Esquema de Marcas de Agua Resistente ante Pérdida de Sincronización Temporal para Video

por

Pedro Aarón Hernández Ávalos

Tesis sometida como requisito parcial
para obtener el grado de

DOCTOR EN CIENCIAS EN EL ÁREA DE CIENCIAS COMPUTACIONALES

en el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Febrero 2012
Tonantzintla, Puebla

Supervisada por:

Dra. Claudia Feregrino Uribe, INAOE

©INAOE 2012
El autor otorga al INAOE el permiso de reproducir
y distribuir copias en su totalidad o en
partes de esta tesis

Resumen

El crecimiento desmedido de Internet en los últimos años, junto con el gran incremento del rendimiento de las computadoras ha facilitado el intercambio de información multimedia, específicamente videos. La necesidad de proteger dichos videos en contra de la piratería se ha vuelto un tema de vital importancia. Las marcas de agua digitales en videos se han utilizado como un medio de protección de derechos de autor, sin embargo éstas pueden servir para diferentes aplicaciones orientadas a enriquecer el video mediante la adición de elementos que se puedan presentar durante la reproducción.

Para que un esquema de marcas de agua sea utilizado en estas aplicaciones se necesita que sea robusto, es decir que pueda extraer el mensaje oculto incluso si el video marcado ha sido modificado. En esta investigación se desarrolla un esquema de marcas de agua, para video, que es resistente ante modificaciones que distorsionan temporalmente al video marcado y ante compresiones MPEG-2, MPEG-4. Adicionalmente, se diseña un benchmark para esquemas de marcas de agua para video, el cual se utiliza para obtener el rendimiento del esquema propuesto. El benchmark propuesto se llama VidMark.

Las principales contribuciones de esta investigación son: el esquema propuesto, el diseño del benchmark, un método para crear redundancia en escenas, un extractor de características las cuales persisten incluso después de haber sufrido distorsiones temporales y finalmente un generador de llaves en usa las características de un bloque de cuadros para la generación de nuevas llaves.

Abstract

The explosive growth of the internet in last years and the great increase in computers performance has allowed the multimedia information exchange. Protecting this information against piracy has become an extremely important requirement, for instance videos must be protected to avoid illegal use. Digital watermarking in videos has been used as a way of copyright protection. Also this can be used for different applications, such as video enhancement through objects addition during video reproduction.

To utilize a video watermarking scheme for this kind of applications the scheme needs to be robust, this means the hidden message must be extracted even if the watermarked video was modified. In this research a video watermarking scheme is developed, which is robust to MPEG-2 and MPEG-4 compressions, and temporal distortions. Additionally, a benchmark for video watermarking schemes is designed, which is utilized to obtain the performance of the proposed scheme. The proposed benchmark is called VidMark.

The main contributions of this work are: the proposed scheme, a benchmark design, a method to create redundancy in frames, a robust feature extractor to temporal distortions, and finally, a key generator which uses features of a block of frames for generating new keys.

Agradecimientos

Al pueblo de México, que mediante el CONACYT se me otorgó la beca número 174444, con la cual me fue posible estudiar este posgrado. De igual manera, agradezco al CONCYTEP por el apoyo otorgado a través del Programa de Becas-Tesis CONCYTEP 2011, el cual me ayudó para la culminación del proyecto de tesis. En la misma medida, agradezco a SEP-CONACYT por apoyar la creación de proyectos de investigación, particularmente el proyecto número CB-2007-1-84668 en donde estuve trabajado con mi investigación doctoral.

Al INAOE, por alimentarme de conocimientos, los cuales me servirán para toda la vida; cuidar de mi bienestar, mediante el otorgamiento de los múltiples apoyos escolares; proveer la infraestructura y las facilidades. Agradezco de todo corazón a mi instituto por ceñirme en sus cálidas alas del saber.

Quiero expresar mi profundo reconocimiento a la Dra. Claudia Feregrino y al Dr. René Cumplido, que durante los últimos años me han ayudado en toda forma posible con sus grandes conocimientos y excelente juicio, ha sido un honor trabajar con ustedes.

Agradezco de igual manera a mis sinodales: Dra. Mariko Nakano, Dr. Gustavo Rodríguez, Dr. Ariel Carrasco, Dr. Miguel Arias, Dr. Saúl Pomares por sus amables comentarios y por compartir su amplia sabiduría conmigo. De igual manera agradezco al Dr. José Juan García por ayudarme tanto, gracias amigo.

Al personal administrativo del INAOE y del CONCYTEP, por sus amables atenciones y buen humor muchas gracias.

Les agradezco de todo corazón a mi familia, Don Pedro, Mi Güera, a mis hermanos Gibrán y Uziel, por su incondicional apoyo en los momentos buenos pero sobre todo en los momentos difíciles, gracias de todo corazón. Te agradezco Jenny por ser parte de mi vida y por apoyarme en todo.

Lamentó mucho que la falta de espacio me impida agradecer nombrando a cada uno de mis amigos por su generosa amistad. Les doy gracias por su estimada compañía.

Dedico esta tesis al viejo, mi abuelo.

Índice general

Índice general VII

1 Introducción **1**
1.1. Importancia de las marcas de agua digitales 1
1.2. Generalidades de marcas de agua 4
Aplicaciones 7
Propiedades 9
1.3. Marcas de agua para video 12
1.4. Objetivos 16
Objetivo general 16
Objetivos específicos 17
1.5. Estructura de la tesis 17

2 Sincronización temporal **19**
2.1. Sincronización en el tiempo 19
Ataques de desincronización 20
2.2. Estrategias utilizadas 21
2.3. Trabajos relacionados 23
2.4. Discusión del capítulo 28

3 Benchmark de ataques temporales **31**
3.1. Justificación 31
3.2. VidMark 33
Ataques temporales 37
Ataques de compresión MPEG 44
3.3. Discusión del capítulo 48

4 Esquema propuesto **51**
4.1. Sincronización temporal basada en llaves 51
4.2. Proceso de incrustación de la marca 53
4.3. Proceso de decodificación de la marca 67
4.4. Análisis 70
4.5. Comparación con el esquema base 75

4.6. Discusión del capítulo 77

5 Resultados **81**
5.1. Descripción de los experimentos 81
5.2. Funcionamiento 82
Objetivo de los experimentos 83
Descripción de los experimentos 83
Desarrollo de los experimentos 84
Discusión de los experimentos 86
5.3. Fidelidad 87
Objetivo del experimento 88
Descripción del experimento 88
Desarrollo del experimento 88
Discusión del experimento 89
5.4. Robustez 89
Objetivo de los experimentos 90
Descripción de los experimentos 90
Desarrollo de los experimentos 91
Ataques del VidMark 91
Comparación con el esquema base 94
Discusión de los experimentos 96
5.5. Capacidad 97
Objetivo de los experimentos 97
Descripción de los experimentos 97
Desarrollo de los experimentos 98
Redundancia de repetición β adaptable 98
Tamaño promedio de escenas 99
Capacidad de inserción del esquema 100
Discusión de los experimentos 102
5.6. Discusión del capítulo 103

6 Conclusiones y trabajo futuro **105**
6.1. Conclusiones 105
6.2. Revisión de objetivos 107
6.3. Contribuciones generadas 109
6.4. Trabajo futuro 109

Apéndices **110**

A Publicaciones derivadas **111**

Índice de figuras **139**

Índice de tablas **141**

Glosario **143**

Siglas **147**

Referencias **149**

Capítulo 1

Introducción

En este capítulo se describe la importancia de las marcas de agua en la sociedad, se detallan los aspectos básicos de las marcas de agua digitales, así como sus aplicaciones y propiedades. Finalmente, se describen brevemente las marcas de agua para video.

1.1. Importancia de las marcas de agua digitales

Los avances tecnológicos de los últimos años junto con el incremento de las redes de comunicación de banda ancha han propiciado la fácil creación, utilización y difusión del video digital. A diferencia del video analógico, el video digital no se degrada con el tiempo y es más fácil de manipular. Estas nuevas posibilidades benefician a un gran número de personas pues incrementa la forma de comunicar y expresar sus ideas. Desafortunadamente, esta gran flexibilidad tiene un revés: los propietarios pueden perder el control de sus videos después de ser distribuidos, es decir, pueden ser copiados y redistribuidos sin consentimiento con mayor facilidad. Este problema es comúnmente llamado *Piratería*.

Para evitar la piratería se han creado medidas legales y técnicas [Anó98, WIP96, DTSP08]. Una de ellas es la *Administración de Derechos Digitales (DRM)* (por sus

siglas en inglés), la cual utiliza tecnologías, que controlan el uso y la distribución de los medios, para proteger los derechos de autor (*Copyright*). Una tecnología DRM para videos usualmente utilizada es el Sistema de Codificación de Contenido (CSS), la cual consiste en cifrar el video mediante un mecanismo criptográfico y sólo los dispositivos regulados para la visualización conocen la forma de descifrarlo [BCK+99]. El problema de este tipo de protección es que una vez roto el sistema criptográfico deja al video sin protección, lo cual puede provocar su copia y la distribución sin restricciones. Aunado a esto, el video desprotegido queda con la misma calidad que el medio protegido.

Para que una técnica DRM sea considerada exitosa, la protección que provee debe ser *persistente*, es decir, la protección debe permanecer en cada copia del medio. Siendo así, las tecnologías DRM de cifrado pueden no ser persistentes, puesto que, como se mencionó, si se logra romper el mecanismo criptográfico el medio queda desprotegido. Para evitar esto, se están empezando a utilizar las técnicas de *marcas de agua digitales*.

Las técnicas de marcas de agua digitales describen métodos que ocultan un mensaje o marca dentro de un medio digital (video, imagen y audio, entre otros) generando un *medio marcado*. De esta forma, si el medio marcado es copiado y distribuido, el mensaje persistirá en cada una de las copias. Cabe destacar que el mensaje, en la mayoría de los casos, es imperceptible (invisible en caso de video e inaudible en caso de audio) e, idealmente, se puede decodificar del medio digital marcado incluso si éste es sometido a procesamientos que lo distorsionen.

En la actualidad, las marcas de agua se usan para proteger contenidos de video. Específicamente, la tecnología DRM llamada Sistema Avanzado de Acceso a Contenido (AACS), utilizada como un esquema de restricción de acceso y de copia para DVD de alta definición (HD-DVD) y discos *Blu-ray*, deja abierta la posibilidad de utilizar marcas de agua en audio para controlar la reproducción del video [Anó06]. Por ejemplo, si un reproductor de video (compatible con AACS) detecta una marca no compatible (p. ej. una marca insertada únicamente en el audio de un video de cine digital), entonces no reproduce el video (pues indicaría que el video es una copia de un filme exclusivo para cine). En este caso, mediante la utilización de marcas de agua se evita que el video se

utilice después de que la protección criptográfica se haya perdido, considerando que un video de cine digital está cifrado. Esta condición en donde la protección criptográfica se pierde al pasar de un medio digital a un medio analógico se le conoce como *hueco analógico*. Lo mismo sucede cuando un video, que originalmente está cifrado, se reproduce en un computadora y se copia mediante la captura digital automática de pantallas al reproducirlo. En este caso, al reproducir el video la protección criptográfica se pierde y éste queda desprotegido, a esta condición se la conoce como *hueco digital*, pues alude a la situación en donde existe una vulnerabilidad en la protección criptográfica.

La principal causa de todas estas medidas de protección de derechos de autor es económica. Las industrias encargadas de la producción y distribución de videos, incluyendo cines, tiendas de video y televisoras de pago por evento, pierden mucho dinero con la piratería. En el último reporte hecho por la Asociación Cinematográfica de Estados Unidos (MPAA), que fue presentado al congreso americano en 2005, se menciona que dichas industrias perdieron un poco más de $18 mil millones de dólares americanos (USD) debido a la piratería. Localmente en México, se estas pérdidas ascendieron a $1100 millones de USD. Adicionalmente a esta pérdida económica, se afectan a los creadores de los videos pues sus creaciones pueden ser modificadas sin su aprobación [MPA09].

Una nueva tendencia de la publicidad es la utilización de las marcas de agua para video usando la piratería para ganar dinero. De esta forma, si un video marcado es subido a un sitio de visualización de videos bajo demanda (como youtube, por citar alguno) y el servidor detecta la marca, entonces en lugar de evitar la reproducción al ser demandado, lo reproduce pero al mismo tiempo muestra publicidad, la cual generará ganancias al propietario del video. De esta forma se planea obtener una ganancia comercial de esta práctica ilegal [Rho07].

Actualmente, la utilización de marcas de agua para video está orientada al campo de la protección, no obstante la utilización de esquemas que oculten mensajes más grandes abre la puerta para el uso de las marcas de agua en aplicaciones emergentes de

multimedia enriquecida como Hipervideo[1], videos interactivos, entre otras [DDMP07]. El ocultar mensajes más grandes implica la utilización de esquemas de marcas de agua más complejos, principalmente para extraer correctamente el mensaje completo. Así, cuando los videos marcados, con mensajes grandes, son distorsionados temporalmente (p. ej. un intercambio de cuadros) la tarea de decodificación se torna compleja, pues el decodificador debe determinar dónde se han ocultado cada una de las partes del mensaje. Este problema no se presenta en esquemas de marcas de agua para video orientados a la protección, pues generalmente se oculta la misma marca en cada cuadro del video. En la sección 1.3 se describen generalidades de marcas de agua para video y se detalla el problema de las distorsiones temporales en videos marcados. En la siguiente sección se describen aspectos básicos de las marcas de agua mencionados a lo largo de este documento de tesis.

1.2. Generalidades de marcas de agua

La idea básica de ocultar un mensaje en un medio tiene como principal objetivo que el mensaje no sea detectado o reconocido por una tercera persona *enemiga* o *atacante*, en otras palabras, ocultar un mensaje importante en un medio inocuo para evitar levantar sospecha de tal ocultamiento. En marcas de agua digitales, además de ocultar imperceptiblemente el mensaje, éste se debe poder extraer de un medio marcado distorsionado, es decir, el mensaje debe ser difícil de separar del medio marcado. Es importante mencionar que existen aplicaciones donde la información agregada puede ser perceptible, pero son poco comunes [TKANR05].

De manera general, un *esquema de marcas de agua* consta de dos etapas: el *proceso de incrustación* y el *proceso de decodificación*, que se muestran en la figura 1.1.

En el *proceso de incrustación* (también llamado proceso de inserción o proceso de ocultamiento) se tiene el medio original X en donde se esconde un mensaje M usando

[1] El hipervideo es una secuencia de video que contiene enlaces incrustados, los cuales permiten la navegación entre el mismo video y otros elementos multimedia

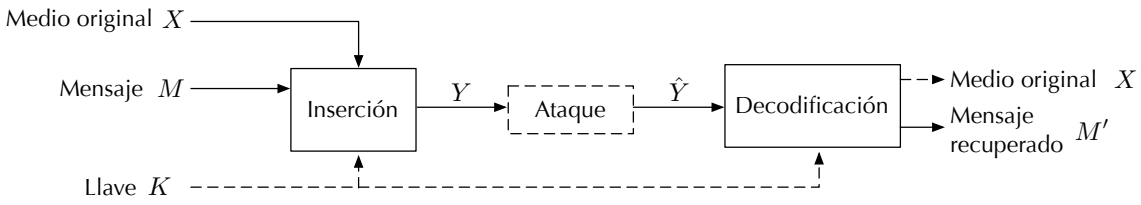


Figura 1.1: Diagrama general de un esquema de marcas de agua

un algoritmo de incrustación, el resultado de este proceso es un medio marcado Y. La diferencia entre el medio original y el medio marcado debe ser muy pequeña para evitar sospechas del atacante. En este proceso una *llave* K puede ser utilizada para producir una marca de agua más segura. En la práctica, la mayoría de los esquemas de marcas de agua son análogos a los sistemas criptográficos simétricos, en los que se emplea una sola llave para la codificación y decodificación, fundamentándose en dicha llave la seguridad del sistema [CMB+08]. Las líneas punteadas representan elementos que pueden o no existir dependiendo de un diseño particular. Es importante mencionar que el mensaje puede ser llamado marca de agua o simplemente marca.

Seguido del proceso de inserción, el medio marcado Y será enviado a un receptor por Internet o por cualquier otro medio. Cuando exista duda de los derechos de autor, de copia o cualquier otro evento relacionado con el medio a probar $\hat{Y}$, la marca será decodificada para recuperar y verificar los datos insertados. El medio a probar $\hat{Y}$ puede ser una versión atacada del medio marcado Y.

El *proceso de decodificación* es utilizado para obtener la marca escondida en un medio marcado, tal proceso puede utilizar el medio original o la llave según sea necesario. Es importante mencionar que el proceso de decodificación puede constar de dos procesos: el proceso de extracción y el proceso de detección. El proceso de extracción se encarga de obtener la marca de agua del medio marcado, mientras que el proceso de detección se encarga de identificar si existe una marca dentro de un medio. Es importante mencionar que existen esquemas de marcas de agua que pueden extraer, además de la marca escondida, el medio original X del medio marcado Y, dichos esquemas

son llamados *reversibles*. En esta tesis no serán abordados.

La importancia de los esquemas de marcas de agua recae en el hecho de que la marca insertada en el medio digital es persistente y, dependiendo la aplicación, puede ser transparente al sistema perceptivo humano.

La técnica para ocultar información más utilizada debido a la capacidad de poder detectar información de un medio marcado distorsionado es *espectro disperso* [CKLS97]. Básicamente, la incrustación de la marca usando esta técnica está definida por la ecuación 1.1:

$$Y = X + sW \tag{1.1}$$

En donde Y es el medio marcado, X es el medio original, s es el factor de robustez y W es la marca. La marca es una es una secuencia pseudoaleatoria de números reales con distribución normal $N(0, 1)$, es decir con media 0 y varianza 1. La experiencia otorgada por investigaciones en telecomunicaciones indica que la distribución normal debe ser utilizada para generar dicha seña pseudoaleatoria. Todo generador de pseudoaleatorio necesita una semilla como punto de partida para empezar a generar la sucesión pseudoaleatoria. Por construcción, un generador de secuencias pseudoaleatorias genera una sucesión diferente para cada semilla. En espectro disperso una secuencia pseudoaleatoria es generada usando un número semilla K, llamado *llave* o *llave de incrustación*. El factor de robustez s modifica la magnitud de la marca W que será añadida directamente al medio original. En la detección de la marca la misma secuencia pseudoaleatoria W es generada usando la llave K. Una función de comparación se utiliza con el objetivo de decidir la presencia de la marca incrustada W. Esto requiere una perfecta sincronización con el medio a probar $\hat{Y}$. Una de las funciones de comparación más utilizadas es la correlación lineal entre dos vectores x y y, donde los dos vectores tienen N elementos. Dicha función se muestra en la ecuación 1.2:

$$C(x, y) = \frac{1}{N} \sum_{i=1}^{N} x[i]y[j] \tag{1.2}$$

La detección de la marca se lleva a cabo usando el resultado c de la operación de correlación entre el medio a probar $\hat{Y}$ y la marca a detectar W. Si la correlación c es mayor a un umbral U, se entiende que la marca es detectada. En otro caso, la marca no fue detectada. En la ecuación 1.3, se muestra el proceso de detección usado en espectro disperso.

$$\begin{gathered} c = C(\hat{Y} \cdot W) \\ \text{si } c >= U\text{, la marca existe} \\ \text{si } c < U\text{, la marca existe} \end{gathered} \tag{1.3}$$

El espectro disperso es una técnica muy utilizada para diferentes tipos de medios debido a la alta robustez contra manipulaciones de señal. Si una llave secreta se utiliza para generar una secuencia pseudoaleatoria, este algoritmo no necesita el medio original para detectar la marca incrustada. La principal desventaja es la vulnerabilidad contra ataques de sincronización espacial. Además, la longitud del correlacionador debe ser suficiente para asegurar pequeñas probabilidades de errores.

Aplicaciones

Existen diversas aplicaciones donde se usan los esquemas de marca de agua [CMB+08, GH08]. A continuación se describen las más importantes:

Monitoreo de las transmisiones de radiodifusión. La información que se puede ocultar en un medio puede ser usada para identificar al propietario del medio o al comprador de una copia del mismo. Esta información puede ser utilizada por un sistema automatizado que rastrea las transmisiones de radiodifusión (como la televisión y la radio), las redes de computadoras u otros canales de difusión. De esta forma se puede llegar a prescindir de los observadores humanos o de la utilización de información adjunta como es el caso de los datos *Close caption*[2] en la televisión para la detección de comerciales por ejemplo. De esta manera, los

[2]Close caption es el nombre en inglés del sistema de subtítulos para programas de televisión y video destinado a permitir que las personas sordas o con dificultades para captar la señal de audio, puedan comprender lo que se dice en televisión o en los videos.

esquemas de marcas de agua son una alternativa real para ocultar información de identificación para monitoreo. Las principal ventaja es que la información insertada es persistente, además de ser compatible con el equipo de radiodifusión utilizado. La principal desventaja es que el proceso de incrustación de la marca es más complejo que modificar los datos Close caption o metadatos del video transmitido. Además, se debe considerar que la información que se oculta puede degradar la calidad visual o auditiva del medio.

Protección de derechos de autor. Para proteger la propiedad intelectual de un medio, el poseedor del mismo puede insertar una marca de agua que represente la información de los derechos de autor de éste. El impacto legal de este tipo de sistema no ha sido aprobado en una corte. Es importante mencionar que existen alternativas para identificar si en un medio marcado se hizo un ataque de inserción no autorizada, es decir se ocultó nuevamente información de propiedad. Generalmente se utilizan herramientas criptográficas, como firmas digitales del medio.

Control de distribución de copias. Para encontrar el origen de copias ilegales, el propietario puede usar la técnica conocida como *Fingerprinting* o seguimiento de transacciones. Mediante el uso de esquemas de marcas de agua se puede incrustar información que sirva para identificar a cada copia de un mismo medio, es decir la marca de agua representa información asociada al consumidor que adquirió el medio. Esto es análogo a una huella digital humana que identifica a una persona. Esta técnica posibilita la identificación del consumidor que ha violado la licencia que se le otorgó, por ejemplo al distribuir el trabajo intelectual a terceras partes.

Autenticación de contenido. En este caso es posible usar marcas de agua frágiles, esto es, sensibles de perderse al sufrir cualquier ataque o procesamiento por mínimo que este sea. Una marca de agua frágil indica cuando el trabajo intelectual ha sido alterado y la manera en que se alteró.

Protección contra copias ilegales. Con una marca de agua es posible directamente controlar dispositivos de grabación digitales para fines de protección contra copias. En este caso si el dispositivo de grabación detecta la marca entonces no realiza la copia del trabajo intelectual.

Comunicaciones secretas. En esta aplicación la marca incrustada en los archivos multimedia se utiliza por dos o más personas para comunicarse secretamente sin levantar sospecha en terceros. En el caso de videos es necesario que el esquema de marcas de agua sea robusto pues la mayoría de videos están codificados con algún algoritmo de compresión.

Metadatos adicionales. La utilización de esquemas marcas de agua que oculten más información abre la puerta para el uso de las marcas de agua en aplicaciones emergentes de multimedia enriquecida como Hipervideo, videos interactivos, subtítulos, entre otras [DDMP07].

Propiedades

Los esquemas de marcas de agua pueden ser caracterizados por un número definido de propiedades. La relativa importancia de cada propiedad depende de los requerimientos de la aplicación y el rol que tomen las marcas de agua. A continuación describiremos las propiedades asociadas al proceso de incrustación de la marca: efectividad, fidelidad y carga útil de la marca. Después, las típicamente asociadas a la detección: detección ciega o informada, probabilidad de error y robustez. Y finalmente la propiedad de seguridad.

Efectividad. Un medio marcado se define como un medio en donde el detector ha detectado una marca. Con esta definición, la efectividad de un esquema de marcas de agua es la probabilidad de detección después de una incrustación. La efectividad puede determinarse analíticamente o mediante una estimación empírica mediante la incrustación de una marca en un conjunto grande de imágenes de

prueba. El porcentaje de imágenes marcadas que tengan una detección positiva aproximará la efectividad del sistema. Es importante mencionar que las imágenes utilizadas deben tener la misma distribución que las esperadas en la aplicación.

Fidelidad. En general, la fidelidad de un esquema de marcas de agua se refiere a la similitud perceptual entre el medio original y el medio marcado en el punto en el que son presentados a un consumidor. Para disminuir el impacto perceptual que genera la incrustación de la marca en un video se han tomado en cuenta modelos del sistema visual humano.

Carga útil de la marca. La carga útil de la marca se refiere a la cantidad de información que puede ser incrustada en una medio. Esta propiedad es llamada también capacidad. La carga útil depende de la aplicación; en control de distribución de copias normalmente son necesarios 2-4 bits para cada 5 minutos de video. En contraste, en algunos escenarios de monitoreo de difusión de televisión son necesarios 24 bits por segundo.

Decodificación ciega o informada. En algunas aplicaciones, como protección de derechos de autor y monitoreo de datos, los algoritmos de decodificación pueden usar el medio original para extraer el mensaje, aquí se dice que es un esquema informado, no ciego o dependiente del medio original. En aplicaciones como protección contra copias los algoritmos de decodificación no tienen acceso al medio original, esto provoca que la extracción del mensaje sea más compleja. Este tipo de esquemas son conocidos como públicos, ciegos o no dependientes del medio original.

Probabilidad de error. Puesto que es importante saber si un medio está marcado o no, la probabilidad de error al detectar una marca debe ser muy pequeña. Existen dos tipos de errores que se presentan al detectar una marca. El primero, llamado *falso negativo* se presenta cuando el decodificador no detecta la marca en un medio que tiene incrustada la misma marca. El otro error se llama *falso positivo* y se

presenta cuando el decodificador detecta una marca en un medio no marcado con dicha marca. La probabilidad de ocurrencia de dichos errores es un punto fundamental para determinar si un esquema de marcas de agua se utiliza en una determinada aplicación.

Robustez. La robustez se refiere a la capacidad de detectar la marca después de aplicar una técnica de procesamiento (ataque no intencional) o un ataque intencional en el medio marcado. Algunos ejemplos de procesamientos no intencionales en videos son: compresión con pérdidas, cambio de razón de cuadros por segundo, conversión analógico-digital (A/D) y digital-analógico (D/A), filtrado, adición de ruido, cambio de razón de bits por segundo o *bitrate*[3], entre otros. No todas las aplicaciones de marcas de agua requieren ser robustas ante todos los posibles procesamientos, es decir, la robustez ante algún procesamiento es dependiente de la aplicación. En algunos casos, la robustez es irrelevante o no se toma en cuenta. Existe una forma de esquemas de marcas de agua llamada *frágil*, estos tipos de esquemas están diseñados para no ser robustos ante distorsiones. Por ejemplo, un esquema de marcas de agua diseñado para autenticación debe ser frágil, es decir, cualquier distorsión en el medio marcado provocará que la marca se pierda, puesto que una falla en el proceso de detección muestra que el medio ha sido alterado y no es auténtico. Por otro lado, existen los intentos de remoción intencional de la marca o análisis intencionales para estimar la marca incrustada mediante un ataque de confabulación (o colusión) entre propietarios de copias de medios marcados. En general, idealmente, no debiera existir forma alguna de remover o alterar la marca sin degradar suficientemente la calidad perceptual del medio haciéndolo inutilizable.

Seguridad. La seguridad de los esquemas de marcas de agua puede ser interpretada como se hace en las técnicas de cifrado. Una de las suposiciones de Kerckhoffs

[3]El término bitrate, en codificación de video, es el número de bits máximo que debe ocupar una secuencia de video con una duración de tiempo fija (por lo general un segundo) después de ser codificada.

establece que se debería asumir que el método usado para cifrar/descifrar el mensaje es conocido por terceras partes y que la seguridad radica en el conocimiento de la llave [MvOV96]. Así, un esquema de marca de agua es seguro si el conocimiento de los algoritmos de inserción y extracción no ayuda a detectar o remover la marca incrustada.

1.3. Marcas de agua para video

Un video puede ser considerado como una secuencia de imágenes del mismo tamaño, llamadas cuadros, con una separación en el tiempo. Normalmente se confunde la definición de video con la de formato contenedor, siendo éste último un formato de archivo que almacena información de video, audio, subtítulos, metadatos, etc. Ejemplos de contenedores multimedia son avi, mpg, mov, etc. Así, un archivo ".avi" representa un contenedor en donde están almacenados flujos de video y de audio. Generalmente, estos flujos de video y de audio están comprimidos pues en ellos existe una enorme redundancia de información.

Tomando en cuenta que, en la mayoría de los casos, no es práctico tener los videos sin compresión (videos planos), un esquema de marcas de agua para video debe ser robusto ante compresión con pérdidas, principalmente ante compresiones MPEG puesto que la mayoría de los videos comprimidos utiliza esta compresión.

El MPEG es un grupo de trabajo del Organización Internacional para la Estandarización/Comisión Electrotécnica Internacional (ISO/IEC) encargado de desarrollar estándares de codificación de audio y vídeo. Dicho grupo es el creador de una familia de estándares de compresión de audio y video que dominan el mercado mundial: MPEG-2 y MPEG-4.

Los estándares MPEG-2 definen estándares de audio y vídeo para difusión de Televisión digital terrestre, por satélite o cable. Con algunas modificaciones, es también el formato de codificación usado por los discos SVCD y DVD comerciales de películas. Los estándares que conforman a MPEG-2 son llamados partes. MPEG-2 parte 2

define el estándar de compresión de video. A partir de este momento cuando se haga referencia a MPEG-2 se referirá específicamente a la parte 2 del estándar MPEG-2.

Los estándares MPEG-4 definen estándares de audio y vídeo basados en el estándar MPEG-2 y del estándar Apple QuickTime. De esta manera, MPEG-4 toma muchas de las características de MPEG-2 así como de otros estándares relacionados, tales como soporte del Lenguaje de Modelado de Realidad Virtual (VRML) extendido para Visualización 3D, archivos compuestos en orientación a objetos, soporte para la gestión de Derechos Digitales externos y variados tipos de interactividad. Todas estas características están definidas en cada una de las partes que lo conforman. Es de especial interés para esta investigación las partes 2 y 10 de MPEG-4, pues ahí se definen dos estándares de compresión de video utilizados ampliamente en el mundo. MPEG-4 parte 2 es usado por los principales *codecs*(codificador/decodificador) del mercado entre los que encontramos: DivX, Xvid, Nero Digital y 3ivx. Por otra parte MPEG-4 parte 10 (MPEG-4 AVC ó H.264) es usado por los codecs x264, Nero Digital AVC, Quicktime 7 y por discos de video de alta resolución Blu-ray. Al final de la sección 3.2 se describe el funcionamiento de los estándares de compresión MPEG-2, MPEG-4 parte 2 y 10.

Es importante mencionar que las compresiones MPEG son con pérdida, es decir que el video comprimido no representa al video plano original en su totalidad. Esta pérdida de información debe ser considerada al diseñar esquemas de marcas de agua para video.

Una de las clasificaciones de los esquemas de marcas de agua para video se define de acuerdo al momento en que se hace el marcado. De esta forma existen esquemas que ocultan información:

1. Directamente en el video plano. Se toma el video como una serie de cuadros sin compresión. Este tipo de esquemas es independiente de la compresión de video utilizada. Ejemplos de este tipo de esquemas están descritos en [KDHM99, HG06, CM08, DHZQ08].

2. Durante la compresión del video. Normalmente se utilizan los procesos inter-

nos de un algoritmo de compresión de video para ocultar la marca. Es común la utilización de los coeficientes de la Transformada Discreta de Coseno (DCT) ([CHO+98, WZWP05, CgGzL08]) y los vectores de movimiento, utilizados por las codificaciones MPEG, para esconder la marca ([hZXjZjS07, KWP08]). Esta clase de esquemas son más eficientes que los esquemas que ocultan la marca directamente en el video plano, lo que los hace útiles para aplicaciones en tiempo real. No obstante, son fuertemente dependientes del codificador utilizado, es decir, si el video se recodifica con otro codificador la marca se pierde. Debido a esta característica son considerados dependientes del método de codificación.

