ;# protocolo de enrutamiento
set val(x)                        1000
set val(y)                        1000

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# DEFINICION PROTOCOLO 802.11P
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
Phy/WirelessPhyExt set CSThresh_ 3.9810717e-13 ;# -94 dBm w. Sensibilidad
Phy/WirelessPhyExt set Pt_       0.1      ;# 20dBm - Ganancia de la antena 1.0
Phy/WirelessPhyExt set freq_      5.9e+9
Phy/WirelessPhyExt set noise_floor_  1.26e-13   ;# -99 dBm for 10MHz ancho de banda
Phy/WirelessPhyExt set L_         1.0       ;# ganancia / pérdida predeterminada
Phy/WirelessPhyExt set PowerMonitorThresh_ 3.981071705534985e-18 ;# -174 dBm
Phy/WirelessPhyExt set HeaderDuration_          0.000040           ;# 40 us
Phy/WirelessPhyExt set BasicModulationScheme_   0
Phy/WirelessPhyExt set PreambleCaptureSwitch_   1
Phy/WirelessPhyExt set DataCaptureSwitch_       1
Phy/WirelessPhyExt set SINR_PreambleCapture_    3.1623;            ;# 5 dB
Phy/WirelessPhyExt set SINR_DataCapture_        10.0;       ;# 10 dB
Phy/WirelessPhyExt set trace_dist_       1e6       ;# PHY trace Phy/WirelessPhyExt
set PHY_DBG_                 0

Mac/802_11Ext set CWMin_      15
Mac/802_11Ext set CWMax_                        1023
Mac/802_11Ext set SlotTime_                     0.000013
Mac/802_11Ext set SIFS_                         0.000032
Mac/802_11Ext set ShortRetryLimit_              7
Mac/802_11Ext set LongRetryLimit_               4
Mac/802_11Ext set HeaderDuration_               0.000040
```

```
Mac/802_11Ext set SymbolDuration_               0.000008
Mac/802_11Ext set BasicModulationScheme_        0
Mac/802_11Ext set use_802_11a_flag_             true
Mac/802_11Ext set RTSThreshold_                 2346
Mac/802_11Ext set MAC_DBG                       0

Mac/802_11Ext set dataRate_ 11Mb
Mac/802_11Ext set basicRate_ 11Mb

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DEL SIMULADOR
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
set ns [new Simulator]

#Configuración de la traza de eventos
set Traza_eventos [open eventos.tr w]
$ns trace-all $Traza_eventos

# Configuración de la traza con los datos de la visualización de la topologia
set Traza_nam [open traza_animacion.nam w]
$ns namtrace-all-wireless $Traza_nam $val(x) $val(y)

#Se configura el tamaño de la topologia
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
#
create-god $val(nn)
set god [God instance]
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CREACION DE LOS NODOS Y CONFIGURACION
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# Se asignan las configuraciones a todos los nodos que se van a crear
$ns node-config -adhocRouting $val(rp) \
                                                   -llType $val(ll) \
                                                   -macType $val(mac) \
                                                   -ifqType $val(ifq) \
                                                   -ifqLen $val(ifqlen) \
                                                   -antType $val(ant) \
                                                   -propType $val(prop) \
                                                   -phyType $val(netif) \
                                                   -channelType $val(chan) \
                                                   -topoInstance $topo \
                                                   -agentTrace ON \
                                                   -routerTrace ON \
                                                   -macTrace OFF \
                                                   -movementTrace ON \

#Creación de los nodos
for {set i 0} {$i < $val(nn)} {incr i} {
        set node_($i) [$ns node]
}

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# Tamaño inicial de los nodos (para la visualización de la topologia en nam)
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
for {set i 0} {$i < $val(nn)} {incr i} {
        $ns initial_node_pos $node_($i) 40
}

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# Se carga el escenario que esta definido en un archivo adicional
```

```
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
source move-n30-p20-speed3-t300-1000-1000.sn       ;# se carga el escenario

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DEL TRAFICO
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~

#Se configura el nodo receptor (nodo 11)
set sink [new Agent/myEvalvid_Sink]
$ns attach-agent $node_(11) $sink

# Se configura el nodo emisor (nodo 17)
set udp_emisor [new Agent/myUDP]
$ns attach-agent $node_(17) $udp_emisor
$udp_emisor set packetSize_ $packetSize
$udp_emisor set_filename $Traza_emisor

# Se configura la comunicacion nodo 11 -> nodo 17
$ns connect $udp_emisor $sink
$sink set_filename $Traza_receptor ;# lo que reciba el nodo receptor lo registra en
la traza

# Se le pasa a Evalvid el nombre de a traza que describe el video a Tx
set trace_file [new Tracefile]
$trace_file filename $traza_original_video

#Se crea la fuente de video
set Fuente_video [new Application/Traffic/myEvalvid]
$Fuente_video attach-agent $udp_emisor    ;# La fuente de enlaza con el protocolo de
transporte del nodo emisor
$Fuente_video attach-tracefile $trace_file ;# la fuente de video genera el trafico
de acuerdo a la traza

# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
# CONFIGURACION DE EVENTOS
# =~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~=~
$ns at 100.0 "$Fuente_video start"
$ns at $tiempo_simulacion "$Fuente_video stop"
$ns at [expr $tiempo_simulacion+1.0] "$sink closefile"
$ns at [expr $tiempo_simulacion+1.0] "final"

# Se crea el procedimiento "final"
proc final {} {
        global ns Traza_eventos Traza_nam
        $ns flush-trace
        close $Traza_nam
        close $Traza_eventos
        exit 0
}
puts "Inicio de la Simulacion.."
$ns run
```

7.4.1 Algoritmo para calcular el flujo de datos (programado en perl)

```
   $archivo=$ARGV[0];
$nodo="_$ARGV[1]_";
$intt=$ARGV[2];

#El algoritmo calcula cuantos bytes son transmitidos durante un intervalo de tiempo
(intt) en segundos
$sum=0;
```

```
$Acum=0;
      open (DATA,"<$archivo")|| die "Can't open $archivo $!";

 #mira lo que hay a la izquierda del || (que es el "o" lÃ³gico), y si es falso,
ejecuta lo que hay a la derecha

    while (<DATA>) {

@x = split(' '); #split divide la variable por defecto en una serie de cadenas
separadas por espacios y depositando cada una de esas cadenas en un elemento de la
matriz @x

#columna 1 es el tiempo
if ($x[1]-$Acum <= $intt)
{
#checking if the event corresponds to a reception
        if ($x[0] eq 'r')
        {
                #print " eq r \n";
        #checking if the destination corresponds to arg 1
                if ($x[2] eq $nodo)
                {
                        #print " eq nodo \n";
                # mactrace must be OFF in script of simulation

                                if ($x[3] eq 'AGT')
                                {
                                        #print " eq AGT \n";

                                        if($x[6] eq 'video')
                                        {
                                                #print " eq video \n";
                                $sum=$sum+$x[7];
                                }
                                }

                }
        }
}
else
{
 #print STDOUT " $x[1] $sum \n";
        if ($sum ne 0)
        {
        $th=($sum*8/$intt)*(1/1000000);
        print STDOUT " $x[1] $th \n";
        $Acum=$Acum+$intt;
        $sum=0;
        }
        else
        {
        $Acum=$Acum+$intt;
        }
    }
   $th=($sum*8/$intt)*(1/1000000);
   print STDOUT "$x[1] $th\n";

    close DATA;
exit(0);
```

7.4.2 Algoritmo para calcular el retardo y el porcentaje de paquetes perdidos (AWK)

```
    #calcula el retardo y el porcentaje de paquetes perdidos
    #awk -f delay-NOL-PDR_mod.awk traza_eventos.tr >> resulados.txt
    BEGIN {
         No_paq_enviados=0;
         No_paq_recibidos=0;
         TotalEliminados=0;
         nn=30;
         i=0;
    # la identificación del paquete más alta se puede verificar en el archivo de
rastreo para un valor aproximado
         npaq =560;
         sumatoria_retados=0;
    }

    {
         #funciona solo para traza inalàmbrica
         action = $1;
         time = $2;
         packet_id = $6;
         node = $3;
    #
    #==========PACKETS SENDS, RECVS, DROPS ===========
    #Se cuentan los paquetes enviados por el nodo emisor
         if (( action == "s") && ( $7 == "video" ) && ( $4=="AGT" ) )
                 {
                 No_paq_enviados++;
                 #se guarda el tiempo en que fue enviado este paquete
                 send_time[packet_id] = time;
         #printf ("%d ",No_paq_enviados);
                 }
    #Se cuentas los paquetes recibidos
         if ( (action == "r") && ( $7 == "video" ) && ( $4=="AGT" ) )
         #solamente el nodo receptor recibe los paquetes a nivel AGT
         #mactrace deb estar en OFF en el archivo tcl
                 {
                 No_paq_recibidos++;
    #Se guarda el tiempo en que fue recibido el paquete
                 recv_time[packet_id] = time;
         #printf ("%d ",No_paq_recibidos);
                 }

    # Se cuentan los paquetes que se borraron
         if ( (action == "D") && ($7 == "video") && (time >0 ) )
         {
         TotalEliminados++;

         recv_time[packet_id] = -1; #flag para registrar dicho paquete como
eliminado
         }

    }
    END {

     for ( i=0; i <= No_paq_recibidos; i++ )
     {

      tiempo_envio = send_time[i];
      tiempo_llegada = recv_time[i];
      retardo_paquete = tiempo_llegada - tiempo_envio;
```

```
     if ( tiempo_envio < tiempo_llegada)
      {
       sumatoria_retados= retardo_paquete+sumatoria_retardos;
       printf ("%d\t%f\t%f\n " ,i,tiempo_llegada,retardo_paquete);
      }
    }
   #printf ("%d " ,No_paq_recibidos);
   retardo_promedio=sumatoria_retados/No_paq_recibidos;
   porcentaje_perdidos = (TotalEliminados)*100/No_paq_enviados; # porcentaje de
paquetes perdidos (%)

   #printf("#[DataRate][PacketLoss(%)][Average_e-e_delay]\n");
   printf ("pp= %.6f\n", porcentaje_perdidos);

   }
```