3. Durante la difusión. Este tipo de esquemas ocultan la marca dentro del flujo de bits generado por un algoritmo de *Streaming* (streaming consiste en la distribución de video por Internet). Unos ejemplos de este tipo de esquemas se encuentran en [HG97, LSH06], ellos incrustan la marca en el flujo de bits generado por MPEG (el flujo de bits consiste en información de encabezados, vectores de movimiento y bloques DCT codificados). Dichos esquemas primero decodifican los bloques DCT (del flujo de bits), luego los modifican y los vuelven a codificar, generando así un nuevo flujo de bits marcado. En [KKHW07] se describe un método que oculta la marca modificando el codificador utilizado para la generación de flujo de bits MPEG-4 parte 10.

Un problema común al diseñar esquemas de marcas de agua para video es la resistencia ante distorsiones en el video marcado, es decir que el decodificador pueda encontrar la marca en un video marcado que ha sido distorsionado. Las distorsiones que más afectan en la detección de la marca son las aquellas que provocan *pérdida de sincronización*.

La *sincronización* es el proceso de identificar la correspondencia entre las coordenadas del medio marcado y las coordenadas de la marca incrustada. De esta manera, si la marca se incrusta en una región de un cuadro y el cuadro es rotado posteriormente, el detector de la marca no podrá encontrar la marca porque existió una pérdida de sincro-

nización espacial (ver figura 1.2), esto aplica también a ataques en el tiempo (ver figura 1.3). Este tipo de Ataques temporales no eliminan la marca, sólo hacen que el detector quede imposibilitado para encontrarla.

Figura 1.2: Cuadro marcado y después de un ataque espacial de rotación

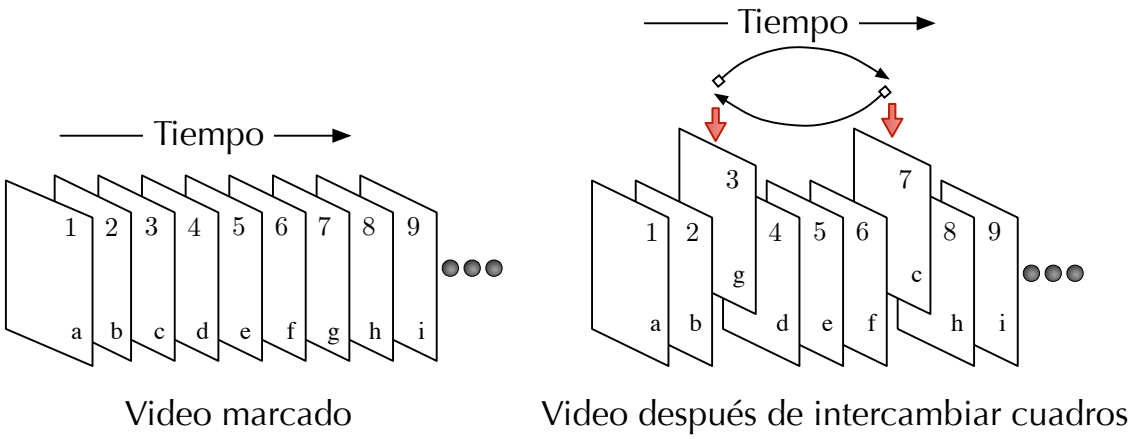


Figura 1.3: Video marcado y después de un ataque temporal de intercambio de cuadros

Ejemplos de ataques de sincronización espacial en video incluyen el cambio de formato de presentación (p. ej. cambios de 16:9 a 4:3) y cambio en la resolución espacial (p. ej. cambios de NTSC a PAL o viceversa) [DD03]. En [CM08] se propone un método que soporta ligeras distorsiones geométricas (rotación y escalamiento), recortes

y compresión con pérdida mediante el uso de la Transformada Wavelet Compleja de Árbol Dual (DT CWT).

Los ataques de desincronización temporal o *Ataques temporales* son aquellos que cambian el orden o la cantidad de cuadros provocando modificaciones temporales en el video. Dichos ataques son un reto cuando la marca a incrustar no es la misma para todos los cuadros en el video, puesto que los datos ocultos se pueden desincronizar con una operación simple, como por ejemplo intercambio de cuadros (figura 1.3). La sincronización temporal es la piedra angular para esquemas que necesitan más de un cuadro para ocultar un mensaje.

1.4. Objetivos

Se ha mencionado que la tendencia en el campo de las marcas de agua para video es la utilización de esquemas que oculten mensajes más grandes, pues se abre la puerta para su uso en aplicaciones emergentes, como la multimedia enriquecida. El problema fundamental de esconder mensajes grandes es que los ataques temporales provocan la pérdida de sincronización temporal. Si hay pérdida de sincronización el mensaje no puede decodificarse. Además, hay que tomar en cuenta que todo video está comprimido, por lo tanto los esquemas de marcas de agua deben soportar ataques de compresión. De esta manera, los objetivos de este trabajo de investigación se describen a continuación:

Objetivo general

El objetivo general de la investigación es:

Diseñar un esquema de marcas de agua para video, robusto ante ataques de pérdida de sincronización temporal y ante ataques de compresión MPEG-2 y MPEG-4.

Alcances del objetivo:

- Los ataques de pérdida de sincronización temporal que serán utilizados para medir la robustez del esquema que se propondrá son: inserción de cuadros no marcados, eliminación de cuadros, intercambio de cuadros, decimación de cuadros, promedio de cuadros y cambios de razón de cuadros por segundo, para más detalle acerca de dichos ataques ir a las sección 2.1.

- MPEG-4 permite la composición de objetos audio-visuales, en el caso de la presente investigación se considera que la secuencia de video está compuesta por un sólo objeto, es decir el video mismo. Las codificaciones que MPEG-4 ofrece son: el MPEG-4 parte 2 y el MPEG-4 parte 10, estas dos codificaciones serán utilizadas para medir la robustez del esquema ante compresión MPEG-4. Para más detalle acerca de MPEG-4 ir al final de la sección 3.2.

Objetivos específicos

- Diseñar un método que evite la pérdida de la sincronización temporal entre el detector de la marca y el mensaje incrustado en un video. Este método debe ser robusto ante ataques temporales incluyendo los llevados a cabo por las codificaciones MPEG-2 y MPEG-4.

- Crear un Benchmark de ataques temporales para medir el rendimiento del esquema propuesto.

1.5. Estructura de la tesis

Esta tesis está estructurada de la siguiente manera:

Capítulo 2. En este capítulo se describe el problema de la sincronización temporal y el estado del trabajo previo de los principales esquemas de marcas de agua que resisten ataques temporales.

Capítulo 3. Se presenta el diseño general del benchmark de ataques temporales propuesto. Además, se describe el funcionamiento de los ataques temporales y los ataques de compresión MPEG.

Capítulo 4. En este capítulo se describe el esquema de marcas de agua para video propuesto. Se detallan los procesos internos de los algoritmos de incrustación y de decodificación. Finalmente, se analiza el comportamiento del esquema propuesto ante los principales ataques temporales y de compresión.

Capítulo 5. En este capítulo se muestran los resultados obtenidos al evaluar al esquema propuesto para medir su funcionamiento, fidelidad, robustez y capacidad de inserción.

Capítulo 6. En este último capítulo se presentan las conclusiones obtenidas, las aportaciones y el trabajo futuro.

Capítulo 2

Sincronización temporal

En este capítulo se describe el problema de la pérdida de sincronización temporal y los ataques que la ocasionan. Adicionalmente, se presentan las principales estrategias utilizadas por los diseñadores de marcas de agua para que sus esquemas detecten la información incrustada en videos distorsionados temporalmente. Finalmente, se muestra el estado actual de la investigación previa de los esquemas de marcas de agua para video resistentes ante dichos ataques.

2.1. Sincronización en el tiempo

La Sincronización es el proceso de identificar la correspondencia entre las coordenadas de la señal marcada y las coordenadas de la marca incrustada. De esta forma, la sincronización temporal consiste en encontrar en dónde se ocultó la marca a través del tiempo. Como se mostró en la figura 1.3, un intercambio de cuadros provoca que el detector quede imposibilitado para hallar la marca correspondiente en la posición dentro del video en la cual se insertó. Debido al problema de la pérdida de sincronización temporal muchos autores prefieren ocultar la misma marca en cada cuadro, pero al hacer esto, no se explota la dimensión temporal que ofrecen los videos, es decir no varían la marca incrustada a lo largo del video.

La sincronización es un proceso crítico en la fase de detección de los esquemas robustos de marcas de agua para video, incluso si este medio no es atacado maliciosamente. Muchos de los procesamientos, no maliciosos, a los que son sometidos los videos están relacionados con procesos de compresión. El proceso de compresión puede provocar pérdida de sincronización temporal porque durante el proceso para mantener el bitrate requerido se pueden eliminar cuadros. La eliminación de cuadros evidentemente distorsiona temporalmente al video.

Ataques de desincronización

Los principales requerimientos de robustez de los esquemas de marcas de agua para video son: compresión de video, mejoramiento de la señal, adición de ruido, conversión A/D y D/A, conversión de formato de video, ataques de Colusión y ataques temporales [EUT+01]. La mayoría de esos requerimientos de robustez son ataques espaciales, es decir, ataques que modifican los cuadros separadamente como imágenes. Los ataques temporales cambian el orden o la cantidad de cuadros provocando modificaciones temporales en el video. Estas modificaciones temporales deben ser consideradas cuando el mensaje a ocultar necesita más de un cuadro para ser incrustado en el video, puesto que una cambio en el orden en que los cuadros fueron marcados es fatal durante la decodificación del mensaje. Para resistir este tipo de ataques se requiere de una fase de sincronización temporal que ayude al decodificador a encontrar el mensaje.

Los principales ataques que pueden causar pérdida de sincronización temporal son:

1. Inserción de cuadros no marcados: se insertan cuadros no marcado en el video original.
2. Eliminación de cuadros: se eliminan cuadros del video original.
3. Intercambio de cuadros: se cambia de posición un cuadro o se intercambian dos cuadros a la vez del video original.
4. Decimación de cuadros: se mantiene un cuadro por cada n cuadros consecutivos, este ataque es un ataque de eliminación consecutiva.
5. Promedio de cuadros: se sustituyen cuadros del video original por cuadros promediados.

6. Cambio de *Framerate* o cambio de Cuadros por segundo (FPS): Dado un video original con duración de t segundos, se cambia el número de cuadros que serán presentados por segundo manteniendo el mismo tiempo t.

7. Combinación de los anteriores

8. Compresión de video: es usada para disminuir el tamaño del video original.

En el capítulo 3 se describen a detalle los ataques mencionados anteriormente.

2.2. Estrategias utilizadas

Para llevar a cabo la sincronización temporal, los diseñadores de esquemas de marcas de agua para video han utilizado diversas estrategias, las cuales pueden dividirse en aquellas que:

- Realizan una búsqueda exhaustiva
- Usan plantillas
  - Ocultan una señal de sincronización
  - Aplican restricciones a la forma de la marca
  - Usan características del video

En la sincronización por búsqueda exhaustiva, el decodificador busca la marca en el espacio posible de las transformaciones a la que ésta se pueda someter. Por ejemplo, se puede intentar detectar una marca en un cuadro atacado al buscar versiones rotadas de la marca en dicho cuadro. Esta búsqueda genera una resistencia ante ataques de rotación pues es posible que alguna versión rotada de la marca se detecte en un cuadro rotado. El problema de usar este tipo de medida de sincronización es el costo que se requiere para analizar todas las combinaciones de búsquedas. Sin embargo, existen esquemas que limitan el tamaño del espacio de búsqueda, para disminuir el costo [HG98]. No obstante, se ha probado experimentalmente que la búsqueda exhaustiva es propensa

a generar falsos positivos [LSKL03]. Además, dicha búsqueda es realizada sin usar propiedades de la marca o del medio.

Otro enfoque para la sincronización es mediante el uso de plantillas de sincronización. Una plantilla es un patrón que describe la localización o parámetros usados para extraer la marca incrustada en el mismo medio. Cabe destacar que los ataques que sufra el medio marcado afectarán a la plantilla y a la marca por igual. Por lo anterior, las plantillas deben tener la misma robustez que la marca incrustada. Existen tres métodos para generar plantillas de sincronización, los cuales se describen a continuación.

El primer método para generar plantillas consiste en ocultar una señal de sincronización en el medio marcado, es decir, una marca auxiliar que ayude a localizar la marca original. Existen varios esquemas de marcas de agua para video en la literatura [DCRP99, BKS01, LMT02, NSB02, GLM05] que usan este tipo de medida de sincronización .

En el segundo método se manipula la forma de la marca para generar un patrón de sincronización. La construcción de una serie de marcas (llamada patrón) donde cada una indique el bit a incrustar en el tiempo es utilizada en [KDHM99]. La estructura del patrón provee la plantilla, por lo cual, no es necesario incrustar otra marca auxiliar. La capacidad de información del método está dada en función de la forma del patrón a utilizar.

El tercer método para generar plantillas de sincronización utiliza las características del video. Este método analiza el video para encontrar características que resalten. A diferencia de los otros métodos no se necesitan señales adicionales, ni someter la marca a un patrón especial para que sea utilizada como plantilla. Ejemplos de esquemas que realizan este tipo de sincronización son encontrados en [HKM05, SLY06, YG06, CgGzL08]. En [KWP08, hZXjZjS07], la selección de vectores de movimiento son la base para la incrustación de la marca. Desafortunadamente estos esquemas son dependientes del método de generación de vectores de movimiento. Los puntos característicos detectados en cada cuadro sirven de base para ocultar información [BCM02]. Considerando que un mensaje es una colección de marcas y que cada marca es generada

usando una llave (número), la creación de llaves tomando en cuenta las características de los cuadros permite llevar a cabo la sincronización temporal; ejemplo de esquemas que lo hacen así son los reportados en [LD02, LD04, DL05, DHZQ08].

2.3. Trabajos relacionados

Hasta el momento se ha mostrado una amplia variedad de esquemas de marcas de agua para video, pero son pocos los esquemas que realizan una etapa de sincronización temporal. En la tabla 2.1 se muestran los esquemas de marcas de agua para video más significativos, en términos de robustez, que realizan sincronización temporal.

Se puede observar que los esquemas mostrados miden su rendimiento ante algunos ataques temporales. Los ataques temporales que reportan se muestran en la tercera columna. Es importante notar que los ataques de inserción y eliminación son los ataques que más se usan para reportar el rendimiento de un esquema de marcas de agua. Por otra parte el ataque de decimación es reportado por solo un trabajo. Existe una carencia en la selección de ataques temporales que son usados para medir el rendimiento de un esquema de marcas de agua para video, este problema es abordado en el siguiente capítulo. Finalmente, se puede observar también que el esquema de Lin reporta resistencia ante una mayor cantidad de ataques temporales y que los ataques más reportados son inserción y eliminación de cuadros. Es importante mencionar que el esquema de Guerrini genera una plantilla de autenticación para video que no incrusta información, por lo que está marcado con un *. Dicha plantilla puede ser usada por esquemas de marcas de agua para soportar ataques temporales. Los esquemas descritos en la tabla se describen brevemente a continuación.

El esquema propuesto por Sun *et al.* en [SLY06] es un esquema dependiente del contenido del video. En él, el video se segmenta en escenas, luego se extraen características de cada una mediante el Análisis en dos Dimensiones de los Componentes Principales (2DPCA). Posteriormente, la marca es incrustada en esas características

Esquema	Capacidad	Ataques temporales probados
Sun [SLY06]	32×32 píxeles	Intercambio, eliminación, cambio de framerate
Lin [LD02, LD04, DL05], Dong [DHZQ08]	15 bits	Inserción, eliminación, decimación, intercambio y promedio de cuadros
Guerrini [GLM05]*	_	Eliminación, intercambio de cuadros
Yang [YG06]	64 bits	Inserción, eliminación, cambio de framerate
Zhang [hZXjZjS07]	-	Inserción, eliminación
Kezheng [KWP08]	32×32 píxeles	Inserción, eliminación
Chen [CgGzL08]	-	Inserción, eliminación

* genera una plantilla de autenticación para video que no incrusta información.

Tabla 2.1: Comparación de los principales esquemas de marcas de agua que contemplan la sincronización temporal

(p. ej. el fondo de la escena). La detección se lleva a cabo de igual manera que la incrustación, con la diferencia de detectar la marca en lugar de incrustarla. Este esquema no crea un método especial para controlar la sincronización temporal, sólo se basa en las propiedades invariantes de las características obtenidas. Debido a que este esquema extrae información de una sucesión de cuadros, tiene problemas con el ataque de inserción de cuadros.

A diferencia del esquema de Sun, el esquema de Lin *et al.*, propuesto en [LD02, LD04, DL05], lleva a cabo sincronización temporal mediante la manipulación de la llave de incrustación. La llave se cambia cada determinados cuadros utilizando las características de un cuadro particular y la misma llave. De esta forma, la marca (generada con una llave) que se incrusta en cada cuadro lleva implícitamente información temporal. En la figura 2.1 se puede apreciar la redundancia generada por el esquema de Lin. Los cuadrados con el mismo relleno indican que la misma llave fue usada para ocultar la marca. Se puede notar que cada 3 cuadros la llave es cambiada y cada 12 cuadros la llave es restablecida. Después de restablecer la llave, el esquema de Lin toma las características del último cuadro marcado para generar la siguiente llave, por ejemplo,

toma las características del cuadro 2 para generar la llave que se usará para marcar los cuadros 3, 4 y 5. Esto provoca que las llaves generadas después de un restablecimiento sean diferentes entre si, se puede apreciar en la figura que las llaves de los cuadros 3, 4 y 5 no son las mismas que la de los cuadros 15, 16 y 17, puesto que se generaron usando características de cuadros diferentes. En este ejemplo, si en cada cuadro se oculta un bit entonces la cantidad de bits incrustados es 4.

La tarea de sincronización se lleva a cabo en la detección. Aquí, cuando una marca es detectada se genera la siguiente y se almacena en una cola. Al analizar el siguiente cuadro se buscan las marcas almacenadas previamente. Una modificación en el manejo de la cola se presenta en [DHZQ08], en este trabajo se mejora la robustez del esquema ante el ataque de intercambio de cuadros un 13.6 %. En el esquema de Lin, la dependencia de un sólo cuadro en la generación de llaves puede ocasionar problemas en la detección. No obstante, el rendimiento de dicho esquema ante ataques temporales es el mejor comparado con el rendimiento de los otros esquemas mencionados en este capítulo. Es importante mencionar que el esquema de Lin es la base del esquema que se propone en esta investigación.

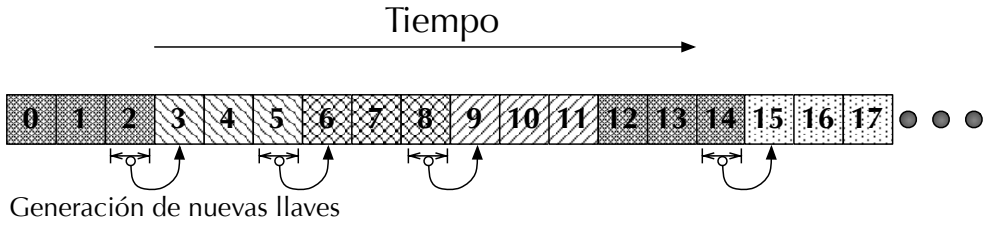


Figura 2.1: Redundancia generada por el esquema de Lin

El esquema propuesto por Guerrini *et al.* en [GLM05] genera una plantilla de autenticación para video que no incrusta información (por eso está marcado con * en la tabla 2.1). Esta plantilla puede ser utilizada por algún esquema de marcas de agua para ayudar a mantener la sincronización temporal. No obstante, la utilización de una plantilla como información adicional puede limitar las aplicaciones en donde se pueda utilizar, por ejemplo, esto no es útil en aplicaciones donde se desee un bajo costo computacio-

nal, pues el uso de una plantilla de sincronización aumenta el costo computacional. Este esquema genera un muestreo en burbujas, como se puede apreciar en la figura 2.2, que incluye la dimensión temporal. Cada burbuja 3D consiste en una secuencia de burbujas 2D, llamadas burbujas intra-cuadro. De estas burbujas se extraen histogramas y se realiza un Hashing[1] de esta información. Este hashing representa la plantilla de autenticación. Este esquema no es resistente a cambios espaciales puesto que cada burbuja es generada por medio de la especificación del usuario (el usuario genera el número, disposición y posición de las burbujas).

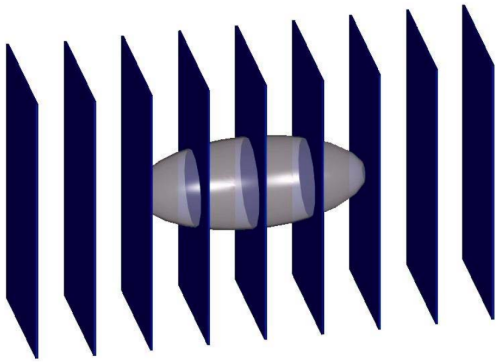


Figura 2.2: Burbuja elíptica 3D dentro de un video, los cuadros son representados por rectángulos

Por su parte, el esquema de Yang *et al.* soporta ciertas distorsiones espaciales debido a la modificación del promedio de los componentes de frecuencia media de la transformada DCT entre cuadros contiguos [YG06]. El esquema parte de que la media de dichos componentes entre cuadros adyacentes es muy cercana. De esta forma, se toma un cuadro y se calcula la diferencia con respecto a sus dos cuadros adyacentes. La incrustación de la marca se lleva a cabo modificando los componentes de frecuencia media de los cuadros adyacentes, para provocar que dicha diferencia sea menor o mayor a cero. Para incrustar un bit es necesario modificar tres cuadros. La desventaja principal de este esquema es que no oculta un mismo bit redundantemente. Esto provoca que

[1] función o método para generar claves o llaves que representen de manera casi unívoca a un documento, registro, archivo, etc.

si un bit no se detecta correctamente se pierda la sincronización temporal. Además, el esquema presenta una falla de seguridad en la detección de la marca en el sentido de que se puede extraer información en videos no marcados. Lo anterior sucede porque en el proceso de detección se realiza un cálculo del error de predicción entre cuadros consecutivos (éste indica si la diferencia del cuadro central con los adyacentes es mayor o menor que 0), el cual se compara con un umbral para obtener la marca (compuesta por 0s, 1s y -1s). Se puede notar que no hay una medida para saber si el video fue marcado o no. Además, se parte del hecho de que los cuadros no van a ser cambiados de posición, es decir, los bits incrustados no van a ser intercalados. Adicionalmente, en este esquema se oculta todo el mensaje dentro de cada escena. Así, la detección de una escena sirve como un punto de resincronización por si el mensaje no fue detectado.

Como se ha mostrado, a diferencia del esquema de Yang *et al.*, la mayoría de los esquemas que resuelven el problema de sincronización temporal dejan a un lado la sincronización espacial, pues una etapa de sincronización temporal es compleja. Pese a que el esquema de Yang tiene resistencia ante ataques espaciales, éste no tiene una etapa de sincronización temporal. En los experimentos reportados sólo se realiza una correlación entre la marca extraída y la insertada para comprobar la efectividad del método ante los ataques temporales de eliminación e inserción de cuadros. Es importante mencionar que el rendimiento del dicho esquema ante el ataque de intercambio de cuadros no es mencionado, puesto que la utilización de segmentos de tres cuadros para ocultar un bit de información lo hace susceptible ante este tipo de distorsiones. Se reporta también la resistencia ante compresión MPEG-1.

El esquema propuesto por Chen en [CgGzL08] oculta la información en los componentes DCT de cada cuadro del video; para tener una resistencia ante ataques temporales oculta una marca en cada escena. Esta dependencia en escenas provoca que dicho esquema no soporte ataques de intercambio de cuadros.

Los esquemas descritos en [KWP08, hZXjZjS07] ocultan la información al modificar ligeramente las magnitudes de los vectores de movimiento de toda una escena. Para ello realizan una clasificación de vectores de movimiento en diversos grupos, lue-

go, dependiendo de una cierta regla, se incrusta un bit de información en un grupo. De esta forma se oculta un bit dentro de un grupo de vectores con características similares. Los ataques temporales soportados por estos esquemas son inserción y eliminación de cuadros. Nótese que ésta es una tarea interna propia de la compresión MPEG, y por ello estos esquemas presentan robustez ante dicho ataque. Esto también es una desventaja puesto que si el video es recodificado (usando otra compresión o modificando los parámetros de la compresión MPEG actual), la marca se perderá.

En conclusión, se puede generalizar que para soportar ataques temporales y de compresión es necesario ocultar información redundantemente en el video. Por ejemplo, el esquema propuesto por Coria en [CM08] incrusta la misma marca en 30 cuadros seguidos para poder soportar un ataque de codificación MPEG-4. Hay que hacer notar que Coria no diseñó su esquema para soportar ataques temporales, pero él toma en cuenta que la mayoría de los videos están codificados y por ende su esquema debe soportar ataques de codificación.

2.4. Discusión del capítulo

En este capítulo se definió la sincronización temporal. Básicamente, la sincronización temporal es un proceso del decodificador que se encarga de buscar en donde se escondió la marca dentro de un video para poder extraerla. Se describieron las diversas estrategias que los diseñadores de marcas de agua han utilizado para que sus esquemas puedan resistir ataques temporales. Las estrategias mencionadas son: el incrustar una marca adicional de resincronización, modificar la forma de la marca incrustada e incrustar la marca usando características del video.

Además, se mencionaron los principales esquemas de marcas de agua que resisten ataques temporales y la forma en que llevan a cabo esta labor. De los esquemas descritos, el de Lin *et al.* es el que muestra un mejor comportamiento ante tales ataques, ya que puede determinar si parte del mensaje fue atacado. Además, si detecta un cambio temporal en el video y se interrumpe el proceso de detección puede resincronizarse pa-

ra intentar detectar el mensaje nuevamente. Este esquema es la base del esquema que se propone en esta investigación. Finalmente, es importante mencionar que todos los esquemas que resisten ataques temporales y que se describieron en este capítulo miden su desempeño ante diversos de ataques temporales, esto se debe a que no existe un Benchmark para esquemas de marcas de agua para video que incluya pruebas ante ataques temporales. En el siguiente capítulo se describe detalladamente esta situación.

Capítulo 3

Benchmark de ataques temporales

En este capítulo se detalla el diseño de un nuevo benchmark de ataques temporales, llamado VidMark, para permitir la comparación justa de los esquemas de marcas de agua para video. Este benchmark será utilizado posteriormente para medir el rendimiento del esquema propuesto en esta tesis.

3.1. Justificación

En la actualidad, la investigación en marcas de agua crece rápidamente, lo que provoca que se desarrollen cada vez más esquemas de marcas de agua. La forma de evaluarlos y compararlos está convergiendo a la utilización de sistemas de benchmark. Estos sistemas miden el rendimiento de los esquemas ante la presencia de diversos tipos de ataques [KP99]. Enfoques de evaluación y sistemas de benchmark para esquemas de marcas de agua para imágenes y audio han sido reportados, sin embargo, no existen benchmarks para esquemas de marcas de agua para video.

Los principales sistemas de benchmark son: Stirmark [PAK98, Pet00]: especializado en imágenes y audio; Checkmark [PVM+01], Optimark [STN+01] y WET (Watermark Evaluation Testbed) [KOGD04]: todos ellos especializados en imágenes. Existen planes de añadir ataques para videos al Checkmark (tales como edición de cuadros,

cambio de resolución de cuadros, reordenamiento de cuadros, cambios de framerate, etc.), pero hasta hoy en día no existe dicha adición. Los benchmarks actuales se pueden utilizar para videos si se aplican los ataques de imágenes a cada cuadro del video. No obstante, dichos ataques no modificarían al video temporalmente, ya que la posición y el número de cuadros no se vería afectado. Existe una carencia de un benchmark que soporte los principales ataques temporales, como son: la inserción, eliminación, intercambio, promedio de cuadros, etc.

La importancia de un benchmark especializado en ataques temporales reside en la capacidad de obtener índices de rendimiento de los esquemas de marcas de agua para video. Dichos índices deben medir la resistencia ante ataques temporales. La carencia de este tipo de benchmark provoca que cada diseñador mida el rendimiento de su esquema ante los ataques que considere pertinentes, tomando en cuenta parámetros muy específicos. Todo esto sin considerar que las implementaciones de un mismo ataque pueden variar entre sí.

Por ejemplo, la tabla 3.1 muestra los ataques temporales soportados por cada uno de los esquemas descritos en el capítulo anterior (y mostrados en la tabla 2.1). En dicha tabla se puede notar que los ataques reportados difieren entre sí y que los parámetros utilizados para realizar cada ataque son diferentes. Además, existen diversas implementaciones de un mismo ataque.

Esta falta de estandarización hace que cada autor realice las pruebas que considere adecuadas, no obstante muchas de esas pruebas no toman en cuenta lo que puede suceder en un caso real. Por ejemplo, todos los ataques de inserción insertan un cuadro no marcado en una posición específica, pero dicho cuadro incrustado (por imperceptibilidad) debe ser parecido a los cuadros adyacentes. Otro ejemplo, con respecto al intercambio de cuadros, es llevado a cabo intercambiando un par de cuadros en posiciones definidas aleatoriamente, esto puede causar una distorsión notoria si los cuadros intercambiados son poco parecidos.

Otro punto que no se toma en cuenta por los autores al diseñar ataques temporales es la duración de las escenas. Por ejemplo, específicamente en aplicaciones de marcas de

agua para video cinematográfico dicha duración influye principalmente en los ataques de eliminación e intercambio de los cuadros, puesto que las posiciones de intercambio y la cantidad de cuadros eliminados deben ser tomados en cuenta dependiendo de la duración promedio de las escenas. La duración de las escenas de dichos videos ha cambiado a través de los años: en la actualidad la duración promedio es de cuatro a seis segundos, mientras unas décadas atrás era de ocho a once segundos (casi lo doble) [TC11]. Este tipo de consideraciones importantes deben ser tomadas en cuenta para desarrollar un benchmark que sea útil en escenarios reales.

El diseño e implementación de un benchmark para esquemas de marcas de agua para video es una tarea difícil. Generalmente un equipo de investigación inicia esta labor pero al pasar el tiempo se unen diversos grupos o investigadores para incrementar el aporte del mismo. Uno de los objetivos de esta investigación es iniciar el diseño de un benchmark para esquemas de marcas de agua para video, al que llamaremos VidMark.

3.2. VidMark

En general, las entradas a un sistema de benchmark para esquemas de marcas de agua (para imágenes, audio o video) son los algoritmos de incrustación y extracción y las salidas son los índices de rendimiento. Estos últimos muestran el comportamiento del esquema de marcas de agua ante diversos experimentos, como por ejemplo ataques temporales. Internamente, el sistema de benchmark tiene un conjunto de datos fijos (colección de medios, llaves de incrustación/extracción, mensajes, etc.) para realizar una medición justa. Los índices de rendimiento obtenidos, usando dichos datos fijos, pueden ser utilizados para comparar dos o más esquemas de marcas de agua bajo las mismas condiciones. Es importante mencionar que este conjunto de datos puede ser manipulado para adaptarse a un escenario específico de aplicación.

En esta investigación se inicia el diseño de un benchmark para esquemas de marcas de agua para video llamado VidMark. El diagrama general del VidMark se muestra en

Esquema	Ataques temporales reportados					
	Inserción	Intercambio	Promedio	Eliminación	Decimación	Cambio de framerate.
Sun [SLY06]	-	Si	-	Si	-	25 a 24 fps y 25 a 30 fps
Lin [LD02, LD04, DL05], Dong [DHZQ08]	Inserción de cuadros: 5 %, 25 % y 50 %. Cada cuadro insertado no está marcado y se posiciona entre dos cuadros marcados	Intercambio de cuadros: 5 %, 25 % y 50 %. Cada cuadro es intercambiado con otro de su proximidad dado por un número obtenido por un generador de números aleatorios con dist. Gaussiana ($\sigma^2 = 5$)	Promedio de cuadros: 2, 3 y 4 cuadros. Una ventana deslizante de N número de cuadros sobre los cuadros del video genera un nuevo cuadro (mediante el promedio de los mismos)	Eliminación de cuadros: 5 %, 25 % y 50 %	Factor de decimación: 3, 4 ,5 y 6. Decimación por un factor N retiene el primer cuadro de cada N cuadros consecutivos	-
Guerrini [GLM05]*	-	Si	-	Si	-	-
Yang [YG06]	Inserción de cuadros: hasta 20 %	-	-	Eliminación de cuadros: hasta 20 %	-	25 a 30 fps y 25 a 15 fps
Zhang [hZXjZjS07]	Inserción de cuadros: 8, 12, 16, 20 y 22 %	-	-	Eliminación de cuadros: 8, 12, 16, 20 y 22 %	-	-
Kezheng [KWP08]	Inserción de cuadros: hasta 20 %	-	-	Eliminación de cuadros: hasta 20 %	-	-
Chen [CgGzL08]	Inserción de cuadros: 5, 10 y 20 %	-	-	Eliminación de cuadros: 5, 10 y 20 %	-	-

* genera una plantilla de autenticación para video que no incrusta información.

Tabla 3.1: Características de los ataques temporales reportados en los principales esquemas de marcas de agua

la figura 3.1. Los bloques sombreados mostrados en la figura representan los elementos que han sido implementados. Es importante mencionar que con dichos elementos el VidMark es totalmente funcional y puede ser usado para medir el rendimiento de un esquema de marcas de agua para video.

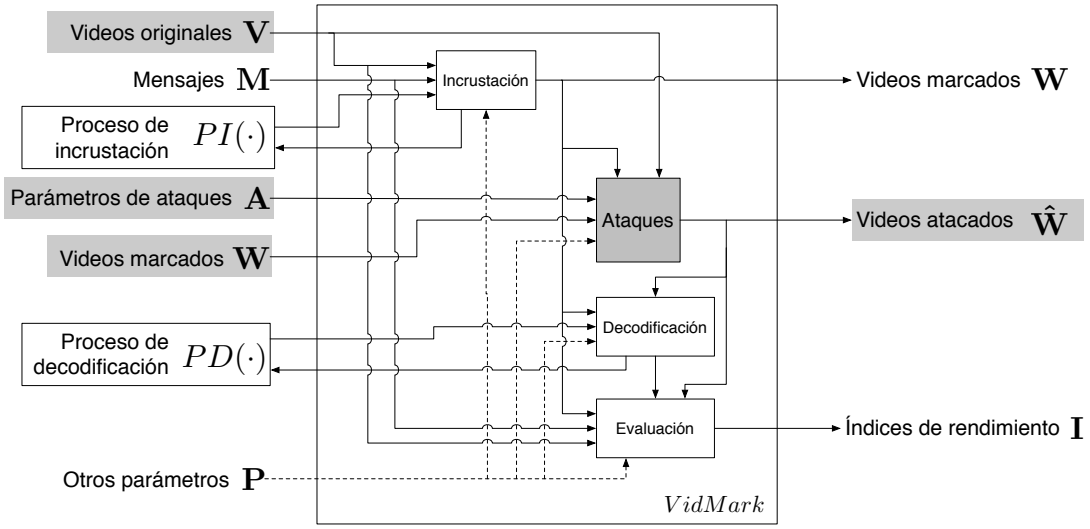


Figura 3.1: Diagrama general del VidMark

El benchmark VidMark está compuesto por los módulos de incrustación, ataques, decodificación y evaluación. En el módulo de incrustación, un mensaje $m \in \mathbf{M}$ es incrustado en un video original $x \in \mathbf{V}$ usando el proceso de incrustación $PI(\cdot)$. Este módulo puede usar otros parámetros dados en **P** para realizar la incrustación, p. ej. llaves de incrustación, parámetros de robustez, parámetros de fidelidad, etc. La salida de éste es un video marcado $y \in \mathbf{W}$. El módulo de ataques distorsiona a un video marcado y usando un conjunto de ataques temporales y de compresión MPEG (descritos en la siguientes subsecciones), configurados internamente por los parámetros **A**. Dependiendo del ataque puede ser necesario la utilización del video original x o algún parámetro adicional de **P**. La salida de este módulo es un conjunto de videos atacados $\hat{\mathbf{Y}} \subseteq \hat{\mathbf{W}}$, cada elemento de $\hat{\mathbf{Y}}$ es un video atacado usando un ataque con un parámetro específico, es decir se generarán tantos videos atacados como variaciones haya en el factor del ataque,

por ejemplo se pueden generar tres videos atacados usando el ataque de eliminación de cuadros si se eliminan el 10 %, 20 % y 30 % de cuadros. El módulo de decodificación realiza la extracción/detección de un mensaje en un video, original o atacado, usando el proceso de decodificación $PD(\cdot)$. Es posible la utilización de parámetros adicionales que sean elementos de **P**. La salida de este módulo es una aproximación del mensaje incrustado. Finalmente, el módulo de evaluación de rendimiento obtiene los índices de rendimiento del esquema de marcas de agua, es decir, aquí se realizan evaluaciones de robustez, fidelidad, falsos positivos, entre otras.

Los índices de rendimiento miden las propiedades del esquema a evaluar. De esta forma, generalmente se mide la robustez, la fidelidad, la carga útil, etc. Puesto que la robustez es la capacidad de detectar la marca en medios distorsionados, un índice de medición de robustez es la Razón de bits Correctamente Detectados (BCR), en videos que han sido atacados. El BCR es el número de bits correctamente detectados dividido por el total de bits incrustados, normalmente se expresa en porcentaje. Generalmente, la fidelidad, que es la similitud perceptual entre el video original y el video marcado, puede ser definida en términos de una Relación Señal a Ruido (SNR), también puede ser usada una variante llamada Relación Señal a Ruido Pico (PSNR). La carga útil es el tamaño máximo del mensaje que puede ser incrustado en el video, generalmente es expresada en una cantidad de bits por unidad de tiempo. Hay que destacar que existen diversas maneras de medir cada propiedad y que se pueden evaluar diversas propiedades como las descritas en la sección 1.2. Lo importante de una evaluación es que se utilicen las mismas métricas en condiciones controladas.

El VidMark es un proyecto incipiente ya que sólo uno de los cuatro módulos ha sido implementado: el módulo de ataques. Los bloques sombreados mostrados en la figura 3.1 representan los elementos que interactúan con el módulo de ataques. Actualmente se tiene una colección de 58 videos sin comprimir usando el formato YUV 4:2:0 y con resoluciones CIF y QCIF (352×288 y 176×144 respectivamente) a 25 fps. Dichos videos son utilizados comúnmente por diferentes autores para publicar sus resultados en tareas de procesamiento de videos. Un trabajo futuro es crear una colección de videos

útiles para diversos escenarios de aplicación, por ejemplo cine, documentales, series de televisión, noticieros, comerciales, etc. El resultado de la investigación documental en el estado del arte muestra que existen diversos ataques temporales que deben ser incluidos como medida de evaluación de robustez. Estos ataques temporales son incluidos en el benchmark VidMark y a continuación se describen detalladamente.

Ataques temporales

Análogamente al diagrama general de un esquema de marcas de agua presentado en la figura 1.1, se muestra un diagrama general de marcas de agua para video en la figura 3.2.

El proceso de incrustación oculta un mensaje M en el video original $\mathbf{X}$ usando, si es necesario, una llave de incrustación K_E. El resultado de este proceso es un video marcado $\mathbf{Y}$ que puede ser atacado, generando $\hat{\mathbf{Y}}$. El proceso de decodificación obtiene el mensaje M' del video $\hat{\mathbf{Y}}$ usando, si es necesario, la llave de decodificación K_D.

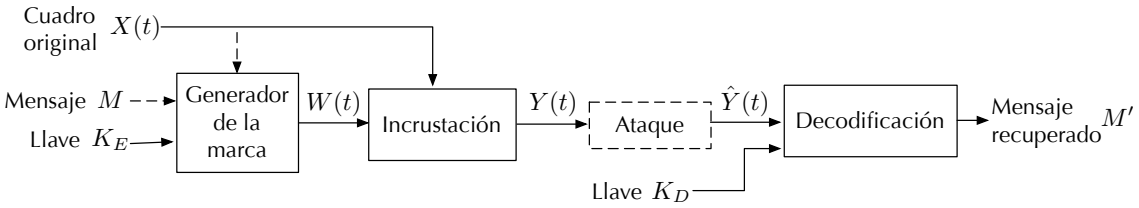


Figura 3.2: Diagrama general de un esquema de marcas de agua para video

El video original $\mathbf{X}$ es la secuencia ordenada

$$\mathbf{X} = \langle X(0), X(1), \ldots, X(t), \ldots, X(n-2), X(n-1) \rangle \tag{3.1}$$

donde $X(t)$ es el cuadro mostrado en el tiempo t y la cardinalidad de $\mathbf{X}$ es el número de cuadros que contiene $\mathbf{X}$, representada por $|\mathbf{X}| = n$. Aquí, $\langle \cdot \rangle$ representa una secuencia ordenada y $\{\cdot\}$ denota membresía de grupo. Es importante mencionar que por facilidad es posible tratar a una secuencia ordenada como un conjunto, recordando que sus

elementos tienen un orden preestablecido. De esta forma un video marcado $\mathbf{Y}$ es un secuencia ordenada

$$\mathbf{Y} = \langle Y(0), Y(1), \ldots, Y(t), \ldots, Y(n-2), Y(n-1) \rangle \tag{3.2}$$

y el video posiblemente atacado $\hat{\mathbf{Y}}$ es una secuencia

$$\hat{\mathbf{Y}} = \langle \hat{Y}(0), \hat{Y}(1), \hat{Y}(2), \ldots, \hat{Y}(t), \ldots \rangle \tag{3.3}$$

en este caso el número de cuadros de $\hat{\mathbf{Y}}$ puede ser diferente de n debido a los ataques temporales. Si $\hat{\mathbf{Y}}$ es idéntico a $\mathbf{Y}$, es decir $\hat{\mathbf{Y}}(t) = \mathbf{Y}(t)$ para todo t, el video marcado no fue atacado.

A continuación se describe detalladamente cómo afectan los Ataques temporales incluidos en el benchmark VidMark a los videos marcados.

Eliminación de cuadros: Los cuadros pueden ser eliminados o perdidos durante la entrega en diversos escenarios como en video streaming o en compresión de video. Esto implica que $|\hat{Y}| < |Y|$ y $\hat{Y} \subset Y$. Además, todos los cuadros de $\hat{\mathbf{Y}}$ mantienen un orden de presentación temporal ascendente. Esto es notablemente evidente pues no existen cambios de posición simultáneos entre los cuadros. La eliminación de cuadros se puede expresar como sigue:

$$\hat{\mathbf{Y}} = \mathbf{Y} - \{Y(i_1), Y(i_2), ..., Y(i_e)\} \tag{3.4}$$

donde $\{Y(i_1), Y(i_2), ..., Y(i_e)\}$ representa el conjunto de cuadros seleccionados aleatoriamente para ser eliminados. Aquí, el parámetro de control es e, el cual define el número de cuadros a eliminar. Es importante mencionar que cada cuadro tiene la misma probabilidad de ser seleccionado para ser eliminado. La figura 3.3 muestra este tipo de ataque.

Por ejemplo, sea $\mathbf{Y} = \langle y_1, y_2, y_3, y_4, y_5 \rangle$ un video marcado. Se desea eliminar el 40 % de cuadros, esto es $e = 2$. Así, si el conjunto de cuadros seleccionados para

ser eliminados es, por ejemplo, $\{Y(0) = y_1, Y(3) = y_4\}$. Después del ataque de eliminación, el video generado es $\hat{\mathbf{Y}} = \langle y_2, y_3, y_5 \rangle$. Se puede observar que $|\hat{Y}| < |Y|$ y que el orden de presentación se mantiene ascendentemente, es decir y_2 se presenta antes que y_3 y y_3 antes que y_5 como en el video marcado sin atacar.

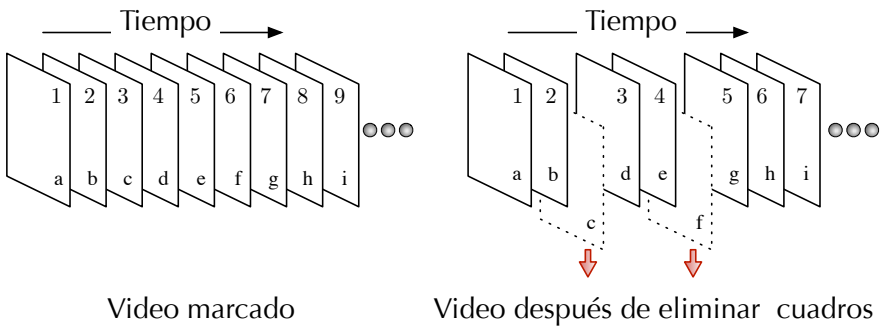


Figura 3.3: Ataque temporal de eliminación de cuadros

Inserción de cuadros: Este ataque consiste en insertar aleatoriamente cuadros no marcados (del video original X) en el video marcado Y. Después de este ataque, $|\hat{Y}| > |Y|$. El ataque de inserción de cuadros satisface las condiciones siguientes:

1. $Y \subset \hat{Y}$

2. $\exists x \in X \wedge x \in \hat{Y}$

La primera condición asegura que el video atacado $\hat{Y}$ contiene todos los cuadros de Y. La segunda condición asegura que adicionalmente algunos cuadros de $\hat{Y}$ son elementos X. El parámetro de control es el número de cuadros insertados, normalmente expresado en proporción al número de cuadros totales en Y. La figura 3.4 muestra este tipo de ataque. Por ejemplo, sea $Y = \langle y_1, y_2, y_3, y_4, y_5 \rangle$ un video marcado y $X = \langle x_1, x_2, x_3, x_4, x_5 \rangle$ el video original correspondiente. Se desea insertar el 40 % de cuadros no marcados, esto es, el 40 % de $|Y| = 5$. Así, el número de cuadros a insertar es 2. Después del ataque de inserción el video atacado puede ser $\hat{\mathbf{Y}} = \langle x_3, y_1, y_2, y_3, y_4, x_4, y_5 \rangle$. Como se puede observar, las dos condiciones anteriores se

cumplen. Es importante mencionar que los cuadros no marcados son seleccionados aleatoriamente al igual que la posición en donde se insertarán. Los cuadros y las posiciones tienen la misma probabilidad de ser seleccionados.

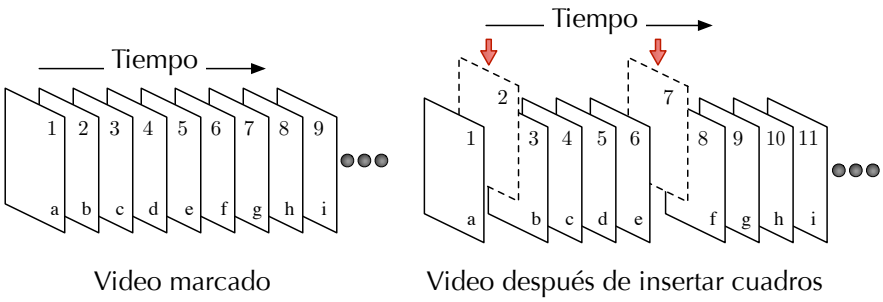


Figura 3.4: Ataque temporal de inserción de cuadros

Intercambio de cuadros: En este ataque dos o más cuadros cambian de posición al mismo tiempo. Hay dos formas de realizar este ataque:

1. Intercambio simple. Un cuadro se cambia de posición en el mismo video. Esto implica que más de un cuadro cambia de posición debido al corrimiento generado. En la figura 3.5 se muestra este ataque.

2. Intercambio simultáneo. Dos cuadros intercambian de posición simultáneamente. De esta manera, sólo dos cuadros modifican su posición. En la figura 1.3 se muestra el cambio de posición de un cuadro.

En este ataque $|\hat{Y}| = |Y|$ pero $\hat{\mathbf{Y}} \neq \mathbf{Y}$ porque $\hat{\mathbf{Y}}$ y $\mathbf{Y}$ no mantienen el mismo orden de presentación temporal. Por ejemplo, sea $\mathbf{Y} = \langle y_1, y_2, y_3, y_4, y_5 \rangle$ un video marcado, después de un ataque de intercambio simultáneo al $40\,\%$, el video generado es $\hat{\mathbf{Y}} = \langle y_1, y_5, y_3, y_4, y_2 \rangle$. En este ejemplo los cuadros intercambiados fueron $Y(1) = y_2$ y $Y(4) = y_5$. Los cuadros son seleccionados aleatoriamente.

Al someter a $\mathbf{Y}$ ante un intercambio simple al $40\,\%$, dos cuadros son seleccionados para cambiar de posición. Suponga que se seleccionan los cuadros $Y(1) = y_2$ y $Y(3) = y_4$, seguido a esto, se seleccionan las nuevas posiciones en donde se insertarán dichos

cuadros, por ejemplo y_2 con $t = 1$ pasará a $t = 3$ y y_4 con $t = 3$ pasará a $t = 0$. El video atacado será $\hat{\mathbf{Y}} = \langle y_4, y_1, y_3, y_2, y_5 \rangle$. De igual forma que los ataques anteriores, los cuadros y las posiciones tienen la misma probabilidad de ser seleccionados.

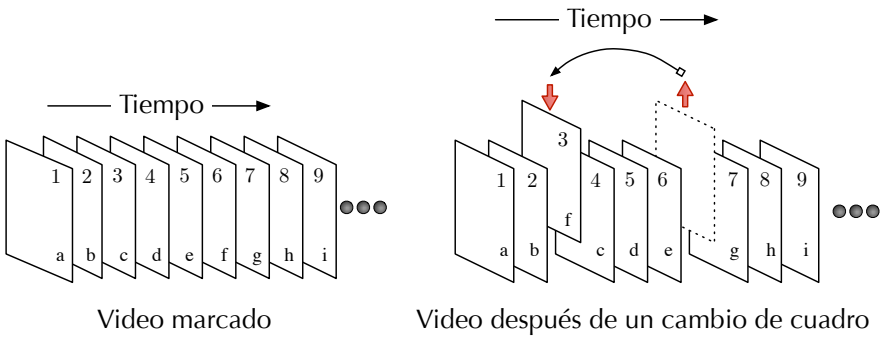


Figura 3.5: Ataque temporal de intercambio simple de cuadros

Decimación de cuadros: La decimación de cuadros ocurre cuando x cuadros consecutivos son eliminados. La decimación con un factor de f conserva el primer cuadro de cada f cuadros consecutivos. Otra forma de realizar la decimación es seleccionando aleatoriamente el cuadro a conservar de los f cuadros. En este ataque $\lfloor |\mathbf{Y}|/f \rfloor \leqslant |\hat{\mathbf{Y}}| \geqslant \lceil |\mathbf{Y}|/f \rceil$. De esta manera, los parámetros de control del ataque son el factor de decimación f y el cuadro a retener. En la figura 3.6 se muestra un ejemplo de decimación con factor de 3 reteniendo el primer cuadro. Por ejemplo, sea $\mathbf{Y} = \langle y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8 \rangle$ un video marcado, después de un ataque de decimación con factor 3 y reteniendo el primer cuadro, el video atacado es $\hat{\mathbf{Y}} = \langle y_1, y_4, y_7 \rangle$.

Promedio de cuadros: Este ataque se puede realizar de dos maneras:

1. Promedio por ventana. Una ventana de cuadros de tamaño fijo v se desliza temporalmente a través del video. En cada paso, se construye un cuadro compuesto calculando el promedio de todos los cuadros en la ventana. Este ataque, dado un video marcado $\mathbf{Y} = \langle Y(0), Y(1), \ldots, Y(n-1) \rangle$, genera un video marcado

$$\hat{\mathbf{Y}} = \langle \hat{Y}(0), \hat{Y}(1), \ldots, \hat{Y}(n-1) \rangle$$

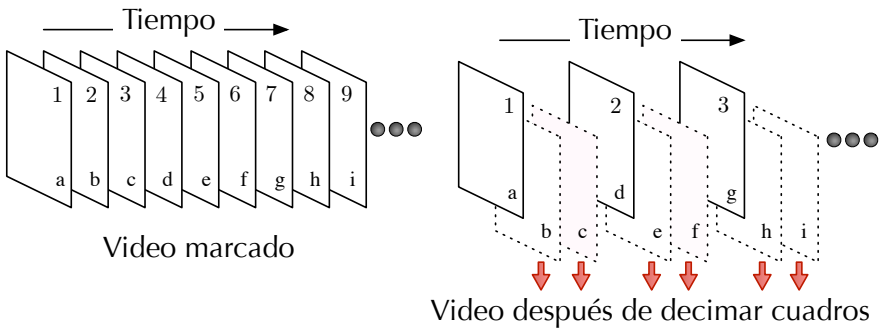


Figura 3.6: Ataque temporal de decimación de cuadros

donde cada cuadro es el resultado del promedio

$$\hat{Y}(t) = \frac{1}{k}\sum_{x=0}^{k-1} Y(t-x)$$

con $0 \leqslant t \leqslant n-1$ y

$$k = \begin{cases} v & \text{si } t-(v-1) \geqslant 0 \\ (v-1)-t & \text{en otro caso} \end{cases}$$

La figura 3.7 muestra un ejemplo de este ataque, se puede observar que la ventana corrediza es de tamaño 3. Se puede percibir que se está generando el segundo cuadro del video atacado, para ello promedian los primeros dos cuadros del video original. Para generar el tercer cuadro atacado la ventana se desliza una posición y promedia los cuadros cuya posición es 1, 2 y 3.

2. Promedio simple. Al igual que en el intercambio simultáneo, se seleccionan dos cuadros aleatoriamente y se sustituyen por el cuadro resultante al promediarlos. La figura 3.8 muestra un ejemplo del ataque de promedio simple, se puede observar que los cuadros 3 y 7 son promediados y sustituidos por una versión promediada de ellos.

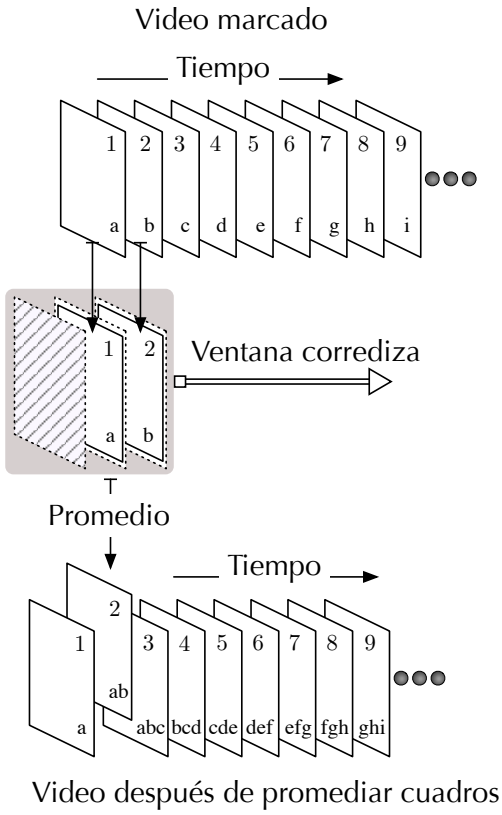


Figura 3.7: Ataque temporal de promedio por ventana

Cambio de framerate: El framerate es la frecuencia en la cual se presentan los cuadros, normalmente se expresa en cuadros por segundo o fps. Es importante notar que un cambio en el framerate no modifica la duración del video sólo el número de cuadros. En el benchmark VidMark se realiza este ataque de dos maneras:

1. Duplicación y eliminación. La manera de obtener los fps requeridos, es duplicando y eliminando cuadros. En la figura 3.9 se muestra un ejemplo de cambio de framerate por duplicación y eliminación, se pasó de 9 fps a 12 fps.

2. Fusión de cuadros. La única operación en cuadros permitida es el promedio de cuadros, ya sea para para insertar un cuadro o sustituir cuadros por una versión promediada de ellos mismos. En la figura 3.10 se muestra un ejemplo de cambio

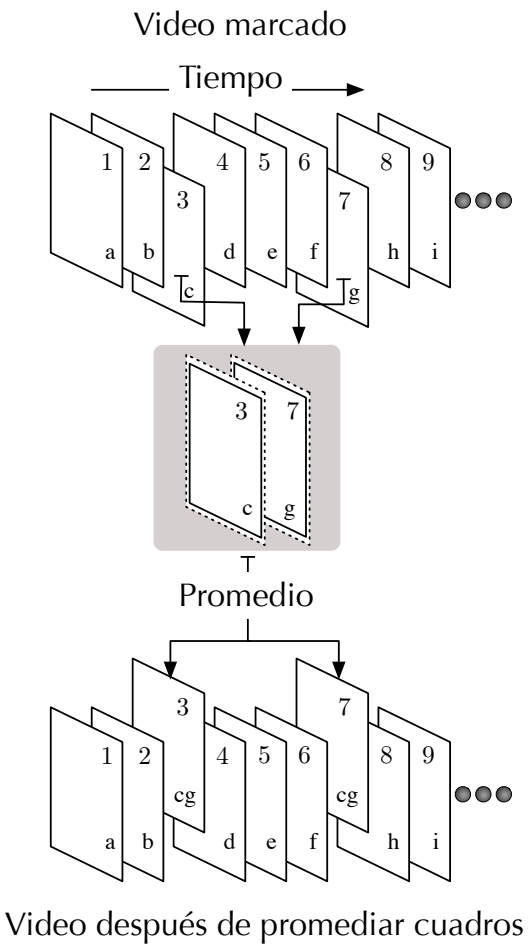


Figura 3.8: Ataque temporal de promedio simple

de framerate por fusión de cuadros, se pasó de 9 fps a 12 fps.

Ataque combinado: Se puede realizar una combinación de los ataques anteriores. Este tipo de ataque puede disminuir drásticamente la calidad visual del video.

Ataques de compresión MPEG

Como se mencionó en el capítulo 1, todo esquema de marcas de agua para video debe resistir ataques de compresión, pues prácticamente todos los videos están comprimidos. Es por eso que el VidMark debe soportar ataques de compresión. Los ataques

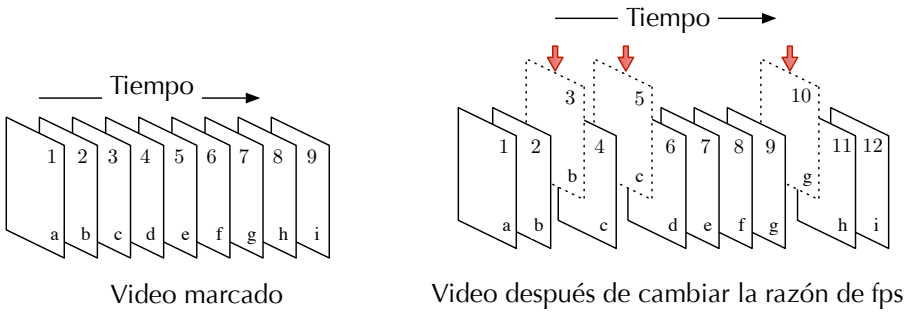


Figura 3.9: Ataque temporal de cambio de framerate por duplicación y eliminación

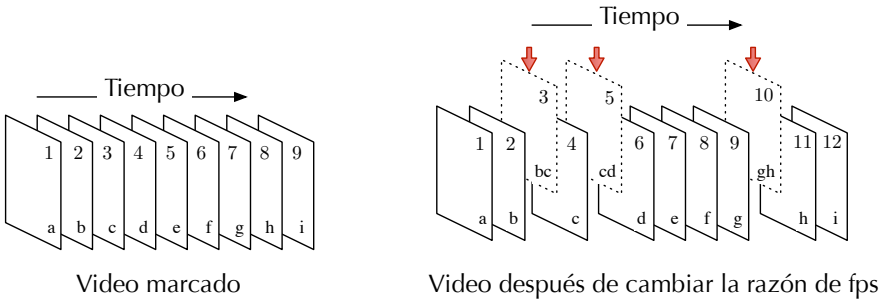


Figura 3.10: Ataque temporal de cambio de framerate por fusión

de compresión que son utilizados por el VidMark son ataques de compresión MPEG. MPEG-2 y MPEG-4 son una serie de estándares que proveen principalmente lineamientos específicos para la compresión de video y de audio. Las implementaciones de estos estándares de compresión dominan el mercado mundial.

MPEG-4 se diferencia de su predecesor MPEG-2 por la inclusión de objetos audiovisuales, dichos objetos pueden ser el fondo de un video, una persona hablando (sin el fondo asociado), la voz de la persona que esta hablando,etc. Este tipo de objetos son llamados objetos *naturales*, pues son extraídos de directamente de un video plano. Por otra parte existen los objetos *sintéticos* los cuales son contenidos credos para enriquecer el video, por ejemplo: texto, gráficos, animaciones,etc. La forma de codificar cada objeto se hace de la forma más eficiente posible. Esto quiere decir que se pueden usar un método de codificación diferente para cada clase de objeto, siempre y cuando el

método esté definido en el estándar. Es importante mencionar que en su forma codificada, los objetos pueden ser representados independientemente de los objetos que los rodean o del fondo mismo. Para sincronizar los objetos audio-visuales MPEG-4 provee un método formal de descripción sintáctica para describir la representación codificada de dichos objetos y los métodos usados para codificarlos.

En la presente investigación se considera que un video está formado por un solo objeto audio-visual, es decir, durante la compresión MPEG-4 no se hace esa separación por objetos, además se descarta el audio.

Los estándares MPEG encargados de la codificación de visual, es decir MPEG-2 parte 2, MPEG-4 parte 2 y MPEG-4 parte-10, tienen un funcionamiento similar. Todos ellos incluyen tres clases de cuadros durante la codificación: cuadros tipo I o intra-cuadros, cuadros de predicción o de tipo P y los cuadros de predicción bidireccional o de tipo B. Típicamente, un flujo de video siempre inicia con un cuadro tipo I, los que aparecen de forma regular (separados uno de otro de 25 a 250 cuadros). Estos cuadros tipo I regulares son cruciales para el acceso aleatorio de un video MPEG. Sin embargo, en el estándar MPEG parte 10, se puede optimizar para que los cuadros tipo I se generen de forma adaptable. Esto provoca que la compresión sea mayor, puesto que los cuadros tipo I son codificados como una imagen. En cambio los cuadros tipo P y B sólo mantienen una parte de la información de la imagen que representan. Esto genera que los cuadros tipo I ocupen más espacio de almacenamiento. Los cuadros tipo P mantienen sólo los cambios de la imagen con respecto al cuadro anterior tipo I. Por ejemplo, en una escena de un auto en movimiento con un fondo estático, únicamente el movimiento será codificado (la diferencia). Para ello se utilizan técnicas de estimación de movimiento, generalmente utilizando vectores de movimiento. De esta forma el codificador no necesita almacenar los píxeles del fondo puesto que serán tomados del cuadro tipo I previamente codificado. Finalmente, los cuadros tipo B ahorran más espacio de almacenamiento pues son codificados tomando en cuenta las diferencias entre el cuadro actual y los cuadros tipo I o P precedente o siguientes.

En la figura 3.11 se muestra la relación de los cuadros I, P y B. Adicionalmente,

se puede observar el bitrate de cada cuadro, nótese que los cuadros tipo I ocupan más espacio de almacenamiento, luego los tipo P y finalmente los tipo B. El bitrate de la secuencia es el promedio del almacenamiento de los cuadros, en este caso es 80 KB. Es importante mencionar que el principal parámetro de la compresión MPEG es el bitrate. Si se escogen bitrates pequeños el codificador modificará parámetros internos para eliminar mayor redundancia y así poder llegar a la tasa de bits por segundo definida. Esta eliminación de redundancia puede ser la eliminación de un cuadro de video completo. El flujo de video codificado consiste de códigos de entropía, vectores de movimiento, información de control acerca de la estructura del video y características de la codificación.

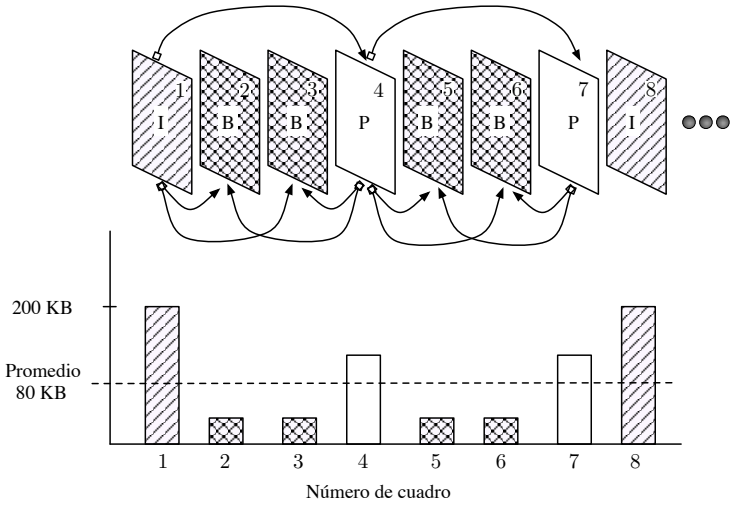


Figura 3.11: Diagrama de cuadros MPEG con el respectivo bitrate

Las codificaciones MPEG soportadas por el benchmark VidMark son: MPEG-2, MPEG-4 parte 2 y MPEG-4 parte 10. El codec utilizado para codificar MPEG-4 parte 10 es el x264[1] y los codecs usados para MPEG-2 y MPEG-4 parte 2 son de FFmpeg[2].

[1]http://www.videolan.org/developers/x264.html
[2]www.ffmpeg.org/

3.3. Discusión del capítulo

La carencia de un estándar de comparación de esquemas de marcas de agua para video provoca que muchos diseñadores midan el rendimiento de su esquema de formas variadas. Específicamente en esquemas resistentes a pérdida de sincronización temporal, la forma de probar la robustez ante ataques temporales es variada. En la tabla 3.1 se muestra que existen diversas implementaciones de un mismo ataque, además, los parámetros de cada ataque difieren entre los esquemas de cada autor.

El resultado de esta investigación es el inicio del diseño de un benchmark para video llamado VidMark. El aporte principal de este diseño es la identificación de los grandes subsistemas o módulos que debe contener la herramienta. A grandes rasgos se identificaron cuatro módulos: incrustación, ataques, decodificación y evaluación. Además, se identificaron y se implementaron los ataques temporales más representativos y utilizados por la comunidad científica. Aunado a esto el VidMark puede medir el rendimiento ante ataques de compresión MPEG. Durante la construcción del VidMark se identificaron oportunidades de mejorar la forma de atacar temporalmente los videos, por ejemplo realizar ataques temporales que analicen a los videos previamente para que el impacto perceptual de la distorsión que generan sea menor. Es importante mencionar que la herramienta no está completa, pero lo implementado permite medir cualquier esquema de marcas de agua para video ante ataques temporales y de compresión. Es decir, es una herramienta funcional, pero queda por realizar trabajo en el módulo de evaluación.

Actualemente, el VidMark genera videos atacados al aplicar un banco de ataques a videos marcados. Es tarea del diseñador de esquemas de marcas de agua evaluar el comportamiento de un diseño particular ante dichos ataques. En esta tesis se utiliza una evaluación de bits correctamente detectados o BCR. Adicionalmente se pueden realizar otras medidas de evaluación, pero debido al alcance de la investigación propuesta queda como trabajo futuro.

El desarrollo de una herramienta de Benchmark es enorme, por lo que se decidió ponerla libre y con oportunidades de colaboración. Para ello se creó el sitio web

http://www.vidmark.org/, en el cual se puso a disposición del visitante el código fuente así como la documentación de uso e instalación. Esto abre la puertas para futuras colaboraciones.

Capítulo 4

Esquema propuesto

En este capítulo se presenta el esquema propuesto de marcas de agua para video. Para ello se describe de forma detallada los módulos que conforman los procesos de incrustación y de decodificación. Finalmente, analiza el comportamiento del esquema propuesto ante los principales ataques temporales.

4.1. Sincronización temporal basada en llaves

El problema de la pérdida de sincronización temporal se reduce a un problema de generación de llaves. Esto es válido sólo si el esquema de marcas de agua para video utiliza llaves de incrustación/extracción para ocultar/detectar información, como es el caso del esquema propuesto. De esta manera, el peso de la sincronización está en la llave misma no en la marca de agua que se incrusta, puesto que existe una relación uno a uno entre la llave y la marca. A continuación se analiza este enfoque.

Nuevamente, en la figura 4.1 se muestra el diagrama general de marcas de agua para video, donde se puede apreciar que una marca $W(t)$ es creada y posteriormente insertada dentro de un cuadro de video original $X(t)$ para producir un cuadro marcado $Y(t)$. Para crear la marca $W(t)$ es necesaria una llave de incrustación K_E y el mensaje M ó parte de él. Este proceso se realiza hasta marcar todos los cuadros del video origi-

nal, generando un video marcado $\mathbf{Y}$. Este video marcado puede ser atacado generando $\hat{\mathbf{Y}}$. Para recuperar el mensaje, el video $\hat{\mathbf{Y}}$ es analizado por el decodificador de la marca. Internamente, el decodificador analiza cada cuadro para determinar la existencia de la marca y la representación binaria de la misma.

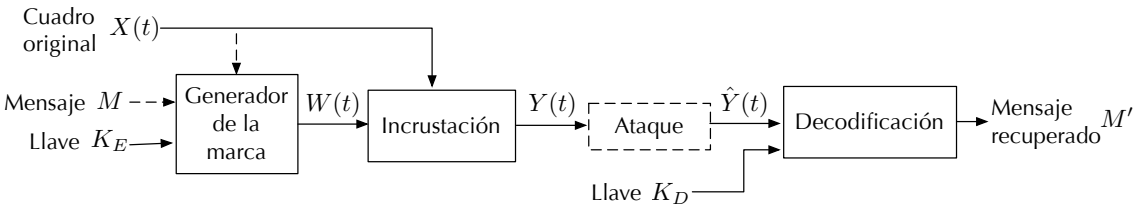


Figura 4.1: Diagrama general de un esquema de marcas de agua para video

El decodificador tiene como entrada adicional a la llave K_D. Cuando $K_E = K_D$ se dice que es un esquema de marcas de agua simétrico (llave privada), los esquemas de marcas de agua no simétricos (llave pública) usan llaves distintas. Si no se utiliza la llave correcta, el detector quedará imposibilitado para hallar la marca incrustada.

La llave es uno de los parámetros que determinan la estructura de la marca que se va a incrustar. Suponiendo que $K_E = K_D$ y que K_E es usado para crear una secuencia de llaves $\mathbf{K}$:

$$\mathbf{K} = \langle K(0), K(1), K(2), \ldots, K(t), \ldots \rangle$$

la cual es una secuencia ordenada de subllaves usadas para generar las marcas a incrustar en cada cuadro $Y(t)$. Específicamente, $K(t)$ es la subllave para generar el cuadro marcado $Y(t)$. La manera de generar esta secuencia de llaves se detalla en la sección 4.2.

La secuencia ordenada $\hat{\mathbf{K}}$:

$$\hat{\mathbf{K}} = \left\langle \hat{K}(0), \hat{K}(1), \hat{K}(2), \ldots, \hat{K}(t), \ldots \right\rangle$$

denota las llaves utilizadas para producir las marcas incrustadas en los cuadros $\hat{Y}$, donde $\hat{K}(t)$ es la llave para generar $\hat{Y}(t)$. Si el video marcado no ha sido modificado

entonces $\hat{Y}(t) = Y(t)$ y por tanto $\hat{K}(t) = K(t)$ para todo t. Sin embargo, los ataques temporales alteran la secuencia de los cuadros dentro del video, causando cambios en $\hat{K}(t)$. Bajo estos ataques no necesariamente $\hat{K}(t) = K(t)$. Los ataques de procesamiento de señales no afectan la estructura temporal del video, p. ej. cambio de brillo, sólo afectan a $\hat{Y}(t)$ pero no a $\hat{K}(t)$.

El objetivo del decodificador de la marca consiste en determinar la llave $\hat{K}(t)$ cuando un cuadro $\hat{Y}(t)$ es examinado. Si el detector puede determinar $\hat{K}(t)$ la sincronización temporal es llevada a cabo con éxito. Si el detector no puede determinar $\hat{K}(t)$, entonces la sincronización temporal se ha perdido. Así, la sincronización temporal es el proceso de determinar $\hat{K}(t)$ cuando un cuadro $\hat{Y}(t)$ es examinado por el detector. Esta definición de sincronización temporal fue propuesta en [LD04].

Esta noción de sincronización temporal sirve como base para el esquema propuesto, pues la sincronización temporal es llevada a cabo generado una secuencia de llaves. Aquí cada llave está ligada a una predecesora, con excepción de la llave maestra K_E. Este encadenamiento de llaves se espera en el proceso de decodificación. Si el encadenamiento se mantiene hasta decodificar el mensaje, no hubo pérdida de sincronización temporal.

A continuación se describe el esquema de marcas de agua propuesto, el cual comprende a los procesos de incrustación y de decodificación de la marca.

4.2. Proceso de incrustación de la marca

El esquema propuesto es un esquema basado en cuadros debido a que incrusta y decodifica una marca en cada cuadro. El Proceso de Incrustación de la Marca (PIM) describe la incrustación del mensaje dentro de un video de entrada.

El general, el esquema propuesto incrusta una marca en un cuadro utilizando información de cuadros marcados con anterioridad. Esta dependencia entre cuadros marcados genera un encadenamiento, lo que permite establecer un orden temporal. El esquema propuesto es diseñado para resistir ataques temporales, para ello se tiene que

guardar el mensaje redundantemente a lo largo del video.

En la figura 4.2 se muestra el diagrama del PIM del esquema propuesto. Se puede apreciar que el PIM esta formado por una serie de módulos que se comunican entre si para generar una secuencia de cuadros marcados a partir de una secuencias de cuadros originales, los que corresponden al video marcado y original respectivamente.

Por facilidad se describe el PIM en dos etapas: la primera etapa se encarga de crear la Redundancia de repetición y la Redundancia de periodo, la segunda etapa añade la Redundancia de escenas.

Los pasos de la primera etapa del PIM se describen a continuación, es recomendable apoyarse en la figura 4.2 para una mejor comprensión. En estos pasos se utilizan dos variables b e i cuya función es contar los cuadros consecutivos marcados usando la misma llave y contar los bits incrustados, respectivamente. Estos cuadros consecutivos marcados se conocen como Repetición β. Como nota adicional, los módulos que se mencionan son descritos a detalle posteriormente.

inicio: $b = 0$, $i = 0$, se reinician el generador de llaves y el control de redundancia temporal, se proveerá la llave K_E.

1. El módulo de control de redundancia temporal provee una llave $K(t)$ al módulo generador de la marca.

2. El generador de la marca utiliza la llave $K(t)$ para generar la marca $W(t)$.

3. El módulo incrustador de la marca esconde $W(t)$ en el cuadro actual $X(t)$, se genera un cuadro marcado $Y(t)$. Se incrementa el contador de repetición: $b = b + 1$.

4. En el módulo control de redundancia temporal:

   - Si el detector de repeticiones β detecta el inicio de una nueva repetición β, entonces

     a) El módulo detector de características analiza los β cuadros previos para obtener un vector de características.

b) El módulo generador de llaves produce una nueva llave $K(t+1)$ usando el vector de características obtenido previamente. Se reinicia el contador de repetición: $b = 0$, y se incrementa el contador de bits incrustados: $i = i + 1$.

c) ·si $i = |M|$, $|M|$ es la longitud del mensaje, entonces se reinician el generador de llaves y el control de redundancia temporal, se proveerá la llave K_E. Regresar al paso 1.

·En caso contrario, ir al paso 2.

- En caso contrario, ir al paso 2.

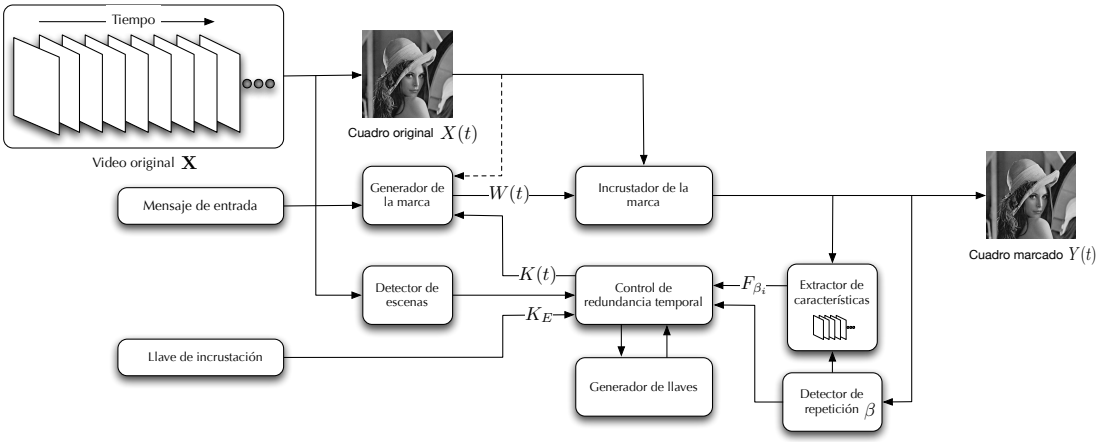


Figura 4.2: Proceso de incrustación de la marca de agua

Los pasos anteriores generan un video marcado $\mathbf{Y}$. La figura 4.3 muestra un ejemplo de la redundancia generada utilizando los pasos previos. En ella, los cuadrados representan cada cuadro de video. Los cuadros rellenos con el mismo patrón indican que fueron marcados con la misma llave. Se puede observar que después de algunos cuadros la llave se cambia. Por ejemplo, los cuadros del 0 al 3 están marcados con la misma llave, y los cuadros 4 y 5 fueron marcados con una llave diferente a la anterior. Como se mencionó, el número de cuadros marcados con la misma llave es llamado

repetición β. Esta repetición β controla la redundancia local y es adaptable, puesto que su tamaño varía dependiendo de las características de los cuadros contiguos.

Adicionalmente, se crea una redundancia de periodo α que controla la redundancia global. Esta redundancia α es usada para ocultar el mismo mensaje en repetidas ocasiones, tantas como quepan en el video. Por ejemplo en la figura 4.3, después de varias repeticiones β, la llave de incrustación es reiniciada, es decir $K(t) = K_E$. En este ejemplo, después de cuatro repeticiones β, la llave se reinicia. Así, los cuadros del 0 al 3 son marcados con la misma llave que los cuadros 13 y 14. Este reinicio de llave comienza un nuevo periodo de incrustación, el periodo α. En cada periodo α, el número de repeticiones β es el mismo. Los cuadros marcados con K_E son conocidos como cuadros de resincronización. Estos cuadros son utilizados por el detector como referencia cuando la sincronización se pierde.

El tamaño del mensaje que se puede ocultar usando el esquema propuesto depende de la cantidad de cuadros del video y del tamaño de las repeticiones β. En el ejemplo de la figura 4.3, el tamaño del mensaje es de 4 bits ($|M| = 4$) puesto que el esquema propuesto incrusta un bit del mensaje dentro de un solo cuadro (la marca representa un bit de información). Es importante notar que cuando el mensaje completo se ha incrustado y quedan cuadros disponibles el control de redundancia temporal incrustará nuevamente el mensaje, esto es llamado redundancia de periodo α.

Es importante observar que la generación de llaves depende de la llave anterior y de las características de la repetición β anterior. Una alta repetición β aumenta la redundancia, y se utiliza para resistir ataques temporales tales como: eliminación de cuadros e intercambio de cuadros, entre otros. En contraste, una repetición baja provoca una redundancia menor, lo que es un problema cuando se ataca temporalmente al video marcado.

La segunda etapa del PIM es una adición a los pasos anteriores. Ésta consiste en una modificación al control de redundancia temporal para esconder el mismo mensaje M dentro de los cuadros siguientes a un cambio de escena, lo que constituye la redundancia de escenas γ. Esta redundancia implica el uso de un detector de escenas que se

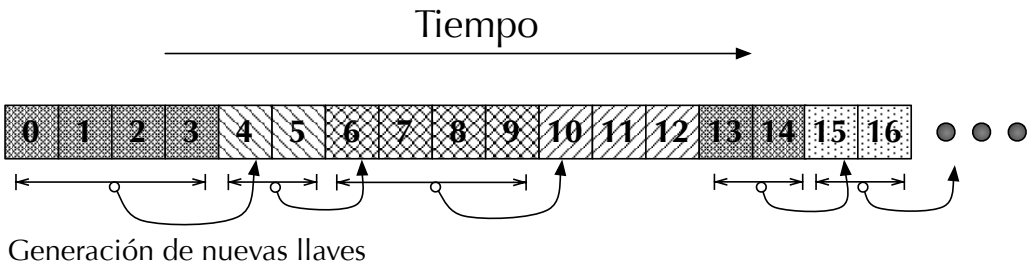


Figura 4.3: Redundancia de llaves usando repetición β adaptable

comunique con el control de redundancia temporal y que tenga el fin de informarle si el siguiente cuadro a procesar pertenece a una escena nueva. Esto se puede apreciar en la figura 4.2. El método que se usa para detectar cambios de escenas es el propuesto en [YL95]. En general, este método primero calcula la distancia entre todos los pares de cuadros adyacentes usando una métrica basada en histogramas. De esta manera, si el video es de N cuadros se obtienen $N - 1$ valores de distancia. Luego, mediante un análisis en dichos valores se determina si existe un cambio de escena. Específicamente, si el valor de la distancia a analizar es el máximo local en una vecindad de 30 valores se establece que hubo un cambio abrupto entre cuadros consecutivos, lo que corresponde a un cambio de escena.

Las redundancias descritas en la dos etapas de PIM se ejemplifican en la figura 4.4. Se puede observar, en la parte superior de los cuadros que existe un encadenamiento en la generación de llaves. Una nueva llave se genera después de un cambio de escena. Esta redundancia es la redundancia de escenas γ. Adicionalmente, se pueden observar las redundancias de repetición y de periodo. La redundancia de repetición β es la redundancia creada al ocultar la misma marca en cuadros consecutivos. Esta redundancia sirve como base para las otras redundancias. Finalmente, la repetición de periodo α se muestra en la parte inferior de la cuadros a través de los encadenamientos en la generación de llaves. Se puede observar que el PIM genera dos encadenamientos de llaves al mismo tiempo: uno oculta el mensaje a lo largo del video y el otro sólo después de

cada cambio de escena.

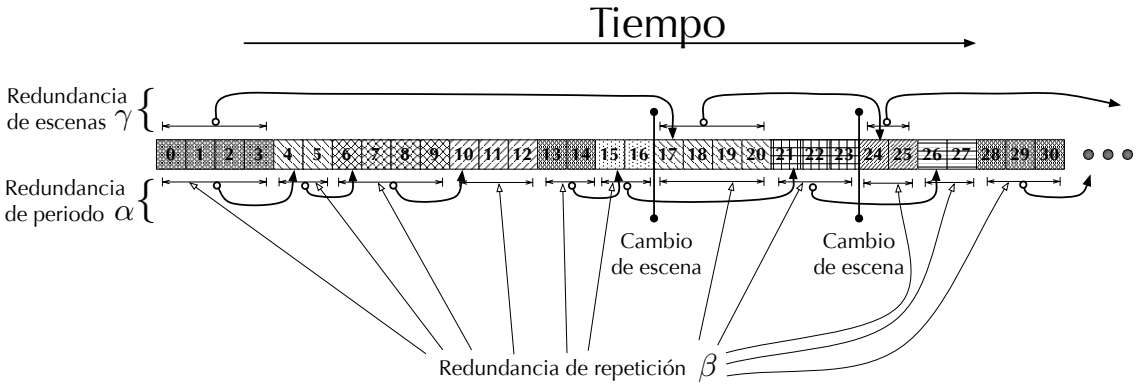


Figura 4.4: Redundancia de llaves generada por el esquema propuesto

Los principales módulos del proceso de incrustación de marca se describen a continuación.

Incrustación y detección de la marca

La incrustación de la marca comienza cuando el control de redundancia temporal libera una llave de incrustación. Esta llave $K(t)$ es una entrada del módulo generador de marca. Este módulo generador crea una marca $W(t)$, la cual es incrustada dentro del cuadro actual $X(t)$, generando un cuadro marcado $Y(t)$.

El esquema propuesto utiliza la técnica de espectro disperso descrita en la sección 1.2. De igual manera la marca $W(t) = \langle w_1, w_2, w_3, ..., w_M \rangle$ es una secuencia pseudoaleatoria de M números reales con distribución normal $N(0, 1)$, es decir con media 0 y varianza 1. Dicha secuencia pseudoaleatoria es generada usando la llave $K(t)$.

La incrustación de la marca se lleva a cabo al modificar algunos coeficientes de la transformada DCT del cuadro original. La modificación es añadir la marca $W(t)$ al valor de ciertos coeficientes. El número de coeficientes de la transformada DCT, o coeficientes DCT, de una imagen es el mismo que el número de píxeles de la imagen misma, p. ej. el número de coeficientes DCT de una imagen de 8×8 píxeles es 64.

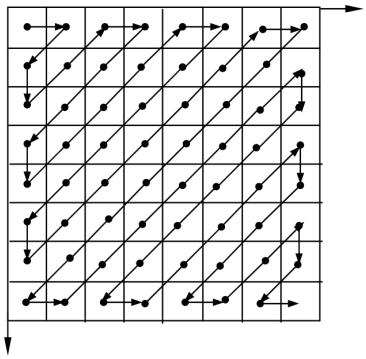
Los coeficientes DCT de una imagen se presentan en una matriz del tamaño de la imagen misma. Por ejemplo, en la figura 4.5 se muestra una imagen X de 8×8 píxeles y sus correspondientes coeficientes DCT; en ella se puede observar que el valor absoluto de los coeficientes se incrementa a medida que se acercan a la esquina superior izquierda. Dicha región representa las bajas frecuencias de la imagen; por otra parte, la región cercana a la esquina inferior derecha representa las altas frecuencias de la imagen. Esta representación frecuencial de los coeficientes es utilizada por algoritmos de procesamiento de imágenes, incluyendo compresión JPEG, en donde es importante distinguir entre bajas, medianas y altas frecuencias. Por convención, los coeficientes DCT de una imagen se reordenan usando un patron en zig-zag para formar un vector DCT cuyos elementos están ordenados de bajas frecuencias a altas frecuencias. La figura 4.6 muestra el patrón zig-zag usado en los coeficientes DCT de una imagen de 8×8 píxeles. Adicionalmente, en la misma imagen el patrón zig-zag indica el índice correspondiente en el vector DCT. De esta manera, el vector DCT de la imagen X de la figura 4.5 es $DCT(X) = \langle -26, -3, 0, -3, -2, -6, 2, -4, 1, -3, 1, 1, 5, 1, 2, -1, 1, -1, 1, 2, 0, 0, 0, 0, 0, -1, -1, 0, 0, ..., 0, 0 \rangle$. El índice 1 corresponde al coeficiente DCT -26, o lo que es lo mismo $DCT_1(X) = -26$, mientras que $DCT_6(X) = -6$.

$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix} . \qquad \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} .$$

Imagen X DCT(X)

Figura 4.5: Coeficientes DCT de una imagen

La incrustación de la marca consiste en generar la secuencia $W(t)$, usando la llave $K(t)$, e incrustarla dentro de M coeficientes consecutivos del vector DCT, como se indica en la ecuación 4.1.



Patrón zig-zag

1	2	6	7	15	16	28	29
3	5	8	14	17	27	30	43
4	9	13	18	26	31	42	44
10	12	19	25	32	41	45	54
11	20	24	33	40	46	53	55
21	23	34	39	47	52	56	61
22	35	38	48	51	57	60	62
36	37	49	50	58	59	63	64

Índices DCT

Figura 4.6: Patrón en zig-zag usado para generar el un vector de coeficientes DCT

$$DCT_{L+i}(X(t))' = DCT_{L+i}(X(t)) + s|DCT_{L+i}(X(t))|w_i \tag{4.1}$$

donde $i = 1, ..., M$, $w_i \in W(t)$, $X(t)$ es el cuadro original, s es un factor de robustez y L representa el número de los coeficientes de baja frecuencia que son saltados para obtener una mejor fidelidad en el marcado, $L = |X|/4$ y $M = |X|/4$. El factor de robustez s como se indicó modifica la magnitud de la marca y su valor normalmente se establece tomando en cuenta la distorsión que ocasiona. En la sección 5.4 se muestra un experimento que mide la distorsión provocada por s. Los coeficientes del vector DCT marcados son reinsertados para formar la matriz DCT. Para obtener el cuadro marcado $Y(t)$ se aplica la transformada inversa DCT.

En la detección de la marca, se realiza primero un filtro pasa bajas (como se sugiere en [KDHM99]), seguido por una correlación cruzada entre los coeficientes DCT del cuadro $\hat{Y}(t)$, que deben ser los mismos que los usados en el marcado, y la marca $W(t)$, generada usando la llave $\hat{K}(t)$. La correlación es calculada usando la ecuación 1.2.

$$c = C(DCT(\hat{Y}(t)), W(t))$$

Normalmente, la detección se realiza comparando el valor de la correlación c con un umbral predefinido U. Si $|c| > U$ se considera que la marca está presente en el cuadro, en otro caso la marca no fue detectada. El umbral U juega un papel muy importante y su valor se determina considerando dos probabilidades: la probabilidad de falso positivo y la probabilidad de falso negativo. El esquema propuesto usa un umbral adaptable U, el cual toma en cuenta la magnitud absoluta de los coeficientes DCT del cuadro marcado. De esta manera si los coeficientes tienen mucha energía el umbral será elevado, pues se espera que dichos coeficientes hayan sido distorsionados con mayor severidad durante el marcado. Por el otro lado, si los coeficientes tienen poca energía el umbral será pequeño para permitir una correcta detección de la marca. De esta manera el umbral varía dependiendo de las características de cada cuadro. Esta mejora en la definición del umbral es descrito en [PBBC97] y garantiza que la probabilidad de falsos positivos sea menor que 10^{-6}. El umbral U se calcula de la siguiente manera:

$$U = \frac{s}{3M} \sum_{i=1}^{M} |DCT_{L+i}(\hat{Y}(t))| \tag{4.2}$$

Hasta este punto se puede detectar la marca o no de un cuadro. Para incrustar un bit de información sólo basta modificar el signo de la marca durante la incrustación, de esta forma si se desea ocultar un '1' lógico $W(t) = W(t)$ y si se desea ocultar un '0' lógico $W(t) = -W(t)$. En la detección se procede de igual manera solo basta con considerar la magnitud de la correlación absoluta, es decir si $|c| > U$ la marca se detectó satisfactoriamente, como se propone en [CMB+08]. El valor del bit extraído depende del signo de la correlación, si es positiva se recuperó un 1, si es negativa un 0. Esta modificación del signo de la marca no influye en la detección, puesto que la secuencia pseudoaleatoria mantiene sus propiedades estadísticas, media 0 y varianza 1. Adicionalmente, el cuadro original $X(t)$ será modificado de igual forma ya sea que si cambia el signo de la marca o no. Lo que sucederá por ejemplo, es que un coeficiente en lugar de sumarle una parte de la marca se le restará. Tomando en consideración todas las modificaciones, los coeficientes tendrán la misma media, pero la varianza se

incrementará, independientemente de que si se le sumó o si se le restó la marca.

Control de redundancia temporal

El control de redundancia temporal es el encargado de mantener la sincronización temporal, generando la redundancia en las llaves de incrustación. La redundancia temporal se logra incrustando la misma marca en cuadros consecutivos del video. Los parámetros principales para manejar la generación de llaves son:

- La llave de incrustación K_E que se utiliza para generar llaves subsecuentes.
- La repetición β, que es el número de cuadros consecutivos que serán marcados utilizando la misma llave. El número de cuadros consecutivos a marcar está dado por la información mutua entre ellos. Así, cuando la semejanza de cuadros consecutivos disminuye, los siguientes cuadros serán parte de la siguiente repetición β. Después de marcar los β cuadros, se genera una nueva llave.
- El periodo α, que es el número de cuadros a marcar después de reiniciado el generador. Así el periodo α depende del tamaño del mensaje de entrada $|M|$ y del tamaño de las β repeticiones utilizadas para incrustar todo el mensaje. Después de la incrustación, el mismo mensaje será incrustado nuevamente.
- El periodo γ, que se refiere a la redundancia de inicio de escenas. El mensaje es incrustado después de detectar un cambio de escena.

Es importante mencionar que el control de redundancia temporal es el elemento principal del PIM, pues controla la generación de llaves del esquema propuesto.

Detector de repeticiones β

La repetición β representa el número de cuadros que serán incrustados con la misma marca. Para determinar el número de cuadros que formarán parte de una repetición β se analiza la similitud entre ellos. Normalmente en una secuencia de video, los

cuadros consecutivos tienen características comunes, conforme más se alejan entre sí, las características difieren más. De esta forma para obtener un bloque de cuadros que compartan características, lo que corresponde a una repetición β, se debe usar una métrica de similitud. Por tanto, se propone el uso de la información mutua que exista entre los cuadros, esta métrica mide la dependencia que existe entre dos cuadros. Si los cuadros son similares, la información mutua es alta y viceversa. La ecuación 4.3 describe la métrica de información mutua entre dos imágenes X y Y[CT91]:

$$I(X,Y) = \sum_{x}\sum_{y} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{4.3}$$

donde $p(x,y)$ es la función de probabilidad de masa conjunta de las imágenes X y Y (es decir cuadro $Y(t)$ y cuadro $Y(t+1)$), $p(x)$ y $p(y)$ representan la función de probabilidad de masa de X y Y, respectivamente.

Para determinar la cantidad de cuadros que formarán parte de una repetición β se utiliza un umbral U. Este umbral sirve para establecer el valor máximo de la información mutua que debe haber entre cuadros consecutivos, de esta manera cuando la información mutua entre dos cuadros lo sobrepase comienza una nueva repetición β. Experimentos acerca de esta métrica se presentan en la sección 5.2. Es tarea del usuario del esquema propuesto definir el umbral necesario que se adapte para los requerimientos de la aplicación. Recordando que a mayor redundancia menor cantidad de bits incrustados.

Extracción de características

La extracción de características permite que las llaves generadas sean dependientes del contenido del video puesto que cada nueva llave se crea usando las características de algunos cuadros previos.

El Extractor de Características (EC) es el encargado de procesar un conjunto de cuadros y obtener un conjunto de características. Específicamente, el EC examina β

cuadros de video y genera un vector de características $F_\beta = (F1, F2, F3)$, donde $F1$, $F2$ y $F3$ son números que representan las características de los β cuadros.

El proceso para generar el vector F_β es el siguiente:

1. Cada cuadro de la repetición β se procesa con un filtro Gaussiano, para eliminar las altas frecuencias que se encuentran en ellos. El filtro Gaussiano utiliza $\sigma = 4$.

2. Los β cuadros procesados son promediados entre si, generando una imagen P.

3. La imagen P se divide en bloques cuadrados de tamaño 32×32 píxeles. Los bloques se clasifican aleatoriamente en tres clases: A, B y C. Esto se aprecia en la parte izquierda de la figura 4.7.

4. Se obtiene el promedio de los píxeles de las tres clases. De esta manera se generan a, b y c, siendo a=promedio(región A), b=promedio(región B) y c=promedio(región C).

5. La imagen P se procesa para obtener puntos característicos, los cuales representan píxeles que se encuentran en intersecciones de bordes. La magnitud del punto característico es diferente a la magnitud del pixel en donde se encuentra. Para ello se propone el uso del detector de Harris [HS88]. Dicho detector de puntos característicos es ampliamente utilizado por tener invarianza ante rotación, escala, cambios de iluminación y ruido. En la parte derecha de la figura 4.7 se muestran, de manera simbólica, los puntos característicos encontrados en la imagen P.

6. De manera análoga al paso 4, se genera el promedio de las magnitudes de los puntos característicos de cada una de las clases. Así, se obtiene $p1$=promedioPuntos(región A), $p2$=promedioPuntos(región B) y $p3$=promedioPuntos(región C).

7. Finalmente se calcula el valor de las características mediante $F1 = a + p1$, $F2 = b + p2$ y $F3 = c + p3$. De esta manera, se forma el vector de características $F_\beta = (F1, F2, F3)$.

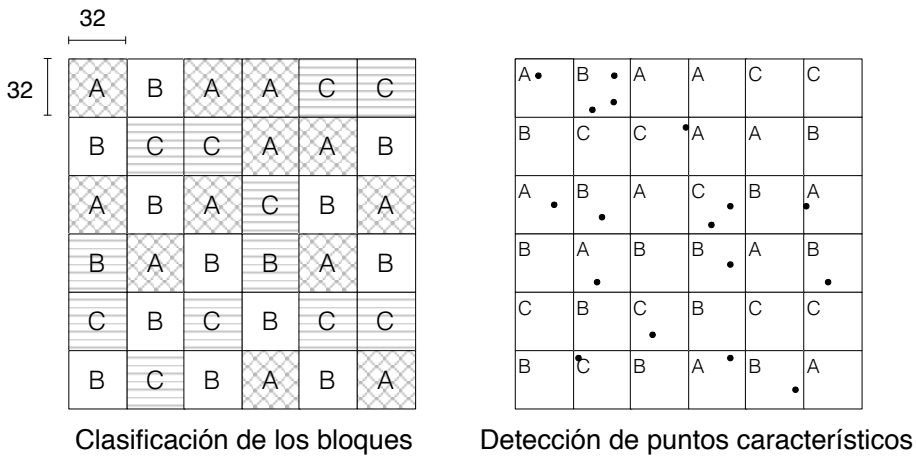


Figura 4.7: Clasificación llevada a cabo durante la extracción de características

En resumen, la extracción de características consiste en procesar los cuadros marcados con la misma llave para extraer características comunes en la forma de un vector F_β. Dicho vector es utilizado por el generador de llaves, descrito a continuación, para proveer una llave nueva.

Generador de llaves

El generador de llaves es utilizado para crear nuevas llaves y es, en general, una Máquina de Estados Finitos (FSM) . Una máquina de estados es una tupla $(S, S_0, I, O, \phi, \lambda)$, donde S es el conjunto de estados, S_0 es el conjunto de estado iniciales, I es el dominio de entrada, O es el rango de salida, ϕ es la función de transición de estados y λ es la función de salida [HMU06].

- $S = \{s_0, s_1, s_2, ..., s_{|s|-1}\}$ es un conjunto finito no vacío de estados en la FSM. Un estado es una representación de memoria que tiene una salida y define el comportamiento de la FSM, un estado depende de la entrada actual y de las entradas pasadas. En cualquier tiempo dado $t \geq 0$, el estado de la máquina (conocido como estado actual $s(t)$) es un miembro de S, es decir $s(t) \in S$.

- S_0 es el conjunto de estados iniciales y es un subconjunto no vacío de S. En esta tesis se asume que S_0 es un conjunto con sólo un elemento.
- I es el dominio de entrada, el cual es el conjunto de todas las posibles entradas a la FSM.
- O es el rango de salida, el cual es el conjunto de todas las posibles salidas de la FSM.
- La función de transición de estados $\phi : S \times I \rightarrow S$ describe la transición de estados de la FSM, si el estado actual de la FSM es $s(t) \in S$ y la entrada actual es $i(t) \in I$, entonces el siguiente estado será $s(t+1) = \phi(s(t), i(t))$.
- La función de salida $\lambda : S \rightarrow O$ es un mapeo de cada estado a un valor de salida. Dado el estado actual $s(t)$, la salida de la FSM es $\lambda(s(t)) \in O$.

El generador de llaves tiene un dominio de entrada I, el cual es el producto cartesiano del espacio de llaves K con el rango de todos los posibles vectores de características, obtenidos por el extractor de características. El rango de salida es el espacio de llaves $O = K$. El estado inicial depende de K_E. En el paso 1 de PIM, la salida del FSM es $\lambda(s(t))$. La transición de estados ocurre en el paso 4(b) donde $s(t+1) = \phi(s(t), K_E, F_\beta)$. Después de una transición de estado, se obtiene una llave $K(t+1)$ con $\lambda(s(t+1))$. La función de salida $\lambda(\cdot)$ es una función uno a uno.

La figura 4.8 muestra el FSM, propuesto por Lin en [LD02], utilizado como generador de llaves. Esta máquina de estados utiliza las características $(F1, F2, F3)$ obtenidas del extractor de características. El estado inicial es el estado cero. La función de transición de las llaves están en cada estado y, exceptuando el estado inicial y el estado 2, incluye el valor de la llave actual y el valor de algunas constantes (c_1, c_2 y c_3), dichas constantes aportan seguridad al sistema pues sirven como llaves para la decodificación. El valor de las constantes es definido por un usuario del esquema propuesto. En el estado 2, la siguiente llave se genera por la negación binaria de $K(t)$. Las características $F1$, $F2$ o $F3$ con los valores máximos determinan la transición de estado.

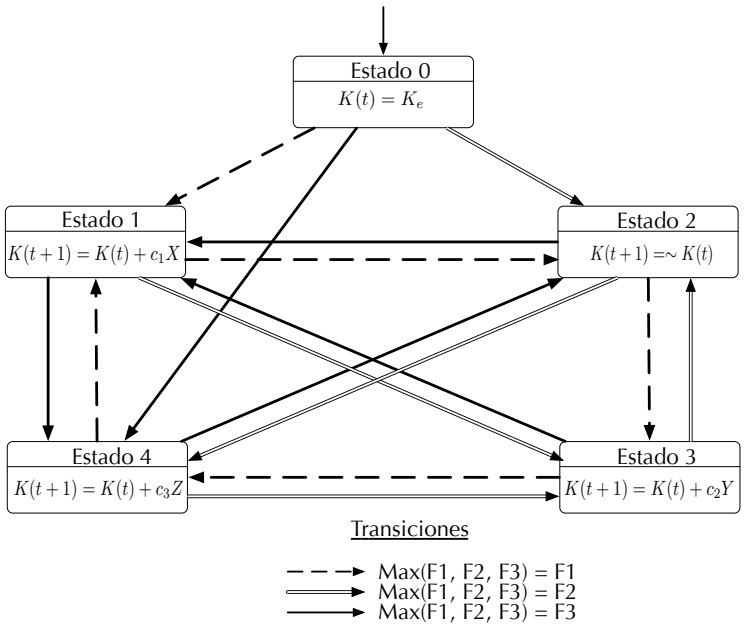


Figura 4.8: FSM usado como generador de llaves

Este método de generación de llaves utiliza la llave actual y las características de los cuadros locales (incluyendo el cuadro actual) para generar una dependencia temporal en las llaves generadas. Una manera de observar la funcionalidad del generador de llave es como *predictor*, el cual predice futuras llaves a partir de la actual y la pasada.

4.3. Proceso de decodificación de la marca

El Proceso de Decodificación de la Marca (PDM) utiliza una memoria para realizar la sincronización temporal. La figura 4.9 muestra el diagrama para este proceso. Aquí, el detector de marca corresponde a la técnica descrita en la sección anterior. El extractor de características y los módulos generadores de llave son idénticos a aquellos utilizados en el PIM.

La memoria es una cola Q, de tamaño $|Q|$, que almacena las llaves utilizadas para establecer y mantener sincronización temporal. La cola almacena en cada entrada la llave de incrustación y el estado dentro del generador de llaves, es decir una tupla

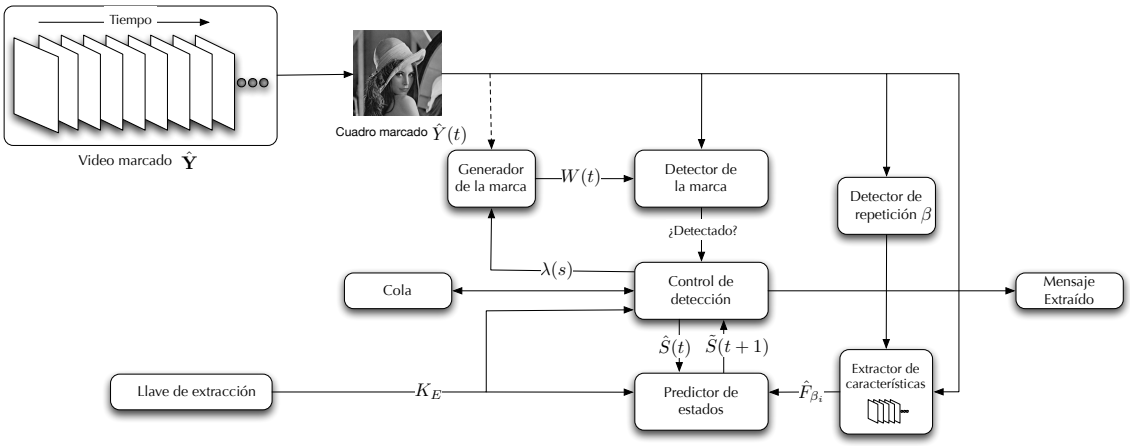


Figura 4.9: Proceso de decodificación de la marca

(llave, estado). El módulo predictor de estados corresponde a la función de transición del FSM usado en el PIM.

El proceso de decodificación de la marca es:

1. Iniciar el detector de marca, con cola vacía.

2. Leer un cuadro de video de entrada.

3. Intentar la detección de la marca utilizando:

   a) La llave K_E, y

   b) Cada llave almacenada en la cola.

4. Si ninguna marca es detectada durante el paso 3, entonces regresar al paso 2 e intentar la detección de la marca en el siguiente cuadro.

5. Usando la misma marca detectada, intentar detectarla en los siguientes cuadros hasta que la detección falle. El número de cuadros detectados corresponden a una repetición β, llamada β_i. El indice i de la repetición detectada representa la posición del bit detectado. Después de la detección de la repetición i se incrementa en 1. Este paso es una extensión del paso 3.

6. Si la marca detectada no corresponde a la llave K_E, esta llave debe estar presente en la cola. Se promueve la tupla (llave, estado) a la cabeza de la cola.

7. Obtener el vector de características de los cuadros actuales β_i. El detector de repeticiones β manda los cuadros de la actual repetición β_i al módulo de extracción de características. Este módulo libera el vector de características de los cuadros actuales marcados es decir $\hat{F_{\beta_i}}$.

8. Utilizando el vector de características del paso 7 y el estado almacenado en la cola (utilizar el estado inicial si fue detectado K_E) como entradas en el predictor de estados, se obtiene la tupla (siguiente llave, siguiente estado), la cual es insertada a la cabeza de la cola. Es importante notar que, el final de la cola podría perderse debido al tamaño finito de la misma. Lo que podría provocar una pérdida de sincronización temporal.

9. Ir al paso 2 para detectar la marca en el siguiente cuadro.

En resumen, los pasos 3 y 5 son usados para detectar la marca en los cuadros pertenecientes a una repetición β, el paso 7 obtiene el vector de características de la repetición y el paso 8 calcula la siguiente llave esperada. La cola almacena las llaves de incrustación detectadas recientemente, entre ellas la llave que se espera en el siguiente cuadro. Incrementar el tamaño de cola permite almacenar más llaves, lo que provee mayor robustez al esquema ante ataques de intercambio de cuadros. Cuando se realiza un ataque de este tipo, es posible que la marca incrustada en un cuadro provenga de una llave generada en el pasado. Sin embargo, el incremento en el tamaño de la cola afecta el costo computacional, ya que en el peor de los casos se intentará detectar una marca usando cada una de las llaves dentro de la cola.

El costo computacional del PDM en el peor de los casos está en el orden de $((2 \times |Q| + 1) \times DM) + EC + GLL$ por cuadro, donde $|Q|$ es el tamaño de la cola, DM es el costo computacional del detector de la marca, EC es el costo computacional del extractor de características y GLL es el costo computacional del generador de llaves.

Es evidente que el costo computacional del esquema propuesto recae en el módulo del detector de la marca.

El detector no requiere conocer parámetros de redundancia temporal utilizados en el proceso de incrustación (α, β y γ) como información adicional, ya que la búsqueda realizada durante los pasos 3 y 5 de PDM permite la detección de la marca sin conocimiento previo de la correspondencia entre la llave y el índice del cuadro. Es decir, al detectar un cuadro se obtiene de forma natural la posición del bit extraído dentro del mensaje, puesto que las detecciones son llevadas a cabo de forma ordenada. A diferencia de PIM, en el PDM el extractor de características y los componentes del generador de llaves (predictor de estado) pueden estar activos durante el procesado de cada cuadro. Los únicos cuadros que no incluyen el uso del extractor de características y generador de llave son aquellos donde ninguna marca puede ser encontrada. Así, cuando en un cuadro no se detecta alguna marca, el cuadro mismo es ignorado.

4.4. Análisis

A continuación se analiza el comportamiento del esquema propuesto ante los principales ataques temporales. El objetivo de este análisis es determinar las condiciones necesarias para que se mantenga la sincronización temporal en el mensaje detectado.

Como se mencionó anteriormente, el detector de la marca debe encontrar la llave $\hat{K}(t)$ cuando un cuadro $\hat{Y}(t)$ es examinado. Puesto que $\hat{K}(t) = \lambda(\tilde{s}(t))$ y $\lambda(\cdot)$ es una función uno a uno, entonces el detector debe determinar $\tilde{s}(t)$ cuando $\hat{Y}(t)$ es examinado. De esta manera, el análisis que a continuación se presenta es basado en estados.

Por facilidad se asume que: 1) dada una llave el detector de la marca siempre es correcto, 2) el detector de características es correcto para los β_i cuadros procesados en el PDM. Con el término "correcto" se refiere a que no hay falsos positivos, falsos negativos ni pérdidas y 3) se usa una máquina de estados determinista.

El siguiente análisis se centra en identificar las condiciones para que se obtenga la sincronización temporal. La sincronización temporal entre dos cuadros $\hat{Y}(t)$ y $\hat{Y}(t+\delta)$

es llevada a cabo si y sólo si:

1. La marca $W(t)$ es detectada en el cuadro $\hat{Y}(t)$. En este punto se genera el siguiente estado $\hat{s}(t+1) = \phi(\hat{s}(t), K_E, \hat{F_{\beta_i}})$. Donde $\hat{s}(t)$ es el estado de $\hat{Y}(t)$. $\hat{F_{\beta_i}}$ representa el vector de características de los β_i cuadros al que pertenece $\hat{Y}(t)$.

2. La marca $W(t+1)$ es detectada en el cuadro $\hat{Y}(t+\delta)$, es decir, que el estado de $\hat{Y}(t+\delta)$ es $\hat{s}(t+1)$.

Estas dos condiciones generan una *cadena de estados*, la cual garantiza la sincronización temporal de los dos cuadros. Para extraer el mensaje completo es necesario que la longitud de la cadena de estados sea igual a la longitud del mensaje.

Ausencia de ataques

Para probar que el esquema propuesto es robusto ante la ausencia de ataques sólo basta con seguir los pasos del PDM. Los primeros β_i cuadros de $\hat{Y}$, marcados usando la llave K_E, son detectados en el paso 3 y 5 del PDM. Luego, el predictor de estados genera el siguiente estado $\hat{s}(t+1) = \phi(\hat{s}(t), K_E, \hat{F_{\beta_i}})$, el cual es usado para generar la llave $\hat{K}(t+1) = \lambda(\hat{s}(t+1))$. Este nuevo estado es almacenado en $q(0)$ (la cabeza de la cola). De igual manera, los siguientes β_{i+1} cuadros son detectados, en el paso 3 y 5 del PDM, usando los estados almacenados en la cola. Recordando que, en el paso previo se generó el estado para detectar la marca del cuadro actual. En este caso, y para todos los siguientes, la cadena de estados generada por el PIM se mantiene en el PDM; entonces, la sincronización temporal es llevada a cabo.

Inserción de cuadros no marcados

Un ataque de inserción de cuadros sucede cuando n cuadros sin marcar son insertados entre dos cuadros marcados consecutivos $Y(t)$ y $Y(t+1)$. El segmento de video que se genera después de este ataque es $\langle \hat{Y}(u), \hat{Y}(u+1), ..., \hat{Y}(u+n), \hat{Y}(u+n+1) \rangle$, donde $Y(t) = \hat{Y}(u)$ and $Y(t+1) = \hat{Y}(u+n+1)$. Suponiendo que $\hat{Y}(u)$ se detecta, el siguiente estado es almacenado en $q(0)$ (paso 6 de PDM). Cuando el siguiente cuadro $\hat{Y}(u+1)$ se procesa, el PDM no puede detectar ninguna marca (paso 4 de PDM) y procesará el siguiente cuadro $\hat{Y}(u+2)$, y así consecutivamente hasta el cuadro $\hat{Y}(u+n+1)$.

Es importante mencionar que la cola no cambia. Así, el detector de la marca detectará la marca incrustada en $\hat{Y}(u+n+1)$ utilizando un estado en Q. Si se logra la detección utililizando la marca almacenada en $q(0)$, entonces $\hat{Y}(u)$ y $\hat{Y}(u+1)$ están en diferentes β cuadros. Por lo tanto la inserción de cuadros no marcados no afecta la sincronización del esquema propuesto.

Eliminación de cuadros

Sea $Y' = \langle Y(u), Y(u+1), ..., Y(u+\beta), Y(u+\beta+1)\rangle$ un segmento de video marcado. Suponga que una transición de estado ocurre después de $Y(u)$ y después de $Y(u+\beta)$. Un ataque de eliminación de cuadros se lleva a cabo sobre el subsegmento $S = \langle Y(u+1), ..., Y(u+\beta)\rangle$ generando $\hat{S}$, $|\hat{S}| = \delta$, $1 \leqslant \delta < \beta$. El segmento de video atacado es $\hat{Y'} = \langle \hat{Y}(t), \hat{Y}(t+1), ..., \hat{Y}(t+\delta), \hat{Y}(t+\delta+1)\rangle$. Aquí, $\hat{Y}(t) = Y(u)$ y $\hat{Y}(t+\delta+1) = Y(u+\beta+1)$. Suponga que se detecta la marca incrustada en $\hat{Y}(t)$, entonces el siguiente estado es almacenado en $q(0)$ (paso 8 de PDM). El siguiente cuadro a procesar es un elemento de $\hat{S}$ porque S fue marcado utilizando $\lambda(s(u+1))$. Entonces todos los cuadros de $\hat{S}$ fueron marcados utilizando $\lambda(s(t+1))$, el cual es almacenado en Q. Así, cuando el PDM procesa el siguiente cuadro $\hat{Y}(t+1)$, se logrará la detección (paso 3b del PDM). Si los siguientes cuadros de $\hat{S}$ existen, entonces la detección se realizará utilizando la misma llave (paso 5 del PDM). En el paso 8 del PDM, se genera un nuevo estado usando $\phi(s(t+1), K_E, \hat{F}_{\hat{S}})$, donde $\hat{F}_{\hat{S}}$ es el vector de características obtenido de los cuadros de $\hat{S}$. Este nuevo estado genera la marca incrustada en el último cuadro $\hat{Y}(t+\delta+1)$. De esta forma se logra la sincronización temporal.

Decimación de cuadros

El ataque de decimación es un ataque de eliminación de cuadros. La decimación con factor n mantiene el primer cuadro de n cuadros consecutivos. Así, la sincronización temporal se mantiene si, después de un ataque de eliminación, por lo menos un cuadro de una repetición β_i se mantiene, en caso contrario la sincronización temporal se pierde.

Intercambio de cuadros

En el ataque de intercambio de cuadros cambia la posición de los cuadros del video.

De esta manera, este ataque se puede tratar como una eliminación de cuadros si por lo menos un cuadro de una repetición β_i está en medio de dos cuadros consecutivos marcados con las condiciones mencionadas en el análisis de eliminación de cuadros. Sea $Y' = \langle Y(u), Y(u+1), ..., Y(u+\beta), Y(u+\beta+1)\rangle$ un segmento de video marcado y suponga que el estado de transición ocurre después de $Y(u)$ y $Y(u+\beta)$. Se lleva a cabo un ataque de intercambio de cuadros sobre un subsegmento $S = \langle Y(u+1), ..., Y(u+\beta)\rangle$ generando $\hat{S}$, $|\hat{S}| = \hat{\delta}$. Este ataque de intercambio de cuadros modifica la estructura temporal de una secuencia de video, ya sea a través de la inserción de un cuadro desde otra posición (fuera de la secuencia de video) o la eliminación de un cuadro local para cambiar su posición fuera de dicha secuencia de video. Entonces la magnitud de $\hat{\delta}$ puede ser $\hat{\delta} > \beta$ o $\hat{\delta} < \beta$ respectivamente. Así, el segmento de video atacado es $\hat{Y}' = \langle \hat{Y}(t), \hat{Y}(t+1), ..., \hat{Y}(t+\hat{\delta}), \hat{Y}(t+\hat{\delta}+1)\rangle$, donde, $\hat{Y}(t) = Y(u)$ y $\hat{Y}(t+\hat{\delta}+1) = Y(u+\beta+1)$. Suponga que la marca incrustada en $\hat{Y}(t)$ es detectada. Así el siguiente estado es almacenado en $q(0)$ (paso 8 del PDM). Suponga que por lo menos un cuadro del video S es un elemento de $\hat{S}$. El siguiente cuadro a procesar es un elemento de $\hat{S}$. Este cuadro $\hat{Y}(t+1)$ puede ser:

1. Un cuadro original de S.

2. Un cuadro atacado proveniente de una futura repetición β a la repetición actual, es decir, se insertó en la posición actual un cuadro marcado con una llave generada en el futuro.

3. Un cuadro atacado que proviene de una repetición β anterior a la repetición actual, es decir, se insertó en la posición actual un cuadro que fue marcado usando una llave generada previamente .

En el primer caso, ya que todos los cuadros en S fueron marcados utilizando $\lambda(s(u+1))$, recordando que $s(t+1) = s(u+1)$ y $q(0) = s(t+1)$, la detección se realizará para el cuadro $\hat{Y}(t+1)$ (paso 3b del PDM). Es importante mencionar que los estados esperados serán almacenados en Q. Además, en este caso se realizará la detección incluso si

el cuadro actual $\hat{Y}(t+1)$ fue atacado previamente con un ataque de intercambio intra-β, es decir el estado actual $s(t+1) \in s(u+2), .., s(u+\beta)$. Después de la detección, en el paso 8 del PDM, se genera un nuevo estado utilizando $\phi(s(t+1), K_E, \hat{F}_{\hat{S}})$. $\hat{F}_{\hat{S}}$ representa el vector de características de la secuencia $\hat{S}$. Este estado genera la marca incrustada en el último cuadro $\hat{Y}(t+\hat{\delta}+1)$, de esta forma se logra la sincronización temporal.

En el segundo caso, si el cuadro no fue marcado utilizando K_E, el cuadro no puede ser detectado porque su estado no se encuentra en la cola, es decir no ha sido creado.

En el tercer caso se logra la detección solamente si el estado está almacenado en la cola. La cola almacena los estados esperados en el futuro, pero debido al tamaño fijo de la cola, puede ocurrir que el último estado sea eliminado. Es importante mencionar que los elementos de la cola más cercanos a la cabeza son los más esperados. En este caso, la detección de la marca en el cuadro actual no garantiza la sincronización con el cuadro $\hat{Y}(t+\hat{\delta}+1)$. Para mantener la sincronización es necesario que por lo menos un cuadro de la repetición S original se encuentre antes del cuadro $\hat{Y}(t+\hat{\delta}+1)$.

Promedio de cuadros

Como se mencionó, el ataque de promedio de cuadros se puede realizar de dos maneras: por ventana y por promedio simple. Este ataque de promedio de cuadros es un ataque de Colusión, en donde se intenta eliminar la marca del medio marcado al promediarlo con otro medio marcado. La técnica de incrustación/decodificación usada en este esquema propuesto reporta la resistencia ante la colusión por promedio [CKLS97], con lo que se asume que se puede detectar la marca de un cuadro generado al promediar dos cuadros marcados. Es importante notar que el proceso de promediar cuadros modifica las características de cuadro generado, lo que puede provocar que el extractor de características obtenga un vector diferente. Si sucede esto, entonces el estado arrojado por el predictor no servirá para detectar la futura marca, con lo que el detector perderá sincronización. Con lo que se concluye que, la resistencia de este ataque depende de dos factores: la resistencia a colusión por promedio y la robustez del extractor de características. Estos factores son considerados por el esquema propuesto, y se comprueban

en la siguiente subsección.

Cambio de framerate

El cambio de framerate se realiza de dos maneras: 1) duplicación y eliminación, y 2) fusión de cuadros, como se menciona en la sección 3.2. Cuando se realiza el cambio de framerate por duplicación y eliminación, sucede lo mismo que un ataque de eliminación de cuadros. Si se lleva a cabo el cambio de framerate por fusión de cuadros, se toman las mismas consideraciones que el ataque de promedio de cuadros.

El análisis previo muestra el comportamiento del esquema propuesto ante diversos ataques temporales. Este esquema intenta mantener siempre la sincronización utilizando la llave maestra K_E. Así, la llave maestra se utiliza para comenzar y mantener la sincronización. La primera llave a detectar es la llave maestra y cuando se pierde la sincronización, el detector buscará la llave maestra en los siguientes cuadros para re-sincronizarse.

4.5. Comparación con el esquema base

El esquema propuesto está basado en el esquema de Lin [LD02, LD04, DL05]. Como se mencionó en la sección 2.3, el esquema de Lin lleva a cabo la sincronización temporal mediante la manipulación de las llaves de incrustación. Las llaves se cambian cada determinados cuadros utilizando las características de un cuadro particular. De esta manera, se genera un encadenamiento de llaves. Como se describió en este capítulo, esta misma idea es utilizada por el esquema propuesto para mantener la sincronización temporal. En la tabla 4.1 se muestra la comparación del esquema propuesto con el esquema de Lin. Se puede resumir que las principales diferencias entre el esquema propuesto y el esquema de Lin son:

- **El esquema propuesto utiliza una redundancia de repetición β adaptable, el de Lin utiliza una redundancia de repetición β fija**. La compresión en videos elimina la redundancia temporal y espacial presente en los cuadros. La redun-

dancia temporal se refiere a la similitud que hay entre cuadros consecutivos. Una regla en la compresión de video es: si un video tiene pocos cambios a través del tiempo entonces la compresión será elevada pues se elimina mucha redundancia temporal. Utilizar una redundancia de repetición β adaptable previene la eliminación de la marca ante la eliminación de redundancia temporal llevada a cabo durante la compresión. Adicionalmente, el usar la redundancia de repetición β adaptable sirve para segmentar cuadros consecutivos con características comunes, esto será útil durante la generación de llaves.

- **El esquema propuesto utiliza una redundancia de inicio de escenas γ, el de Lin no**. Como se mencionó en el capítulo anterior, la compresión MPEG clasifica los cuadros en tres tipos: I, P y B. Los cuadros tipo I son codificados como imágenes, los otros son codificados usando información de los cuadros tipo I. Sabiendo esto, los desarrolladores de esquema de marcas de agua buscan ocultar el mensaje en los cuadros tipo I pues son los que pierden menos redundancia en la compresión. Típicamente, los cuadros tipo I aparecen de forma regular en el video, pero en compresores avanzados como MPEG-4 parte 10, los cuadros tipo I suelen generarse de forma adaptable, por ejemplo después de cambios severos en cuadros consecutivos. Con la redundancia de inicio de escenas γ se busca ocultar el mensaje en cuadros tipo I, los cuales son generados al inicio de escenas. De esta manera, se pretende asegurar la detección de la marca en videos comprimidos con MPEG-4 parte 10.

- **El esquema propuesto genera una llave de incrustación usando características de β cuadros adyacentes, el de Lin usando sólo un cuadro**. El esquema de Lin genera una llave de incrustación usando las características de un solo cuadro previo. Esta manera de generar una nueva llave es vulnerable, pues si el cuadro que se usó para generar la llave se pierde o si es cambiado de posición entonces durante la decodificación no se podrá generar la siguiente llave esperada. La utilización de β cuadros adyacentes para generar la siguiente llave elimina esa

dependencia de un sólo cuadro. Durante la decodificación se genera la siguiente llave usando las características de los cuadros detectados usando la misma llave.

- **El esquema propuesto extrae características de β cuadros, el esquema de Lin de un sólo cuadro**. En el esquema propuesto la generación de llaves utiliza un extractor de características que analiza β cuadros. El esquema de Lin extrae características de un solo cuadro.

- **La extracción de características en el esquema propuesto se lleva a cabo usando promedio de regiones de una imagen y sus puntos característicos, en el esquema de Lin se lleva a cabo usando sólo el promedio de regiones de un cuadro**. El esquema propuesto genera un vector de 3 características de β cuadros usando las bajas frecuencias de los cuadros, el promedio de regiones y puntos característicos como se describió en la sección 4.2. El esquema de Lin obtiene un vector de 3 características promediando clases de regiones. La importancia de seleccionar características resistentes a modificaciones es vital, pues la mala selección de ellas puede generar llaves diferentes a las esperadas. El esquema de Lin al tomar en cuenta sólo el promedio es vulnerable a generar características diferentes después de un ataque de compresión MPEG y de promedio de cuadros.

4.6. Discusión del capítulo

El esquema propuesto ataca el problema de la pérdida de sincronización temporal mediante la generación de llaves. Dichas llaves son generadas tomando en cuenta las características espaciales y temporales de un video así como también una llave predecesora. Esta utilización de características y llaves generan un encadenamiento de llaves. Lo que provee de seguridad al esquema propuesto, ya que no se genera esa secuencia de llaves a priori.

Una de las principales características del esquema propuesto es que para generar una nueva llave se toman en cuenta características de un conjunto de cuadros y no sólo

Esquema propuesto	**Esquema de Lin**
Repetición β adaptable: el número de cuadros de la repetición β se obtiene analizando la similitud de cuadros adyacentes.	Repetición β fija: el número de cuadros de la repetición β se establece antes del proceso de incrustación.
Tamaño de redundancia α variable: el número de cuadros de la redundancia $alpha$ depende de las características del tamaño de la repeticiones beta que contenga, recordando que las repeticiones β son adaptables.	Tamaño de redundancia α fijo: el número de cuadros de la redundancia $alpha$ siempre es el mismo, puesto que las repeticiones β tienen un tamaño preestablecido.
Redundancia de inicio de escenas γ: consiste en ocultar el mensaje después del inicio de cada escena.	No hay redundancia de inicio de escenas.
Generación de llaves usando características de β cuadros adyacentes: para generar una nueva llave se utiliza un vector de características extraídos de β cuadros previos.	Generación de llaves usando características de un solo cuadro previo: para generar una nueva llave se utiliza un vector de características extraído del último cuadro de la repetición β anterior.
Extracción de características de β cuadros: Se analiza un conjunto de cuadros para obtener un vector de tres características que posteriormente se usará para generar una nueva llave.	Extracción de características de un sólo cuadro: Se analiza un solo cuadro para obtener un vector de tres características que será usado para generar una nueva llave.
Extracción de características usando promedio de regiones de imagen y puntos característicos: el módulo extractor de características obtiene un vector de tres características usando el promedio de 3 clases de regiones y la magnitud de los puntos característicos contenidos en las mismas regiones.	Extracción de características usando promedio de regiones de imagen: el módulo extractor de características obtiene un vector de tres características que representan el promedio de 3 clases de regiones.

Tabla 4.1: Tabla comparativa entre el esquema propuesto y el esquema de Lin

de uno, como es el caso del esquema base. De esta manera se fortalece la detección del esquema ante la pérdida de alguno o algunos de los cuadros del conjunto. Es importante destacar que la selección de cuadros se lleva a cabo midiendo la similitud entre ellos.

En el proceso de detección de la marca es importante destacar, que conforme se detecta el mensaje se garantiza el orden del mismo, es decir la detección de cada bit del mensaje es ordenada. Este orden en la extracción del mensaje se debe a que en el proceso de decodificación se descartan los cuadros en donde no se detecta alguna marca.

El análisis presentado detalló las condiciones necesarias para mantener el encadenamiento de estados que permite la sincronización temporal. Para ello se analizó como afecta cada ataque temporal al esquema propuesto y se evidenció que el esquema propuesto puede resistir ataques temporales.

En el siguiente capítulo se muestran diversos experimentos hechos al esquema propuesto con el fin de caracterizarlo. Los experimentos servirán para analizar el funcionamiento del esquema propuesto y para medir las propiedades principales: fidelidad, robustez y capacidad.

Capítulo 5

Resultados

En este capítulo se experimentará con el esquema propuesto. Los experimentos iniciales son usados para medir el funcionamiento del esquema propuesto durante el proceso de incrustación y durante el proceso de detección. Posteriormente, se muestran experimentos para medir las principales propiedades del esquema propuesto: fidelidad, robustez y capacidad de inserción. Finalmente se concluye con una discusión acerca de los resultados obtenidos.

5.1. Descripción de los experimentos

Los experimentos llevados a cabo tienen el objetivo de medir el funcionamiento del esquema ante la variación de parámetros usados en el proceso de incrustación/decodificación de la marca. Adicionalmente, se presentan experimentos en donde se miden las propiedades de fidelidad, robustez y capacidad de inserción que presenta el esquema propuesto. Además, se presenta una comparación del esquema propuesto en contra del esquema de Lin [LD02, LD04, DL05], el cual es el esquema que reporta los mejores resultados ante la pérdida de sincronización temporal.

Todos los experimentos que se describen en las siguientes secciones utilizaron el conjunto de videos sugerido por el VidMark. Este banco de videos está formado por

58 videos sin comprimir codificados usando el formato YUV 4:2:0 y con resoluciones CIF y QCIF (352×288 y 176×144 respectivamente). Para los experimentos de robustez se utilizó el VidMark, el cual se desarrollo en C usando las librerías FFmpeg y x264, las cuales sirven para manipular videos comprimidos. El esquema propuesto y los experimentos fueron programados en Matlab.

5.2. Funcionamiento

Los procesos PIM y PDM están conformados por una serie de módulos que interactúan entre sí para incrustar o decodificar un mensaje de un video respectivamente. Muchos de estos módulos realizan tareas específicas y bien estudiadas, como por ejemplo, el módulo de incrustación/detección de la marca que es en sí la técnica de espectro disperso. Otro ejemplo es el detector de escenas que es un método bien conocido y probado, basado en ventanas de cuadros consecutivos. Por otra parte, el generador de llaves es una FSM propuesta por Lin [LD02, LD04, DL05], todos estos módulos fueron detallados en el capítulo anterior. El funcionamiento del esquema propuesto está orientado a la generación redundancias (de repetición β, de escenas γ y de periodo α), las cuales se generan en PIM. Específicamente, la redundancia más utilizada es la redundancia de repetición, incluso es usada cuando se detecta una escena (es decir se usa con la redundancia de repeticiones de escena), además en la redundancia de periodo se vuelve a usar dicha redundancia de repetición para incrustar el mensaje nuevamente. Así pues, es importante experimentar con la redundancia de repetición β, para poder conocer el funcionamiento del esquema propuesto durante el proceso de incrustación.

En el PDM se analiza cuadro a cuadro para detectar la misma marca, teniendo un conjunto de cuadros consecutivos con la misma marca (estos cuadros son parte de la misma repetición β generada en la incrustación) se analizan en conjunto para obtener de ellos características que servirán para generar la siguiente marca, la cual será buscada en cuadros posteriores. El buen funcionamiento del PDM recae en la buena selección de características de un conjunto de cuadros. Así pues, es importante experimentar

con la extracción de características, para poder conocer el funcionamiento del esquema propuesto durante el proceso de decodificación.

En conclusión, el funcionamiento del esquema depende de dos módulos: 1) el generador de repetición β y b) extractor de características. Los experimentos siguientes muestran el funcionamiento de cada uno de estos módulos.

Objetivo de los experimentos

Medir la redundancia de repeticiones β para saber su tamaño promedio y mostrar la resistencia del detector de características ante diversos ataques.

Descripción de los experimentos

El primer experimento consiste medir la redundancia de repetición β. Específicamente, esta redundancia se obtiene al medir la similitud de cuadros consecutivos mediante la métrica de información mutua. Por tanto, el presente experimento consiste en variar el umbral de información mutua para determinar el tamaño promedio de las repeticiones β correspondientes.

El segundo experimento consiste en hallar los vectores de características de todas las repeticiones β de la colección de videos, luego atacar cada repetición ante eliminación y promedio de cuadros. Después, obtener nuevamente los vectores de características de las repeticiones. Finalmente, comparar gráficamente las características originales con las características obtenidas después de haber atacado las repeticiones, se espera una variación pequeña. Se seleccionaron los ataques anteriores pues representan a los otros ataques temporales, exceptuando al ataque de inserción (que no afecta a las repeticiones β, ya que se descartan en el proceso de detección) y de compresión, que es sabido que elimina frecuencias altas del video original. Esto es considerado por el extractor de características propuesto, pues realiza un filtro Gaussiano antes de procesar los cuadros.

Desarrollo de los experimentos

El primer experimento consiste en variar el umbral la información mutua U para determinar el tamaño de las repeticiones β correspondientes. Es importante mencionar que la información mutua entre dos cuadros es siempre mayor o igual a cero. Ésta, cuando tiene valor 0, indica que los cuadros son independientes entre si; y conforme incrementa de valor, la similitud entre los cuadros es mayor. De esta manera, cuando el umbral U es pequeño el tamaño de las repeticiones β es grande. Por otra parte, cuando el umbral es mayor, la cantidad de cuadros que forman las repeticiones β es menor. En la figura 5.1 se muestra este comportamiento. Se puede apreciar que cuando el umbral es cercano a 0, las repeticiones β son de tamaño superior a 400. Por el contrario, cuando es cercana a 3 el tamaño de las mismas es cercano a 2. En conclusión, este experimento presenta el comportamiento del tamaño de las repeticiones con respecto al umbral U. El usuario final del esquema propuesto puede utilizar el valor del umbral U que se adapte a sus necesidades. En los experimentos de este capítulo se utiliza un umbral arbitrario $U = 1.0$, que genera en promedio repeticiones de tamaño 10.

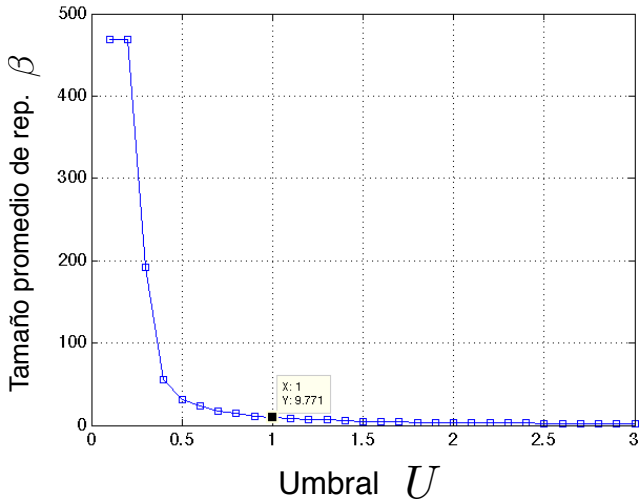


Figura 5.1: Tamaño promedio de las repeticiones β al variar el umbral de la información mutua entre cuadros

El segundo experimento consiste en extraer los vectores de características de cada repetición β de los videos del benchmark VidMark usando un umbral $U = 1$. Luego se obtuvieron las tres características $F1$, $F2$ y $F3$ de cada repetición. Finalmente, se selecciona la característica dominante de cada una y se hace un histograma de las características dominantes. La figura 5.2 muestra el histograma de características, de cada repetición, de la colección de videos. Se puede observar que la característica $F3$ es la que más domina mientras que la $F2$ se presenta con menos frecuencia.

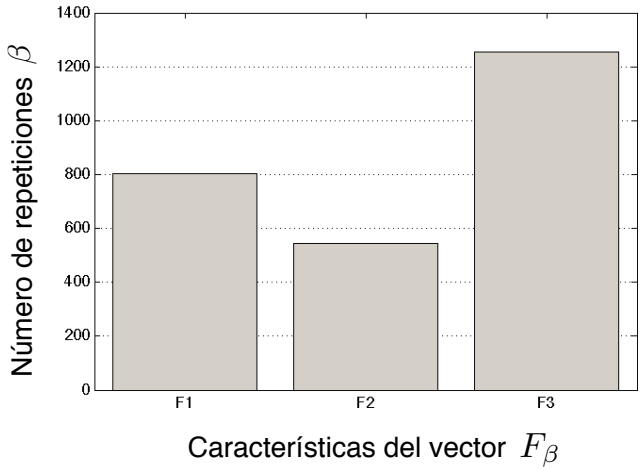


Figura 5.2: Características de las repeticiones β de los videos del VidMark

A continuación las repeticiones fueron sometidas a diversos ataques, como son: eliminación al 25 %, 50 % y 75 %; otro ataque es mantener sólo un cuadro de cada repetición mientras que los otros son eliminados; el último ataque es promediar toda la repetición y extraer las características de esa versión promediada. Las versiones atacadas de las repeticiones β son examinadas nuevamente por el extractor de características, las características obtenidas se observan en la figura 5.3. En esta figura se muestra el comportamiento del extractor de características, se puede observar que las características mantienen la relación con respecto a las originales, lo cual indica que las características seleccionadas se mantienen después de ser sometidas ante ataques de eliminación y promedio. El hecho de que las características no cambien cuando los cua-

dros son distorsionados genera robustez en el esquema propuesto ya que en el proceso de decodificación se evita predecir llaves erróneas.

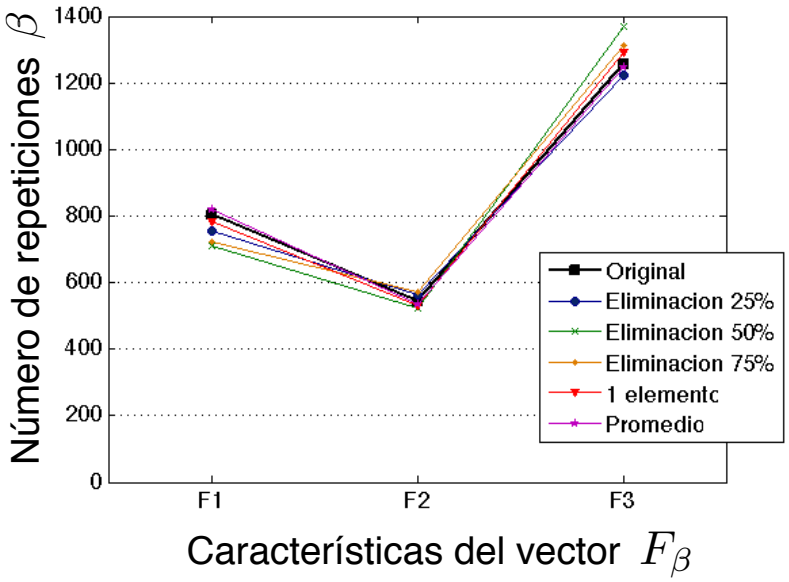


Figura 5.3: Características obtenidas al atacar las repeticiones β

Discusión de los experimentos

El funcionamiento del esquema depende principalmente de la redundancia de repetición β y de sus características detectadas. Los resultados arrojados por los experimentos mostraron que con sólo variar un umbral se puede establecer el tamaño de la redundancia de repetición a un valor establecido. Es tarea del usuario final elegir la opción que se apega más a la aplicación en donde será usado el esquema propuesto. En la sección 5.5 se describe el impacto de la redundancia de repetición en la capacidad de inserción del esquema propuesto.

La importancia de seleccionar características resistentes a modificaciones es vital en el esquema propuesto, pues la mala selección de ellas puede generar llaves diferentes a las esperadas. La generación de llaves erróneas en la detección incrementa la cantidad de llaves en la cola. Este incremento de llaves en la cola puede provocar la pérdida de

la llave almacenada en la última posición de esta estructura de datos. Dicha pérdida puede provocar una pérdida de sincronización temporal. Además, al incrementar el número de elementos de la cola, el decodificador intenta detectar la marca usando esta llave errónea, lo cual genera pérdida de tiempo en la decodificación. En conclusión, las características seleccionadas son resistentes ante los ataques de eliminación y promedio de cuadros como lo muestra el experimento.

5.3. Fidelidad

La fidelidad se define como la similitud perceptual entre el video marcado y el video original. El proceso de incrustación genera un video marcado, el cual representa al video original modificado con la información de un mensaje, la diferencia visual entre el video marcado y el original debe ser pequeña, si el video marcado se distorsiona en gran medida con respecto al video original no es útil. De esta manera, es importante conocer la distorsión que se genera al marcar un video. Esta distorsión visual es conocida como impacto perceptual.

La forma más aceptada de calcular este impacto perceptual es a través de la métrica PSNR, definida por 5.1

$$PSNR(dB) = 10 \log_{10} \left(\frac{255}{MSE} \right) \tag{5.1}$$

donde el Error Cuadrático Medio (MSE) entre dos videos X y Y es calculado por

$$MSE = \frac{1}{K * M * N} \sum_{k=1}^{K} \sum_{i=1}^{M} \sum_{j=1}^{N} (X(k)_{i,j} - Y(k)_{i,j})^2 \tag{5.2}$$

aquí, $K = |X| = |Y|$ es el número de cuadros de los videos, M y N representan el tamaño de los videos en píxeles. Es importante notar que en este caso particular el PSNR es de un sólo plano del video, específicamente del plano Y, descartando los planos U y V. Generalmente, en marcas de agua para video se desea un valor de PSNR mayor a 37 dB [BB04], puesto con este valor generalmente la distorsión provocada

no es notable a simple vista, con valores mayores en el PSNR obtenido la distorsión seguirá siendo imperceptible.

Objetivo del experimento

Medir el PSNR entre los videos originales y los videos marcados al variar el factor de robustez s, además determinar la magnitud de s que genera un PSNR cercano a 40 dB.

Descripción del experimento

Se marcarán los videos usando diferentes valores del factor de robustez s. Por cada valor de s se obtendrá el PSNR generado en los videos y se promediará. Los promedios de PSNR de cada s se graficarán.

Desarrollo del experimento

Puesto que el factor de robustez es la principal variable que modifica el impacto perceptual en el proceso de incrustación, se modificará su valor para poder caracterizar su comportamiento. Para ello, el rango de variación del factor de robustez se establecerá en el rango cerrado de [0.1,2], con dicho rango se puede caracterizar el comportamiento del esquema propuesto al variar s. La figura 5.4 muestra la fidelidad del esquema ante la variación del factor de robustez s. Se puede observar que cuando el factor de robustez es pequeño, el PSNR es elevado, cercano a 55 dB; conforme s aumenta de valor, el PSNR disminuye, se obtienen valores no deseados como por ejemplo con $s = 2$ en donde el PSNR es 28 dB. Con $s = 0.5$ el PSNR generado es 40 dB, con lo que el experimento para determinar el PSNR dado un valor s concluye.

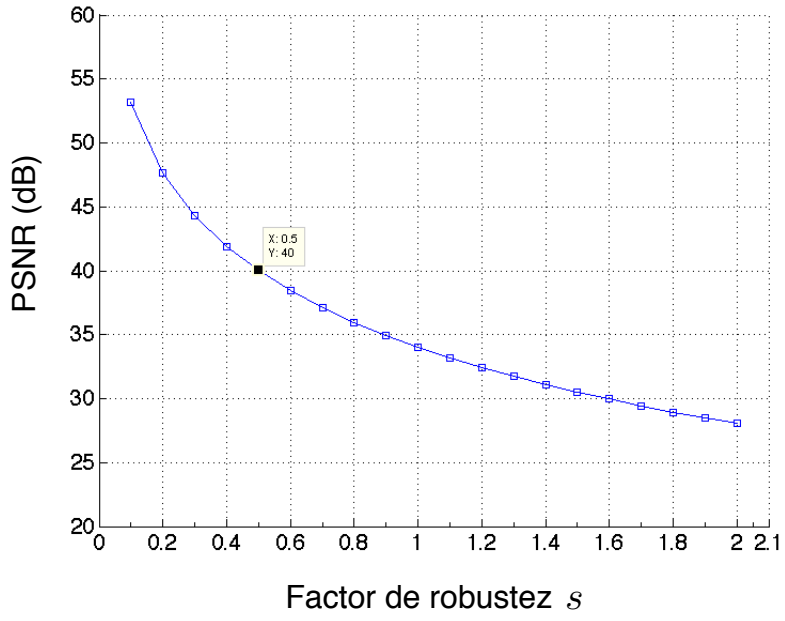


Figura 5.4: Fidelidad del esquema propuesto al variar s

Discusión del experimento

La fidelidad es un aspecto muy importante en esquemas de marcas de agua para video, incluso cuando se orientan a la robustez. Es sabido que existe un compromiso entre impacto perceptual y robustez. En este experimento se puso un límite en el impacto perceptual deseado, el factor de robustez se calculó para que durante la incrustación el medio marcado no se distorsione de gran manera. Es importante mencionar que el valor de s se adapta a la aplicación en donde será usado el esquema propuesto de marcas de agua. En este caso se tomó un valor que genera una distorsión que está dentro de los límites deseables.

5.4. Robustez

La robustez indica la capacidad de un esquema de marcas de agua para decodificar el mensaje incrustado en un video atacado. Comúnmente, la robustez se mide en tér-

minos del porcentaje del mensaje extraído correctamente del video marcado. Existen diversas métricas objetivas utilizadas, como lo son la correlación y la relación de bits correctamente detectados o BCR. Normalmente, la correlación es utilizada para medir la detección de una marca en el medio marcado, es decir, qué tanto de la marca se detectó. Aquí, por lo general la marca es un patrón pseudo aleatorio como el utilizado en espectro disperso. El BCR se utiliza generalmente cuando el mensaje representa bits de información. Esta medida indica el porcentaje del mensaje que se recuperó correctamente. Puesto que en estos experimentos se oculta un mensaje de 15 bits, se utilizará el BCR como métrica. El BCR se calcula como:

$$BCR = \left(1 - \frac{\sum_{i=0}^{n-1} M(i) \bigoplus M'(i)}{n} \right) \times 100\,\% \tag{5.3}$$

donde $\bigoplus$ es la operación XOR, M es el mensaje incrustado, M' es el mensaje extraído, $|M| = n$ es la longitud del mensaje incrustado. Aquí, $|M| = |M'|$.

Objetivo de los experimentos

Medir el rendimiento del esquema propuesto ante los ataques temporales: eliminación de cuadros, inserción de cuadros, intercambio de cuadros, decimación de cuadros, promedio de cuadros, cambio de framerate; y de compresión: MPEG-2, MPEG-4 parte 2 y MPEG-4 parte 10. Adicionalmente, comparar el esquema propuesto con el esquema de Lin en términos de robustez.

Descripción de los experimentos

En este experimento consiste en someter al esquema propuesto ante los ataques temporales definidos en el benchmark VidMark. El objetivo del experimento es medir la robustez del esquema propuesto ante la presencia de ataques temporales con parámetros diversos. Además se medirá la robustez ante ataques de compresión: MPEG-2,

MPEG-4 parte 2 y MPEG-4 parte 10. El rendimiento del esquema ante estos ataques es medido usando la métrica BCR. Finalmente, se comparará el rendimiento del esquema propuesto con el rendimiento del esquema de Lin en términos de robustez.

Desarrollo de los experimentos

Ataques del VidMark

Con el fin de respaldar el análisis hecho en el capítulo anterior y medir la resistencia contra ataques temporales, se evalúa el esquema propuesto utilizando los ataques proporcionados por el benchmark VidMark descrito en el capítulo 3. Los experimentos consisten en marcar los videos ocultando un mensaje de 15 bits en cada uno. Después los videos marcados son atacados temporalmente y por compresión utilizando el benchmark VidMark, finalmente se decodifica el mensaje de los videos atacados y se obtiene el BCR.

En la tabla 5.1 se muestra el desempeño del esquema propuesto contra ataques temporales. A continuación se presenta una discusión del comportamiento del esquema contra cada ataque:

- El ataque de inserción no afecta a la detección de la marca del esquema propuesto. Si la marca a detectar no se encuentra en la cola o no es K_E, el detector ignora el cuadro y trata de detectar la marca en el siguiente cuadro (paso 4 del PDM). Como se puede observar en la tabla 5.1, el ataque temporal de inserción no afecta la detección, el BCR en todos los casos es del 100 %.

- El ataque de eliminación de cuadros puede romper con la cadena de estados generados por $\lambda(\cdot)$. Se puede intuir que la resistencia del esquema propuesto ante este ataque depende de la redundancia de la repetición β. Cuando aumenta la magnitud del ataque, el riesgo de pérdida de la cadena de estados también se incrementa. A pesar de esto, el esquema propuesto puede detectar arriba del 60 % de la marca cuando la magnitud del ataque por eliminación es el máximo (50 %

Test	BCR (%)	Test	BCR (%)
Inserción 5 %	100	Promedio simple 5 %	100
Inserción 10 %	100	Promedio simple 10 %	100
Inserción 15 %	100	Promedio simple 15 %	100
Inserción 20 %	100	Promedio simple 20 %	94.24
Inserción 25 %	100	Promedio simple 25 %	88.52
Inserción 30 %	100	Promedio simple 30 %	84.02
Inserción 35 %	100	Promedio simple 35 %	76.86
Inserción 40 %	100	Promedio simple 40 %	70.12
Inserción 45 %	100	Promedio simple 45 %	64.74
Inserción 50 %	100	Promedio simple 50 %	66.46
Eliminación 5 %	100	Promedio ventana 2	40.12
Eliminación 10 %	98.08	Promedio ventana 3	24.03
Eliminación 15 %	96.10	Promedio ventana 4	15.46
Eliminación 20 %	94.28	Decimación 3	100
Eliminación 25 %	91.42	Decimación 4	100
Eliminación 30 %	82.85	Decimación 5	100
Eliminación 35 %	80.95	Decimación 6	100
Eliminación 40 %	75.23	Fps por duplicación 15	76.12
Eliminación 45 %	71.42	Fps por duplicación 24	100
Eliminación 50 %	62.85	Fps por duplicación 25	100
Intercambio simple 5 %	100	Fps por duplicación 30	100
Intercambio simple 10 %	99.04	Fps por fusión 15	36.64
Intercambio simple 15 %	95.23	Fps por fusión 24	100
Intercambio simple 20 %	94.28	Fps por fusión 25	100
Intercambio simple 25 %	90.47	Fps por fusión 30	100
Intercambio simple 30 %	91.42	Intercambio simultáneo 5 %	99.04
Intercambio simple 35 %	88.57	Intercambio simultáneo 10 %	98.09
Intercambio simple 40 %	88.57	Intercambio simultáneo 15 %	94.28
Intercambio simple 45 %	86.60	Intercambio simultáneo 20 %	96.19
Intercambio simple 50 %	85.71	Intercambio simultáneo 25 %	93.33
MPEG-2 100Kbps	58.45	Intercambio simultáneo 30 %	90.47
MPEG-2 200Kbps	87.24	Intercambio simultáneo 35 %	88.57
MPEG-2 300Kbps	98.86	Intercambio simultáneo 40 %	84.76
MPEG-2 400Kbps	100	Intercambio simultáneo 45 %	85.71
MPEG-4 parte 2 100Kbps	48.6	Intercambio simultáneo 50 %	80
MPEG-4 parte 2 200Kbps	80.12	MPEG-4 parte 10 100Kbps	36.68
MPEG-4 parte 2 300Kbps	97.29	MPEG-4 parte 10 200Kbps	82.57
MPEG-4 parte 2 400Kbps	100	MPEG-4 parte 10 300Kbps	94.28
Combinación	5.71	MPEG-4 parte 10 400Kbps	100

Tabla 5.1: Rendimiento del esquema propuesto ante los ataques del VidMark

de los cuadros son eliminados). Por otro lado, se obtiene un BCR del 100 % cuando la magnitud del ataque por eliminación es el mínimo (5 %).

- El ataque de decimación es similar a un ataque de eliminación en ráfaga, puesto que cuadros consecutivos son eliminados. Debido a la alta redundancia de repetición β, este ataque no afecta a la detección y el BCR en todos los casos es del 100 %.

- El ataque de intercambio de cuadros, simple o simultáneo, también puede romper con la cadena de estados, pero es posible repararla debido al uso de una cola en el proceso de decodificación. Esta cola es una memoria que almacena los siguientes estados a detectar (paso 8 del PDM). Así, cuando un cuadro es cambiado puede ser detectado utilizando el estado almacenado en la cola (paso 3b del PDM). Como se puede observar en la tabla, estos dos ataques son menos severos que el ataque de eliminación. En el ataque simultáneo, los resultados muestran en el peor caso un BCR de 80 % y cuando la magnitud del ataque es bajo se obtiene un BCR de 99.04 %, mientras que en el ataque simple es de 85.71 % y 100 % respectivamente.

- Los resultados obtenidos por el esquema propuesto ante el ataque de promedio de cuadros muestran que el promedio por ventana corrediza causa una gran pérdida de información, por otro lado los resultados ante el ataque de promedio simple muestran que el esquema resiste el ataque de buena medida, pues ante ataques menores o iguales a 15 % se detecta completamente la marca, mientras que en el peor caso se detecta el más del 65 % de la marca.

- Cuando se realiza el cambio de framerate por duplicación y eliminación, y el framerate debe ser incrementado, se realiza un ataque de inserción (utilizando cuadros marcados). Así, el resultado de este ataque no afecta la detección, es decir, el BCR en todos los casos es del 100 %. Por otro lado, cuando el framerate debe ser reducido, este ataque se comporta como un ataque de eliminación. En

este caso, cuando el framerate cambia de 25 a 15, el BCR es 76.16 %. El ataque por fusión de cuadros es similar al ataque anterior, pero considerando el ataque de promedio. Cuando el framerate debe ser aumentado, se realiza un ataque de inserción (utilizando cuadros marcados promediados), con lo que no se afecta la detección. Cuando el framerate debe ser disminuido, se lleva a cabo un ataque de promedio simple.

- El bitrate de la compresión MPEG influye en el impacto de los ataques espaciales y temporales del video marcado. A mayor bitrate menos agresivos son los ataques espaciales y temporales, de la misma forma, a menor bitrate los ataques temporales y espaciales son más severos. El bitrate también afecta la detección de la marca, como se muestra en la tabla, donde el porcentaje de detección se incremente a medida que el bitrate incrementa. Se puede observar que MPEG-4 parte 10 es más agresivo que las otras compresiones MPEG debido a que elimina más redundancia en el video.

- El ataque de combinación aplica dos ataques aleatorios consecutivos al video. El resultado del esquema propuesto ante este ataque depende de los ataques seleccionados. El BCR en este ataque es de 5.71 %. Es importante mencionar que los ataques seleccionados no fueron decimación e inserción, porque el BCR habría sido del 100 %.

Comparación con el esquema base

En la tabla 5.2 se muestra el desempeño del esquema propuesto comparado con el desempeño del esquema de Lin ante ataques temporales y de compresión. Se puede observar claramente que el esquema propuesto tiene un mejor desempeño ante los ataques de compresión MPEG.2, MPEG-14 parte 2 y parte10. Con respecto a los ataques temporales, el esquema propuesto tiene de igual manera un mejor rendimiento que el esquema de Lin, todo esto gracias a la utilización de las redundancias utilizadas.

Test	BCR (%)		Test	BCR (%)	
	E. Propuesto	E. Lin		E. Propuesto	E. Lin
Inserción 5 %	100	100	Promedio simple 5 %	100	100
Inserción 10 %	100	100	Promedio simple 10 %	100	100
Inserción 15 %	100	100	Promedio simple 15 %	100	95.64
Inserción 20 %	100	100	Promedio simple 20 %	94.24	90.1
Inserción 25 %	100	100	Promedio simple 25 %	88.52	85.72
Inserción 30 %	100	100	Promedio simple 30 %	84.02	79.64
Inserción 35 %	100	100	Promedio simple 35 %	76.86	70.19
Inserción 40 %	100	100	Promedio simple 40 %	70.12	56.87
Inserción 45 %	100	100	Promedio simple 45 %	64.74	46.39
Inserción 50 %	100	100	Promedio simple 50 %	66.46	44.05
Eliminación 5 %	100	100	Promedio ventana 2	40.12	40.72
Eliminación 10 %	98.08	97.21	Promedio ventana 3	24.03	34.12
Eliminación 15 %	96.10	95.84	Promedio ventana 4	15.46	19.81
Eliminación 20 %	94.28	93.6	Decimación 3	100	100
Eliminación 25 %	91.42	84.37	Decimación 4	100	100
Eliminación 30 %	82.85	79.45	Decimación 5	100	98.08
Eliminación 35 %	80.95	75.23	Decimación 6	100	98.08
Eliminación 40 %	75.23	71.56	Fps por duplicación 15	76.12	68.52
Eliminación 45 %	71.42	64.4	Fps por duplicación 24	100	100
Eliminación 50 %	62.85	62.13	Fps por duplicación 25	100	100
Intercambio simple 5 %	100	100	Fps por duplicación 30	100	100
Intercambio simple 10 %	99.04	99.04	Fps por fusión 15	36.64	31.71
Intercambio simple 15 %	95.23	95.23	Fps por fusión 24	100	100
Intercambio simple 20 %	94.28	93.84	Fps por fusión 25	100	100
Intercambio simple 25 %	90.47	89.01	Fps por fusión 30	100	100
Intercambio simple 30 %	91.42	88.57	Intercambio simultáneo 5 %	99.04	100
Intercambio simple 35 %	88.57	87.32	Intercambio simultáneo 10 %	98.09	97.96
Intercambio simple 40 %	88.57	86.6	Intercambio simultáneo 15 %	94.28	94.58
Intercambio simple 45 %	86.60	85.2	Intercambio simultáneo 20 %	96.19	92.1
Intercambio simple 50 %	85.71	83.74	Intercambio simultáneo 25 %	93.33	90.2
MPEG-2 100Kbps	58.45	38	Intercambio simultáneo 30 %	90.47	86.30
MPEG-2 200Kbps	87.24	56.7	Intercambio simultáneo 35 %	88.57	82
MPEG-2 300Kbps	98.86	86.7	Intercambio simultáneo 40 %	84.76	80.7
MPEG-2 400Kbps	100	100	Intercambio simultáneo 45 %	85.71	78.03
MPEG-4 parte 2 100Kbps	48.6	34.7	Intercambio simultáneo 50 %	80	75.17
MPEG-4 parte 2 200Kbps	80.12	42	MPEG-4 parte 10 100Kbps	36.68	26.7
MPEG-4 parte 2 300Kbps	97.29	73.3	MPEG-4 parte 10 200Kbps	82.57	40
MPEG-4 parte 2 400Kbps	100	100	MPEG-4 parte 10 300Kbps	94.28	66.7
Combinación	5.71	3.45	MPEG-4 parte 10 400Kbps	100	100

Tabla 5.2: Comparación del esquema propuesto contra el esquema de Lin ante los ataques del VidMark

Discusión de los experimentos

Los ataques realizados en estos experimentos reducen la calidad del video atacado. El ataque de eliminación de cuadros con alto grado de eliminación empobrece la calidad del video. De la misma forma, los ataques de intercambio de cuadros, de eliminación y de promedio de cuadros, producen un video notablemente borroso. Cuando el factor de ataque es bajo dichas distorsiones no son notables, excepto en el ataque de decimación, en donde se reduce notablemente el número de cuadros. Como se describió en la subsección anterior, el comportamiento del esquema ante los ataques temporales muestra una elevada robustez, en algunos casos el esquema propuesto resiste ataques al 100 %, como lo son ataques de inserción, decimación, promedio simple y cambio de framerate. Por otra parte, el esquema propuesto es sensible ante el ataque de promedio por ventana, en donde la detección del mensaje fue menor al 50 %. Es importante mencionar que este último ataque genera un video notablemente borroso, que para fines prácticos es poco útil.

Como se mencionó, durante la compresión MPEG, algunos cuadros son codificados internamente como si fueran imágenes (cuadros tipo I) y otros son generados tomando las características de otros (cuadros tipo P y B). Esta última clase de cuadros son distorsionados en gran medida, lo que puede provocar la pérdida de información si se oculta una marca en ellos. Esta clasificación de cuadros es explotada por el esquema propuesto, puesto que el mensaje también se incrusta en cada inicio de escena (redundancia γ). La razón de ocultar el mensaje en cada inicio de escena es para marcar cuadros tipo I, estos cuadros generalmente se generan después de un cambio de escena.

Como conclusión, el análisis realizado en la sección 4.4 mostró que el esquema propuesto puede resistir ataques temporales. Esto se comprobó experimentalmente mediante el uso del benchmark VidMark. Finalmente, una comparación en contra del esquema base nos mostró que l esquema propuesto es superior al principal esquema que realiza una sincronización temporal basada en llaves. El principal medio para soportar estos ataques es la redundancia de repetición β y el principal mecanismo de robustez

para soportar los ataques MPEG es la redundancia de escena γ. Si la sincronización se pierde, una resincronización se lleva a cabo usando la redundancia de periodo α.

5.5. Capacidad

La capacidad o carga útil indica el tamaño del mensaje que puede ser ocultado en un medio original. El esquema propuesto es un esquema para video basado en cuadros, es decir, se oculta la información cuadro a cuadro. Como se mostró en el capítulo 4, en cada cuadro se oculta un bit del mensaje. Dada esta característica, la capacidad máxima del esquema es igual a $|X|$, esto es el número de cuadros del video X.

El manejo de redundancia reduce la capacidad del esquema drásticamente. Puesto que el principal interés de la investigación propuesta es la robustez, la capacidad queda como segunda opción. Sin embargo, es posible calcular la capacidad promedio del esquema propuesto. Es importante hacer notar que la capacidad del esquema propuesto es dependiente de las características del video debido al uso de la redundancia β adaptable y a la detección de escenas del video.

Los siguientes experimentos fueron realizados usando la colección de videos del VidMark. Estos experimentos tienen el objetivo de determinar el promedio de bits que se puede ocultar en un cuadro. Es importante notar que los videos representan una pequeña muestra, por lo que los resultados reflejan sólo el comportamiento en ellos. Es importante recordar que los videos del VidMark son una colección de videos comúnmente utilizados en el área de procesamiento de videos.

Objetivo de los experimentos

Medir la capacidad de ocultamiento de esquema propuesto.

Descripción de los experimentos

Los experimentos se dividen en las siguientes etapas:

- Determinar la cantidad promedio de bits por cuadro que se pueden insertar en los videos del benchmark VidMark. Recordando que en cada repetición β se oculta un bit de información se propone obtener la cantidad de repeticiones β por video, luego normalizarla con la cantidad de cuadros de cada uno. En este experimento no se considera la redundancia de escena, pues se considerará en la tercer etapa.

- Calcular el número promedio de escenas de la colección de videos del benchmark VidMark.

- Calcular la cantidad de bits por cuadro considerando la redundancia de repetición y de escena, es decir, la capacidad del esquema.

Desarrollo de los experimentos

Redundancia de repetición β adaptable

Considerando que en cada repetición β se oculta un bit de información. El primer experimento consiste en calcular por cada video el número de las repeticiones β.

La figura 5.5 muestra el tamaño promedio de las repeticiones β por video. Se puede observar que existen dos videos con más de 80 repeticiones β, representados en la figura por los dos grandes picos. Estos videos contienen muchas repeticiones pues son los videos con la mayor cantidad de cuadros, alrededor de 1000. El promedio de cuadros de los videos de la colección es cercana a 300 cuadros. En esta figura también se puede observar que el número promedio de repeticiones es 30.7, indicado con una línea horizontal, lo que indica que la capacidad es 30.7 bits por video. Hay que notar que este número no indica de forma fiable la capacidad del esquema (sin considerar la repetición de escenas) porque se toman en cuenta videos enormes que aumentan la capacidad promedio. Para evitar esta situación se obtendrá la capacidad media por cuadro.

La capacidad del esquema propuesto, dada en bits/cuadros, es la razón entre el número de repeticiones β y el número de cuadros de cada video. En la figura 5.6 se muestra la relación entre el número de repeticiones β y los cuadros de los videos co-

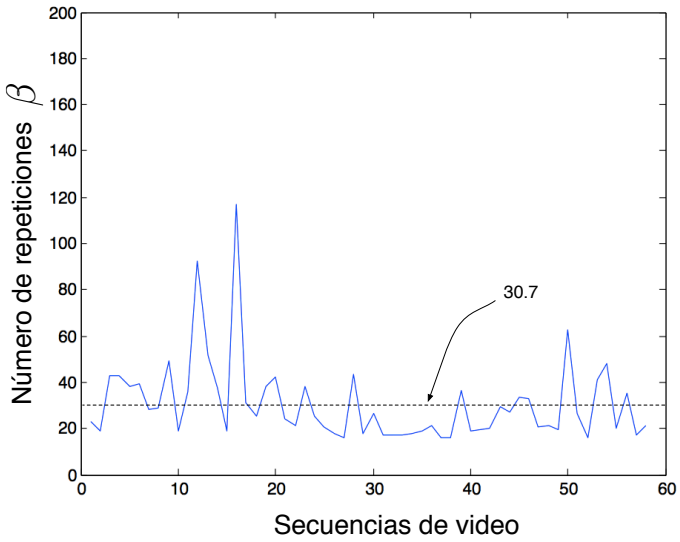


Figura 5.5: Número de las repeticiones β

rrespondientes. Esta relación indica de forma fiable la capacidad del esquema, pues las repeticiones β almacenan un bit de información. Así, la cantidad promedio de información que se puede incrustar en los videos es 0.11 bits por cuadro, como se indica en la gráfica. Se puede observar que los picos que se mostraban en la figura 5.5 se pierden pues esta razón es una normalización con respecto al máximo de cuadros por video. Es importante recalcar que esta capacidad es calculada sin tomar en cuenta la redundancia de escenas.

Tamaño promedio de escenas

Antes de considerar la redundancia por repetición para calcular la capacidad del esquema, es importante conocer la cantidad de escenas por video. La figura 5.7 muestra el número de cambios de escena detectados. Se puede observar que existen pocos cambios de escena, esto se debe a que la mayoría de videos son formados en una sola toma, y no existen cambios abruptos generados por un proceso de edición. El promedio de cambios de escena es 2.12.

La detección de cambio de escena es importante pues inmediatamente después de

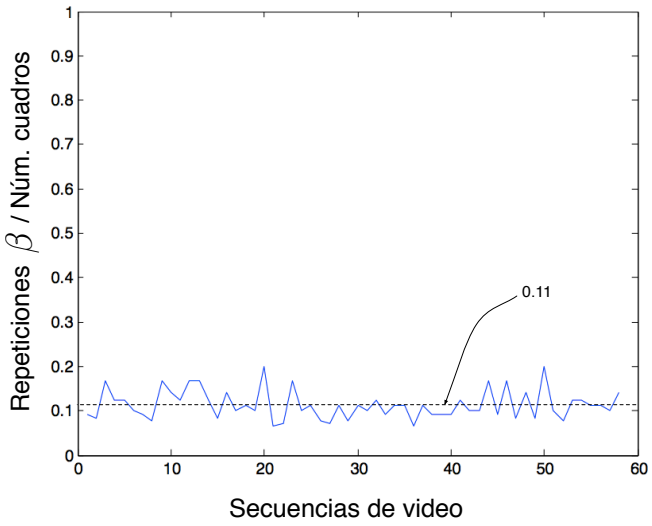


Figura 5.6: Numero de bits por cuadro, sin considerar escenas

conocerse, el control de redundancia temporal ordena que la siguiente repetición β oculte un bit de información propio de la redundancia de inicio de escena γ. Este proceso reduce la capacidad del esquema, pues a mayor redundancia menor capacidad de inserción.

Capacidad de inserción del esquema

La cantidad de bits promedio que se pueden ocultar en los videos de un conjunto dado se obtiene tomando en cuenta las redundancias β y γ. Para ello se obtiene la razón de la cantidad de bits diferentes ocultados en cada video con el número de cuadros respectivo. Es importante notar que no existe una relación entre el número de bits diferentes incrustados y la cantidad de repeticiones β detectadas, pues una repetición β puede ser parte de la redundancia de escena γ. La capacidad de inserción del esquema es definida por:

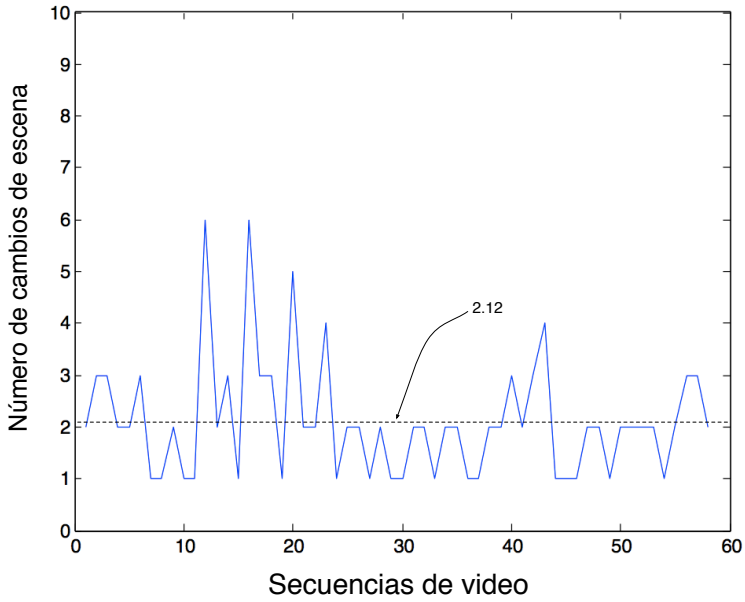


Figura 5.7: Número de cambio de escenas por video

$$\text{Capacidad(bits/cuadro)} = (\#\text{Repeticiones}\beta - \#\text{CambiosDeEscena})/\#\text{cuadros} \tag{5.4}$$

La figura 5.8 muestra la capacidad del esquema en bits por cuadro. La capacidad promedio es 0.104 bits por cuadro, aproximadamente se oculta 1 bit del mensaje cada 10 cuadros.

Un análisis realizado en un conjunto de 7418 videos (que incluye material cinematográfico, series de televisión, dibujos animados y documentales) sobre la duración de las escenas muestra que esta duración se ha reducido en la última década. Hoy en día la duración promedio de las escenas es de 7 segundos. La figura 5.9 muestra la duración promedio de escenas desde inicios del siglo pasado. Cabe destacar que si se toma en cuenta sólo material cinematográfico, la duración promedio de las escenas ha disminuido con mayor intensidad. Por ejemplo, y como se mencionó en el capítulo 3, la duración promedio de las escenas es de cuatro a seis segundos. La base de datos de

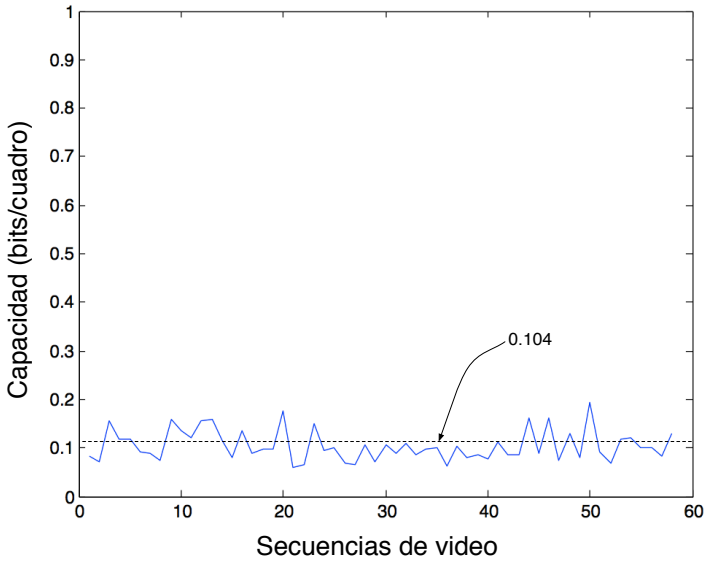


Figura 5.8: Capacidad del esquema propuesto

métricas de videos se encuentra disponible en [TC11].

Tomando en cuenta este análisis, se puede estimar la capacidad de inserción del esquema propuesto en cualquier tipo de video, usando la ecuación anterior 5.4, ya que se conoce la duración de las escenas promedio, que es de siete segundos, y que la duración promedio de repeticiones β es de diez cuadros. Por ejemplo en un video de 30 minutos con una tasa de 24 fps, la capacidad estimada es ((24*60*30)/10 - (60*30)/7)/(24*60*30) = (43200/10 - 1800/7)/43200 = 0.094 bits/cuadro. Es decir, se oculta aproximadamente un bit por cada 11 cuadros y en los 30 minutos se ocultan 4060 bits.

Discusión de los experimentos

La capacidad de inserción del esquema propuesto depende de la cantidad de repeticiones β y de la cantidad de detecciones de escenas, así como de la longitud del video. Los experimentos realizados, usando la colección de videos, muestran que la capacidad de inserción del esquema propuesto es en promedio 1 bit por cada 10 cuadros. Esto es, del orden de 144 bits por minuto en un video de 24 fps. Esta capacidad

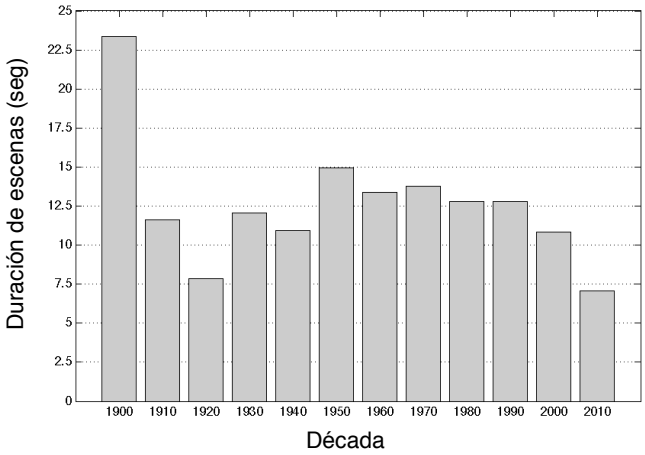


Figura 5.9: Duración promedio de escenas por décadas

es sólo considerando la redundancia de repetición β y de escena γ. Para poder resistir ataques temporales y de compresión se debe utilizar la redundancia de periodo α. Es decir, el mensaje debe ocultarse repetidamente a lo largo del video. Por ejemplo, en los experimentos de robustez se incrustaron 15 bits por video para explotar la redundancia de periodo α y así aumentar la resistencia ante las distorsiones temporales y de compresión llevadas a cabo.

5.6. Discusión del capítulo

Los experimentos realizados en este capítulo mostraron el funcionamiento del esquema propuesto ante diversas situaciones. Se mostró que las características tomadas en cuenta para generar una nueva llave resisten diversos ataques, lo que aumenta la detección del esquema propuesto al impedir que se inserten elementos erróneos en la cola. También se mostró que el tamaño de la repetición β impacta en la capacidad del esquema y en la robustez ante ataques de eliminación e intercambio simultáneo.

Se pudo establecer un umbral de distorsión, generado al incrustar la marca en los videos, es decir, se obtuvo la fidelidad del esquema al fijar el factor de robustez en la incrustación de la marca. Es importante mencionar que el mismo experimento puede

servir para que usuarios que deseen utilizar el esquema propuesto puedan adaptarlo a sus necesidades en cuanto a la distorsión generada. Para efectos de experimentación se seleccionó un PSNR de 40 dB.

Contando con el factor de robustez se procedió a probar la robustez del esquema ante los ataques del VidMark. La primera tarea consistió en analizar las condiciones en las que el esquema propuesto resiste los ataques temporales del VidMark, para ello se definió que la sincronización temporal consiste en mantener el encadenamiento de llaves en la detección hasta obtener el mensaje incrustado. Posteriormente se realizó una prueba experimental al atacar la colección de videos marcados usando los ataques del VidMark y posteriormente tratar de detectar la marca. Los resultados muestran la resistencia del esquema propuesto ante los ataques temporales.

La capacidad de incrustación del esquema se estimó tomando en cuenta la colección de videos del VidMark, los experimentos arrojaron que en promedio se oculta un bit por cada 10 cuadros de video. Sin embargo, al aumentar la robustez esta capacidad disminuye drásticamente, debido al uso de la redundancia de periodo α, es decir, al ocultar el mensaje repetidamente en el mismo video.

En conclusión, gracias a los experimentos realizados se caracterizó el comportamiento del esquema propuesto, esto da pauta para que los usuarios que opten por utilizar el esquema propuesto puedan adaptarlo para una aplicación específica. Hay que destacar que el esquema propuesto fue probado ante los principales ataques temporales reportados en la literatura. Finalmente, pese a que el esquema está orientado a la robustez, se puede utilizar un diferente esquema de marcado/decodificación para los cuadros para que se tenga una mayor capacidad de ocultamiento.

Capítulo 6

Conclusiones y trabajo futuro

En este capítulo se exponen las conclusiones de la tesis, la revisión de objetivos, las contribuciones generadas junto con el trabajo futuro.

6.1. Conclusiones

El resultado de esta investigación es un esquema de marcas de agua para video resistente ante pérdida de sincronización temporal y ante compresiones MPEG-2 y MPEG-4. Se realizó un análisis que muestra las condiciones bajo las cuales el esquema propuesto resiste ataques temporales. Para dar soporte a dicho análisis se llevaron a cabo pruebas experimentales usando el benchmark VidMark propuesto. Las principales conclusiones obtenidas son:

- La redundancia de repetición β impacta fundamentalmente en la carga útil del esquema y en la robustez del mismo. Naturalmente, a mayor redundancia de repetición menos información se puede ocultar, esto se debe considerar para la utilización del esquema propuesto en alguna aplicación específica.

- Las características de las repeticiones β son utilizadas para generar las llaves futuras, ya sea para incrustar o para detectar una marca. La importancia de haber

seleccionado características resistentes a distorsiones incrementa la generación de llaves correctas. El generar llaves erróneas provoca la saturación de la cola, lo cual puede generar pérdida de llaves, además de aumentar el costo computacional, pues el control de detección intenta detectar marcas usando las almacenadas en la cola.

- El esquema propuesto es resistente ante los principales ataques temporales y de compresión, como se mostró en los resultados obtenidos. El conocer las condiciones necesarias para soportar los ataques temporales aporta la información necesaria para poder diseñar el esquema propuesto y proporciona a priori el conocimiento para resistir los ataques temporales y de compresión. En general, el esquema propuesto resiste el ataque de inserción y decimación pues se detecta el mensaje al 100 % en todos los niveles de los ataques. Ante el ataque de eliminación obtiene arriba del 90 % del mensaje cuando se eliminan a lo más 25 % de los cuadros. El ataque de intercambio de cuadros tuvo un comportamiento similar al ataque de eliminación, pues ante las dos versiones de este ataque el esquema propuesto puede detectar arriba del 90 % del mensaje incrustado cuando se atacaban a lo más 30 % de los cuadros. Ante el ataque de promedio simple el esquema propuesto pudo obtener arriba del 90 % del mensaje cuando se promediaban a lo más 20 % de los cuadros. El ataque de promedio por ventana afecta la detección del esquema propuesto, pues en el mejor de los casos se detecta el 40 % del mensaje incrustado. Ante el ataque de cambio de framerate, el esquema siempre detecta el mensaje incrustado excepto cuando se reduce a 15 fps, aquí en el mejor de los casos se detecta el 74 % del mensaje. Finalmente, ante los ataques de compresión MPEG, el esquema propuesto llega a detectar el 100 % del mensaje incrustado.

- El esquema propuesto está orientado a la robustez, no obstante se puede utilizar en aplicaciones en donde las distorsiones sean conocidas. De tal manera, con modificar el valor de alguna variable se puede adaptar, en cierta medida, a las necesidades de la aplicación. Por ejemplo, se puede utilizar un esquema de marcas

para cuadros de mayor capacidad que el propuesto para aumentar la capacidad de inserción del esquema global.

- La propuesta del primer benchmark, para esquemas de marcas de agua para videos, especializado en ataques temporales es muy importante. En la actualidad pocos trabajos son diseñados para resistir ataques temporales. La creación de esta herramienta facilitará la comparación justa de los esquemas de marcas de agua para video que soportan ataques temporales. Además, la creación de este Benchmark permite la colaboración científica para enriquecerlo. El benchmark VidMark también sirvió para medir el rendimiento del esquema propuesto ante los principales ataques temporales y de compresión.

6.2. Revisión de objetivos

La investigación de esta tesis está centrada en el problema de la pérdida de sincronización temporal, para ello se diseñó un esquema de marcas de agua para video que soporta ataques temporales.

Recordando que el objetivo general era:

- Diseñar un esquema de marcas de agua para video, robusto ante ataques de pérdida de sincronización temporal y ante ataques de compresión MPEG-2 y MPEG-4.

Se concluye que éste fue alcanzado, ya que el esquema propuesto:

1. Incrusta un mensaje en videos planos que usan la codificación YUV 4:2:0.
2. Incrusta una marca en cada cuadro del video.
3. Utiliza tres tipos de redundancia: de repetición, de periodo y de escena.
4. La sincronización se basa en la generación de llaves y características de grupos de cuadros.

5. Resiste los principales ataques temporales.

6. Resiste los ataques de compresión MPEG-2, MPEG-4 parte 2 y MPEG-4 parte 10.

A partir de los puntos descritos anteriormente se logró diseñar un nuevo esquema resistente ante ataques temporales y de compresión, manteniendo la fidelidad estable. Adicionalmente, se observó que la capacidad de inserción del esquema propuesto depende de la redundancia de repetición β, la redundancia de inicio de escena γ y del tamaño del video mismo.

Los objetivos específicos fueron los siguientes:

- Diseñar un método que evite la pérdida de la sincronización temporal entre el detector de la marca y el mensaje incrustado en un video. Este método debe ser robusto ante ataques temporales incluyendo los llevados a cabo por las codificaciones MPEG-2 y MPEG-4.

- Crear un benchmark de ataques temporales para medir el rendimiento del esquema propuesto.

Se concluye que el primero fue alcanzado, ya que el esquema propuesto:

1. Utiliza las características de grupos de cuadros para generar nuevas llaves.

2. Utiliza las características del video para generar redundancia: de repetición, de periodo y de escena.

3. Resiste los principales ataques temporales y de compresión.

El segundo objetivo específico se alcanzó parcialmente ya que se inició el diseño de un benchmark de ataques temporales llamado VidMark. Este diseño de benchmark está dividido en una serie de módulos los cuales interactúan entre sí para poder obtener índices de rendimiento de esquemas de marca de agua para video. De los módulos

propuestos se ha creado el de ataques, los cuales se utilizarán para medir la robustez de los esquemas de marca de agua para video ante los principales ataques temporales. En resumen, se diseñó el esquema general del benchmark y se implementó el módulo de ataques temporales. Además se creó un sitio web para la divulgación y la colaboración global. Es importante recalcar que con lo implementado del benchmark es suficiente para evaluar un esquema de marcas de agua para video.

6.3. Contribuciones generadas

Las principales contribuciones de la investigación son:

1. Un extractor de características, de bloques de cuadros, que obtiene un vector de tres características resistentes ante ataques temporales.

2. Un generador de llaves que toma en cuenta características de bloques de cuadros, seleccionados de forma adaptable, para generar nuevas llaves

3. La creación de la redundancia de escena γ, que sirve para incrustar una marca en posibles cuadros tipo I que se generarán durante la compresión MPEG.

4. Un esquema de marcas de agua para video resistente ante ataques de pérdida de sincronización temporal y de compresión MPEG-2, MPEG-4 parte 2 y MPEG-4 parte 10.

5. La propuesta y el diseño del primer benchmark para esquemas de marcas de agua para video especializado en ataques temporales.

6.4. Trabajo futuro

Como trabajo futuro se puede realizar mejoras y análisis al esquema propuesto. Entre las mejoras que se proponen está realizar un análisis del generador de llaves para poder obtener el rango de llaves que se pueden generar. Con esto se puede determinar

la seguridad del esquema pues no hay trabajo alguno que asegure que las llaves no se repetirán en rangos definidos. Además, esto influye en la capacidad de inserción del esquema porque cada llave sirve para ocultar un bit de información.

Otro trabajo futuro es realizar análisis para mejorar la decodificación del esquema al meter un elemento a la cola. Se pueden inspeccionar políticas para el ingreso y posicionamiento de elementos en la cola, por ejemplo, la cola se puede ordenar considerando si la llave es parte de una cadena, dando más prioridad a aquellas llaves que forman parte de cadenas más grandes. Actualmente se da más prioridad a la última llave generada.

Además, se puede generar una cola de tamaño dinámico para evitar tener pérdidas cuando el número de llaves almacenadas sobrepase el tamaño de la cola.

El trabajo futuro más importante es continuar con el desarrollo del VidMark para tener la primera versión completa. Además, una importante contribución sería el realizar un análisis para determinar cuándo los ataques temporales dejan de ser útiles para el benchmark, es decir determinar los niveles de distorsión máximos permitidos para que los ataques no dejan inservible al video.

Apéndice A

Publicaciones derivadas

Las publicaciones derivadas de este trabajo de tesis, al momento de presentar el examen de grado, son [HAFUCGH09], [HAFUCGH10] y [HAFUCPGH65].

En [HAFUCGH09], se proponen las redundancias usadas por el esquema propuesto con el fin de soportar ataques de compresión. El nombre de esta publicación es *Video watermarking scheme resistant to MPEG compression*.

En [HAFUCGH10], se define diseño de un benchmark para esquemas de marcas de agua para video, llamado VidMark. El nombre de esta publicación es *Towards the Construction of a Benchmark for Video Watermarking Systems: Temporal Desynchronization Attacks*.

En la solicitud de patente Americana e Internacional [HAFUCPGH65], se describe el método de marcas de agua propuesto. El nombre de la invención es *Video Watermarking Method resistant to Temporal Desynchronization Attacks*.

A continuación se adjuntan las publicaciones correspondientes.

Video Watermarking Scheme resistant to MPEG Compression

Pedro A. Hernandez-Avalos, Claudia Feregrino-Uribe, Rene Cumplido,
Jose Juan Garcia-Hernandez
National Institute for Astrophysics, Optics and Electronics
Computer Science Department
Luis Enrique Erro No.1, Sta. Maria Tonantzintla, Puebla, Mexico C.P. 72840
{phernandez,cferegrino,rcumplido,jjuan}@inaoep.mx

Abstract

In this article an extension of the watermarking scheme, based on key sequence, proposed by Lin et al. is presented. This new scheme is designed taking into account the temporal modifications of the MPEG-2, MPEG-4 part 2 and MPEG4 Part 10 compressions. Besides the redundancy control of Lin is modified and the characteristics extraction process is improved. In this way, a better performance, up to 40% in the BCR (bit correct ratio) measure is gained. Experimental results show that the proposed scheme achieves a better performance against MPEG video compressions than Lin's scheme.

1. Introduction

Currently, the use of video watermarking is focused on protection issues where just a few bits are enough for copyright protection of content, nevertheless the use of schemes that embed more information opens the door for the use of the watermarking in emerging rich media applications like indexing, subtitles, hypervideo, interactive video, etc. [1]. The protection of videos is achieved usually by hiding a single watermark in a full video [2]. The watermarking schemes for media protection embed little information to ensure a greater probability of detection. Increasing the capacity of information embedding involves the use of more complex watermarking schemes, mainly to ensure the correct extraction of the watermark in the right order, this implies introducing a temporal synchronization phase (temporal synchronization is the process of identifying the correspondence between the temporal coordinates of the watermarked signal and the ones for watermark). Furthermore, it is important to mention that the video will be compressed, which causes the watermarking schemes to be robust to video compression attacks.

The video compression process removes temporal redundancy by using a motion compensation unit and a rate control unit. These two internal tasks are the main cause of temporal desynchronization on compressed videos. The temporal synchronization is crucial for successfully detecting watermarks. If the detector cannot be synchronized with its input video, an embedded watermark cannot be detected even though it is present in the video. The video compression schemes attack the watermark hidden in each frame and desynchronize in time the detector due to the mechanism used to reduce the redundancy in the video. Thus, if two or more frames hide a single watermark (regardless of the reasons), then a temporal synchronization process is necessary to resist the video compression attack.

In order to achieve temporal synchronization and resistance to video compression, some watermarking schemes, like [3, 4], hide information by changing slightly the magnitudes of the motion vectors of an entire scene. To perform this, motion vectors are classified in different groups, then a bit of information is embedded in each one. In this way, a bit is hidden within a group of vectors with similar characteristics. Note that when using this internal task of MPEG compression [5, 6, 7] (the generation of motion vectors), these schemes provide robustness to this attack. This could be seen also as a disadvantage, because if the video is re-encoded, the watermark can be lost. Another scheme hides information using other inner process carried out into the compression process like the use of the DCT coefficients in the MPEG-4 compression [8]. Here, spread spectrum is used to hide the same watermark in a subset of DCT blocks per video scene. This scheme depends on the internal task performed by the encoder. Lin's scheme hides informa-

tion on the raw video, which makes it independent of the encoder. However, it is not resistant to compression attacks. The proposed scheme consists on improving Lin's scheme to make it robust to MPEG-2, MPEG-4 Part 2 and MPEG-4 part 10 compression attacks. It is important to mention that this scheme present the best resistance against the more common temporal synchronization attacks.

2. Scheme proposed by Lin *et al.*

The Lin's scheme [9, 10, 11] performs temporal synchronization by manipulating the embedding keys used for watermarking each frame. In this way, the watermark, that is embedded in each frame, carries along temporal information. The key is changed every β frames using the information of the current frame and the previous key, thereby a new key time-dependent of the current and the previous frame is generated. To perform key generation a finite state machine (or FSM) is used (figure 1). The key generation is achieved in the following way:

1. The FSM is restarted (state 0) and the current embedding key K_i is set to K_e (the master key), then the embedding process generates a watermarked frame $Y(i)$.

2. The characteristics X, Y and Z are obtained from $Y(i)$ and they are used to define the next state in the FSM.

3. The next embedding key K_{i+1} (used to watermark the frame $X(i+1)$) is computed using the state equations shown in the figure 1.

4. The next frame is obtained and the embedding key K_{i+1} is used to watermark it. The steps 2, 3, 4 are carried out, using this new frame, until all frames are generated and watermarked.

There is also a mechanism for generating a period redundancy α (indicates the maximum number of frames that will be watermarked until the use of K_e) and a repeat redundancy β (indicates the amount of frames watermarked using the same key).

Figure 2 shows an example of a key sequence with $\alpha = 8$ y $\beta = 2$. The same color indicates that the same key was used to hide the watermark. It can be noticed that every two frames the key is changed and every eight frames the key is restarted. Moreover, the keys used in the frames 2 and 3 are not the same as those used in the frames 10 and 11. During a period α the only key that remains unchanged is the master key K_e. The frames watermarked with the master key are

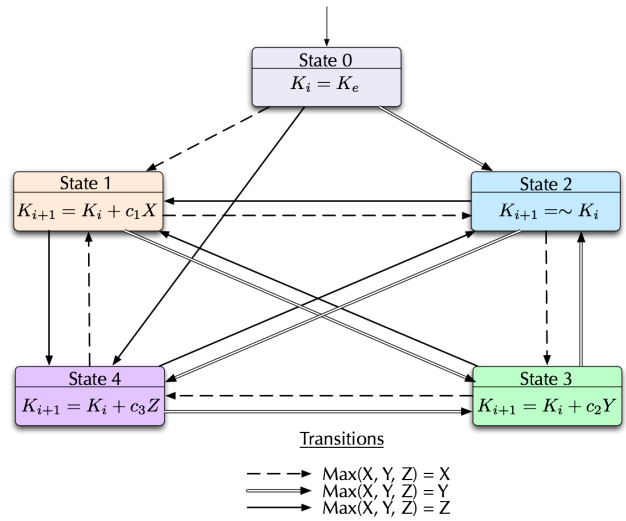


Figure 1. Finite state machine used to generate new keys

called resynchronization frames, which are used as a reference when the detector loses synchronization. The period given by α controls the degree of overall redundancy and determines how many frames the key generator has restarted. The repetition β controls the degree of local synchronization. A high degree of repetition β increases the redundancy and decrements the generation of new keys, which is an advantage to resist temporal attacks such as frame dropping, frame transposing and frame averaging. Moreover, if the degree of repetition is small there is little or no redundancy, which means that when a single watermark is not detected the synchronization period is completely lost.

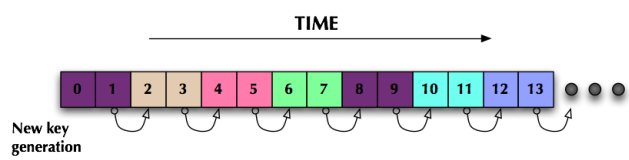


Figure 2. Key sequence with $\alpha = 8$ y $\beta = 2$

In the watermark detection process, the main task is carried out by the finite state machine and by a queue that keeps track of frames not synchronized. A null queue is generated first, and then the key K_e is used for continuously extracting the watermark from the frames. Once this watermark is correctly detected, the predicted state-keys (using the same FSM as in the embedding, figure 1) is pushed into the queue. Then watermark extraction of each frame is repeated with

the keys K_e and those reserved in the queue. When a watermark is detected, the current state-key will be moved to the front of queue if it does not belong to the initial set. Then the predicted state-keys of current state-key is pushed into the queue. It is important to mention that key generation in the embedding and in the extracting processes depends on the use of the FSM. It is important to mention that when the magnitudes of X, Y and Z (from a frame) are close during the embedding process and $Y(i)$ is attacked, the FSM transition (in the extracting process) can be different from the expected one, causing the temporal desynchronization.

3. MPEG video compression

In order to gain robustness in Lin's scheme to MPEG-2, MPEG-4 part 2 and MPEG-4 part 10 compression, three improvements are proposed. These MPEG standards include three kinds of frames: 1) intra picture frames (I-frames); 2) forward-predicted frames (P-frames); 3) bidirectional-predicted frames (B-frames). A video stream or recording will always start with an I-frame and will typically contain regular I-frames throughout the stream. These regular I-frames are crucial for the random access of recorded MPEG-4 files, such as with rewind and seek operations during playback. The disadvantage of I-frames is that they tend to compress much less than P-frames or B-frames [12].

I-frames are coded without reference to other frames, they are coded like an image. P-frame applies motion prediction by referencing an I-frame or P-frame in front of it, motion vector points to the block in the referenced frame. B-frame applies motion prediction, referencing a frame in front of it and/or a frame behind it. Each of the two referenced frames may be I-frame or P-frame. Macro block (MB) in video stream is represented as a 16×16 sample area. Each MB contains six 8×8 blocks, four for luminance and two for chrominance. A block of an I-frame contains six 8×8 blocks, four for luminance and two for chrominance. A block of an I-frame contains simply values of luminance or chrominance of its own. A block of a P-frame or B-frame contains the difference between the values of itself and the referenced block. This process is called motion compensation. Each frame is divided into MBs. Coding process of each block includes DCT, quantization, run length encoding and entropy coding in that order. The resulting video stream consists of entropy codes, motion vectors and control information about the structure of video and characteristics of coding. The general frame sequence of MPEG videos (MPEG-2,MPEG-4 part2 and MPEG-4 part 10) can be seen in figure 3.

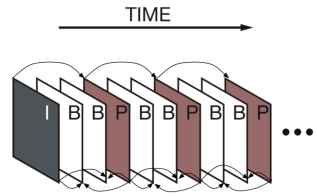


Figure 3. General MPEG frame sequence

4. Modified scheme

The tree proposed improvements to Lin's scheme are:

1. To perform an adaptable β repetition.
2. To modify the characteristic extractor in order to consider β frames.
3. To modify the temporal redundancy in order to generate redundancy in I-frames.

In the described scheme thus far, a fixed β repetition is utilized. However, this can be counterproductive due to MPEG-2, MPEG-4 part 2 and MPEG-4 part 10 compressions strongly attack temporal redundancy through motion compensation. If a frame sequence belongs to an scene with little movement, this compensation destroys part of these frames. If frame destruction happens and the magnitude of β is small, the watermark hidden in that repetition will be lost, which causes temporal desynchronization (the following keys will be lost due to they are related to the previous keys). The proposed improvement is to hide the same watermark in a number of frames given by the mutual information between them. So if similar frames the mutual information is high and disimilar frames the mutual information is low, in this way a measure of similarity in time can be utilized. The metric of mutual information is given by

$$I(X,Y) = \sum_x \sum_y p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) \qquad (1)$$

where $p(x,y)$ is the probability mass function *or pmf* of the random variables X and Y (i. e. frame X and frame Y), $p(x)$ and $p(y)$ represents the marginal pmf of X and Y, respectively.

To determine the amount of frames watermarked with the same key, a threshold U is used, in this case

$U = 1$. Futhermore, a limit in the number of consecutive frames to be watermarked is established to 20. An example of this improvement is shown in figure 4, the values of β are 4, 2, 4, 3, 2 and 2. These values indicates that the first four frames are similar, the next two frames are similar too, the next four frames are similar again and so on. The dissimilarity between the consecutive frames determines the end of each β frames.

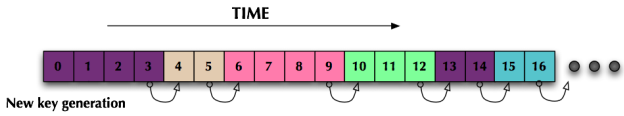


Figure 4. Keys generated using a repetition β adaptive

The second improvement consists on modifying the characteristics extractor used in the Lin's scheme. The extractor obtains the characteristics from the watermarked frame (the average of frame set is being used so far) and uses them to decide the route through FSM transitions. It is important to notice that the extractor invokes the last watermarked frame of each β repetition, which generates a dependence between the frame and the generated key. In this manner, if the last frame of repetition is eliminated during detection, the characteristics of another frame will be taken into acount to generate the new key; which causes that the new key is not the expected one. The proposed improvement consists on taking all the frames watermarked with the same key to extract their characteristics. An image that represents all the frames within β repetition is obtained by averaging the low frequencies of the watermarked frames (utilizing a Gaussian filter with $\sigma = 4$). By adding the magnitudes of the feature points (using the Harris Detector [13]) to the X, Y and Z characteristics (obtained with Lin's original method), Lin's scheme is improved due that the new X, Y and Z characteristics are no longer close and this prevents the erroneous transition in the FSM.

In this way, the proposed extractor is not dependent of a previous frame, thus the resistance against frame dropping is improved. Figure 5 shows this behavior.

Finally, the third improvement consists on modifying the temporal redundancy control to hide a watermark in each posible I-frame(see figure 6). It is worth to mention that MPEG compression encodes some frames internally as if they were images (I-Frame), some others using motion estimation taking I-frames as basis (P-Frame) and finally some others using motion estimation taking P-frames as basis (B-Frame). In

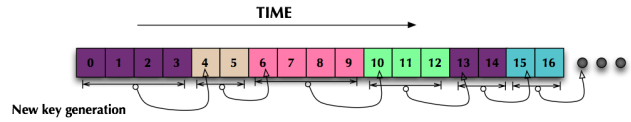


Figure 5. Keys generated using a characteristics extractor that use a set of β frames

this manner, it pretends to hide the watermark on each posible I-frame to avoid the aggresive attack of motion estimation. Also, I-frames are generally created by the codificator when a shift of scene is detected. In this way, the aim is to generate redundancy of keys utilizing the shift of scene (the end of each scene is obtained using the metric of mutual information from equation 1). In this manner there is redundancy of scene γ. The idea behind this new redundancy is to guarantee detection of at least one bit for each I-frame. With these modifications, the scheme is able to support MPEG-2, MPEG-4 part 2 and MPEG-4 part 10 compressions.

It is important to mention that the period redundancy α (unknown at the begining of the embedding process) depends on the amount of information to be embedded and on the size of β (adaptable). For example, if five bits are embedded and the values of β are: 3, 4, 3, 5 and 4, $\alpha = 19$. Finally, the proposed scheme uses two α values, one for embed ding the information in a sequential way (like Lin's scheme) and the other for embedding the information at the begining of each scene (thirth improvement). Finally, key generation on the proposed scheme, in both embedding and extracting processes, is performed as that of Lin's scheme.

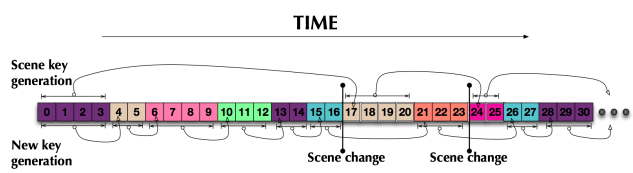


Figure 6. Key generation redundancy per scene

5. Experimental results

In order to evaluate the resistance against compression attacks, the proposed improvements described in section 4 were performed. Seven videos of 352×288 pixels, with a range of 870 to 2101 frames at 25 frames per second (fps) were used. The experiments carried

out consist on testing these videos with different *bitrates* and detecting the hidden watermark, measuring the bit correct ratio (or BCR). The utilized parameters, for Lin's scheme are $\alpha = 150$ and $\beta = 10$, in other words, 15 bits were hidden and the size of the queue is 10 entries.

Figure 7 shows the performance of both schemes, Lin's and the proposed one, against compression attacks. The performance of the proposed scheme overcomes Lin's scheme in all but the case of encrypted video at 400 Kbps bitrate where they perform similarly. It can be noticed that both schemes are affected by MPEG-4 part 10 compression, since this achieves a better reduction of temporal and spatial redundancy mechanism (bitrate control). However, the detection of the watermark using the proposed scheme is up to 40% better that Lin's scheme against the MPEG-4 part 10 compression at 200 Kbps. The bitrate influences the impact of temporal and spacial attacks during compression. The higher the bitrate the less agressive the spatial and temporal attacks, and the lower the bitrate the less severe the spatial and temporal attacks. The bitrate also affects the watermark detection, as shown in figure 7, where the detection percentage improves as the bitrate increases. It is worth to mention that Lin's scheme as well as the proposed scheme utilize a basic spread spectrum technique to embed the watermark as indicated by equation 2.

$$y_i = x_i + sw_i \tag{2}$$

Where y_i is the watermarked i frame, x_i is the original i frame, s is a robustness factor and w_i is a watermark. The watermark is a random variable with normal distribution $N(0,1)$. Detection is performed by using a low pass filter to reduce the effect of the original image (as suggested in [14]) followed by correlation with the watermark signal. The correlation is carried out as shown in equation 3.

$$\begin{gathered} c = E\{\mathbf{y} \cdot \mathbf{w}\} \\ \text{if } c \approx s, \text{ the watermark exist} \\ \text{else } c \approx 0, \text{ the watermark does not exist} \end{gathered} \tag{3}$$

6. Conclusions

A modified state transformation scheme of key sequence for video watermarking robust to MPEG-2, MPEG-4 part 2 and MPEG-4 part 10 is proposed. The proposed modification of redundancy mechanism in embedding process increases the resistance to the main widely used video compression schemes. Experimental results indicate that the proposed scheme has a better resistance to MPEG compression than Lin's scheme.

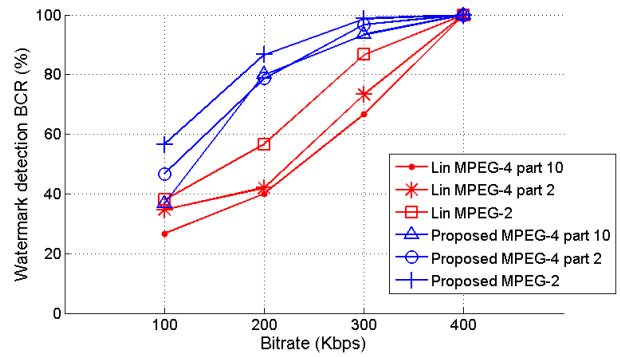


Figure 7. Performance against MPEG-2, MPEG-4 part 2 y MPEG-4 part 10 compressions

As future work, an analysis of the influence of the watermark embedding in the bitrate control mechanism can be performed. In addition, embedding multiple watermarks in a frame to increase the capacity scheme can be performed.

Acknowledgments: The authors acknowledge to CONACYT the support provided through the grant for PhD studies number 174444 and the project grant CB-2007-1-84668.

References

[1] C. O. Dumitru, S. Duta, M. Mitrea, and F. Preteux, "Gaussian hypothesis for video watermarking attacks: Drawbacks and limitations," *The International Conference on "Computer as a Tool" EUROCON, 2007*, pp. 849–855, Sep. 2007.

[2] L. E. Coria-Mendoza, *Low-Complexity methods for image and video watermarking*. PhD thesis, The University of British Columbia, Vancouver, Canada, Mar. 2008.

[3] L. he Zhang, G. Xu, J. jun Zhou, and L. jie Shen, "A video watermarking scheme resistant to synchronization attacks based on statistics and shot segmentation," *Seventh International Conference on Intelligent Systems Design and Applications. ISDA 2007*, pp. 603–608, Oct. 2007.

[4] L. Kezheng, Y. Wei, and L. Pie, "Video watermarking temporal synchronization on motion vector," *3rd International Conference on Intelligent System and Knowledge Engineering. ISKE 2008.*, vol. 1, pp. 1105–1110, Nov. 2008.

[5] Moving Pictures Expert Group, *ISO/IEC 14496-2*.

[6] Moving Pictures Expert Group, *ISO/IEC 14496-10*.

[7] Moving Pictures Expert Group, *ISO 13818*.

[8] C. Chen, T. gang Gao, and L. zong Li, "A compressed video watermarking scheme with temporal synchronization," *Congress on Image and Signal Processing*, vol. 5, pp. 605–612, 2008.

[9] E. T. Lin and E. J. Delp, "Temporal synchronization in video watermarking," in *Proceedings of the SPIE International Conference on Security and Watermarking of Multimedia Contents IV*, vol. 4675, (San Jose, California, U.S.A.), pp. 20–25, 2002.

[10] E. T. Lin and E. J. Delp, "Temporal synchronization in video watermarking," *IEEE Transactions on Signal Processing*, vol. 52, pp. 3007–3022, Oct. 2004.

[11] E. J. Delp and E. T. Lin, *Optical and Digital Techniques for Information Security*, vol. 1, ch. Watermarking Streaming Video: The Temporal Synchronization Problem, pp. 135–153. Springer New York, 2005.

[12] M. Smart and B. Keepence, "Understanding mpeg-4 video," White paper IC-COD-REP012, IndigoVision, Jun 2008.

[13] C. Harris and M. Stephens, "A combined corner and edge detection," in *Proceedings of The Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[14] T. Kalker, G. Depovere, J. Haitsma, and M. Maes, "A video watermarking system for broadcast monitoring," in *In Proc. SPIE Security and Watermarking of Multimedia Contents I*, vol. 3657, (San Jose, California, U.S.A.), pp. 103–112, Jan. 1999.

Towards the Construction of a Benchmark for Video Watermarking Systems: Temporal Desynchronization Attacks

Pedro A. Hernandez-Avalos, Claudia Feregrino-Uribe, Rene Cumplido and Jose Juan Garcia-Hernandez
National Institute for Astrophysics, Optics and Electronics
Luis Enrique Erro No.1, Sta. Maria Tonantzintla, Puebla, Mexico C.P. 72840
Email: {phernandez,cferegrino,rcumplido,jjuan}@inaoep.mx

Abstract— **Although several digital watermarking schemes have emerged as a solution to traditional copyright protection technologies, only a few benchmark suites, specialized in still images and audio, have been presented to measure the robustness and performance of these schemes. The aim of this paper is to single out the particularities in the way of evaluating the performance of video watermarking systems to generate a video watermarking benchmark framework specialized in temporal desynchronization attacks. In this way, this paper presents the most important temporal desynchronization attacks and performance measures for video watermarking systems.**

I. I

Today, the research in digital watermarking is growing rapidly, thus many watermarking schemes have been developed. The way to evaluate and compare them is converging to the utilization of benchmark suites to measure up its performance against various types of attacks [1]. The growing number of attacks has shown the importance of efficient and reliable benchmarking to improve the quality of existing watermarking methods. Image and audio watermarking evaluation approaches and benchmarking suites have been described in the literature, however they have neglected video watermarking.

Nowadays, the main benchmark suites are Stirmark [2], [3]: specialized in images an audio attacks; and Checkmark [4], Optimark [5] and WET (Watermark Evaluation Testbed) [6]: all of them specialized in images attacks. Although there are plans to add video attacks to the Checkmark benchmark (such as frame editing, frame reordering, aspect ratio changes, frame rate changes, collusion and averaging attacks as well as all applicable image processing attacks which can be applied to video frames) they do not include the minimum temporal attacks needed to evaluate video watermarking schemes. The video watermarking schemes are vulnerable to another kind of attacks due to the third dimension that it presents, the time.

Currently, the use of video watermarking is focused on protection issues where hiding a single watermark or hiding a few bits is enough for copyright protection of content, nevertheless the use of schemes that embed more information opens the door for the use of watermarking in emerging rich media applications like indexing, subtitles, hypervideo, interactive video, etc. [7]. Embedding little information ensures a greater probability of detection. Increasing the capacity of embedding information involves the use of more complex watermarking schemes, mainly to ensure the correct extraction of the watermark in the right order, this implies introducing a temporal synchronization phase. These new applications of video watermarking are another motivation for creating a benchmark of temporal attacks to evaluate and fairly compare video watermarking schemes.

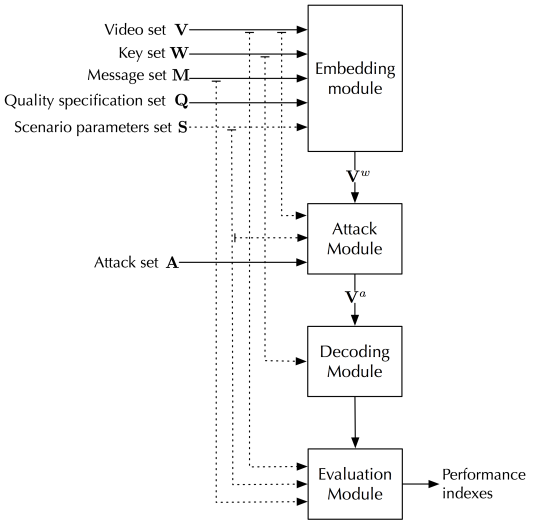


Fig. 1. Watermarking benchmark system.

II. V W B

Like in Optimark, the inputs to the video benchmark system are the embedding and decoding algorithms, and the outputs are the performance indexes that illustrate the behaviour of the watermarking system against sev-

eral temporal attacks. Internally, the benchmark system has a fixed data set (video, keys, messages, etc.) to perform a fair comparison.

The general scheme of a video watermarking benchmark system is shown in figure 1. The benchmark system comprises the watermark embedding module, the attack module, the decoding module and the performance evaluation module. In the embedding module a message M_k is embedded into a video V_i using a key K_j. The watermarked video should satisfy the quality specification Q_l. This is repeated for all elements of $\mathbf{V}$, $\mathbf{M}$, $\mathbf{K}$ and $\mathbf{Q}$. The output is a set $\mathbf{V}^w$ of watermarked videos. The attack module distorts the watermarked videos of $\mathbf{V}^w$ using all the attacks of the set $\mathbf{A}$ and a set $\mathbf{V}^a$ of attacked videos is generated. The decoding module performs the decoding and/or the extracting process into the attacked videos of $\mathbf{V}^a$. The output is an approximation to the embedded message. It is evaluated only if detection is positive. The performance evaluation module is used in order to obtain the performance scores of the watermarking scheme under test or its suitability for a certain application scenario (represented by set $\mathbf{S}$).

The proposed video benchmark system *VidMark* focuses on evaluating the performance of the watermarking schemes against temporal attacks. At this moment only a set of temporal attacks can be used to evaluate the performance of video watermarking schemes. In the future, other attacks can be added such as collusion, additive and multiplicative noise, among others. Its important to highlight that the aim of this paper is to describe the bechmark framework and the temporal attacks (defined in the attack module) performed by VidMark. The VidMark can be found in [8].

III. V

Video sequences consist of a series of consecutive and time spaced still images called video frames. Thus, any watermarking scheme for still images can be used to watermark every frame in the video. The watermarking scheme designer needs to consider some aspects about this medium to implement a specific video watermarking scheme. First, the size of a video is bigger than in an image, thus, generally a video always will be codified to reduce the storage size. This implies that by default the watermarking scheme needs to be robust to video compression (the main compression standards are: MPEG-1, MPEG-2, MPEG-4 part 2 and part 10). Second, the big amount of information that needs to be presented to the user in a specific time increments the restrictions to consider real time applications like video streaming. And finally, the more important aspect is the presence of temporal attacks, which are not present in images e. g. frame dropping, frame transposition, frame inserting, among others.

The main robustness requirements of video watermarking schemes [9] are: video compression, signal enhancement, additive and multiplicative noise, A/D and D/A conversions, video format conversions, averaging and collusion attacks, temporal attacks, among others.

Most of these robustness requirements are spatial attacks i. e. attacks that modify each video frame separately like still images. However, the attacks that change the order or amount of video frames like video compression, A/D and D/A conversions, video format conversions and temporal attacks imply temporal modifications. The temporal modifications must be considered when the message needs to be hidden in two o more video frames, which implies a synchronization phase to extract the message in the correct order. Temporal synchronization is the process of identifying the correspondence between the temporal coordinates of the watermarked signal and the ones for the watermark.

A general diagram of video watermarking is shown in figure 2. The embedding process hides a message M into the original video $\mathbf{X}$ using, if necessary, an embedding key K_e. The result of this process is a watermarked video $\mathbf{Y}$. This watermarked video $\mathbf{Y}$ can possibly be attacked, generating $\hat{\mathbf{Y}}$. The detection process extracts the message M' from the video $\hat{\mathbf{Y}}$ using, if necessary, the detection key K_d. The original video $\mathbf{X}$ is an ordered sequence

$$\mathbf{X} = \langle X(0), X(1), \ldots, X(t), \ldots, X(n-2), X(n-1) \rangle \quad (1)$$

$X(t)$ corresponds to the frame shown at time t and the cardinality of $\mathbf{X}$ is the total number of frames or $|\mathbf{X}| = n$. It is important to mention that, in this paper, $\langle . \rangle$ denotes an ordered sequence and $\{.\}$ denotes set membership. Thus, $\mathbf{X} = \langle X(0), X(1), X(2) \rangle$ is the ordered sequence of $X(0)$ followed by $X(1)$, followed by $X(2)$. $X = \{X(0), X(1), X(2)\}$ is a set with members $X(0), X(1)$ and $X(2)$. In this way, the watermarked video $\mathbf{Y}$ is a sequence

$$\mathbf{Y} = \langle Y(0), Y(1), \ldots, Y(t), \ldots, Y(n-2), Y(n-1) \rangle \quad (2)$$

and the possibly attacked video $\hat{\mathbf{Y}}$ is a sequece

$$\hat{\mathbf{Y}} = \langle \hat{Y}(0), \hat{Y}(1), \hat{Y}(2), \ldots, \hat{Y}(t), \ldots \rangle \quad (3)$$

in this case the number of frames of $\hat{\mathbf{Y}}$ can be different to n due to temporal attacks. If $\hat{\mathbf{Y}}$ is identical to $\mathbf{Y}$ i. e. $\hat{\mathbf{Y}}(t) = \mathbf{Y}(t) \forall t$ then no attacks were performed.

A. Temporal Attacks

Due to the lack of a benchmark suite to evaluate video watermarking schemes against temporal desynchronization attacks, a set of temporal attacks is proposed. It does not differentiate between intentional and unintentional processing.

Frame dropping: Video frames can be deleted or simply lost during the delivery of frames in several scenarios like video streaming or video compression. This implies that $|\hat{Y}| < |Y|$ and $\hat{Y} \subset Y$. Finally, all the frames in $\hat{\mathbf{Y}}$ are ordered ascending in time, thus every pair of attacked frames $\hat{Y}(t-1)$ and $\hat{Y}(t) \in \hat{\mathbf{Y}}$ which correspond to the

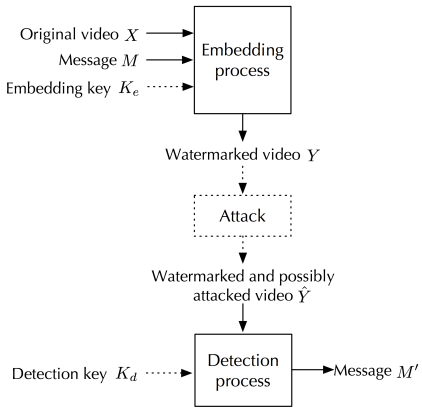


Fig. 2. General diagram of video watermarking.

watermarked frames $Y(t_a)$ and $Y(t_b) \in \mathbf{Y}$ respectively are ordered in time because $t-1 < t$ and $t_a < t_b$.

For example, let $\mathbf{Y} = \langle y_1, y_2, y_3, y_4, y_5 \rangle$ be a watermarked video, after a frame dropping attack the video generated is $\hat{\mathbf{Y}} = \langle y_2, y_3, y_5 \rangle$ (frames $Y(0) = y_1$ and $Y(3) = y_4$ were dropped). This implies that $|\hat{Y}| < |Y|$ and the time index of every pair of attacked frames in comparison with the time index of the same frames before the attack is ascending. Since the pairs of consecutive attacked frames are (y_2, y_3) and (y_3, y_5) and its time index before the attack are (1,2) and (2,4) respectively, thus the indexes operations $1 < 2$ and $2 < 4$ confirm the ascending order presented. Several video watermarking schemes that present results against frame dropping are [10]–[16], however the results they present are not standarized. In the proposed frame dropping attack, each video frame has the same probability of being dropped.

Frame inserting: This attack consists on inserting randomly unwatermarked frames (frames of the original or another video). After this attack, $|\hat{Y}| > |Y|$. Let X be the unwatermarked video that is utilized to insert its frames in the watermarked video, then the frame inserting attack satisfies the conditions: $Y \subset \hat{Y}$ and $\exists x \in X \wedge x \in \hat{Y}$.

The first condition ensures that the attacked video $\hat{Y}$ consists of all the frames of Y and the second one ensures that, additionally, some frames of $\hat{Y}$ are elements of X. An example of this attack is the next. Let $\mathbf{Y} = \langle y_1, y_2, y_3, y_4, y_5 \rangle$ be a watermarked video and $\mathbf{X} = \langle x_1, x_2, x_3, x_4 \rangle$ be an unwatermarked video. After frame inserting the attacked video would be $\hat{\mathbf{Y}} = \langle x_3, y_1, y_2, x_1, y_3, y_4, x_4, y_5 \rangle$. As it can be observed, the previous two conditions are satisfied. Some of the works that report this attack are [10]–[13], [15]. In the proposed frame inserting attack, the unwatermarked frame insertion position is randomly selected.

Frame transposing: This attack consists on interchanging two or more frames at the same time. There are two ways to perform this attack. 1)Simple transposition: A frame is selected randomly and a new position is given. This implies that more than one frame changes its position. 2)Simultaneous transposition: A frame is interchanged with any frame in the whole video. In this way, only two frames modify their positions.

In this attack $|\hat{Y}| = |Y|$ and $y \in \hat{Y} \wedge y \in Y$, but $\hat{\mathbf{Y}} \neq \mathbf{Y}$ because $\hat{\mathbf{Y}}$ and $\mathbf{Y}$ are not equally ordered.

For example, let $\mathbf{Y} = \langle y_1, y_2, y_3, y_4, y_5 \rangle$ be a watermarked video, after a simultaneous transposition attack the generated video is $\hat{\mathbf{Y}} = \langle y_1, y_5, y_3, y_4, y_2 \rangle$. In this example the frames interchanged were $Y(1) = y_2$ and $Y(4) = y_5$. Some of the works that report this attack are [13] and [16].

Frame decimation: The frame decimation occurs when x consecutive frames are dropped. Decimation with a factor of N_D retains the first frame out of every N_D consecutive frames. Another way of performing the frame decimation is by randomly selecting the frame to retain out of the N_D frames, thus the selected frame can be anyone. In this attack $|\hat{\mathbf{Y}}| = \lfloor |\mathbf{Y}|/N_D \rfloor$. A work that reports this attack is [13].

Frame averaging: This attack can be performed in two ways. 1)Sliding averaging: A window of fixed size is swept temporally across the video. At each step, a composite frame is constructed by computing the average pixel value over all the frames in the window. 2)Simple averaging: Like the simultaneous transposition, a frame is averaged with other frame in the whole video. This attack generates a video

$$\hat{\mathbf{Y}} = \langle \hat{Y}(0), \hat{Y}(1), \ldots, \hat{Y}(n-1) \rangle$$

where

$$\hat{Y}(t)_{ij} = \frac{1}{k} \sum_{x=0}^{k-1} \bar{Y}(x)_{ij}$$

$$\bar{\mathbf{Y}} = \langle \bar{Y}(0), \bar{Y}(1), \ldots, \bar{Y}(k-1) \rangle$$

here, $\bar{\mathbf{Y}}$ is the frame set selected to be averaged with $K > 0$, $0 \leqslant t \leqslant n-1$ and in all the cases $|\hat{Y}| = |Y|$. In [13] this attack is reported.

Frame rate conversion: In this work this attack is performed in two ways. 1)Duplication and dropping. The way to achieve the FPS (frame per second) required is by duplicating and deleting frames. 2)Frame blending: The only frame operation permitted is frame averaging to insert a frame or to substitute a frame. A work that reports this attack is [15].

Combined attack: Combination of previous attacks can be performed. This kind of attack generally decreases drastically the visual quality of the video.

MPEG compression: In most cases, it is impractical to store raw videos due to the huge information redundancy they present, so, video compression algorithms are used to reduce this redundancy. The MPEG compression supported by the proposed benchmark are: MPEG-2, MPEG-4 part 2 and MPEG-4 part 10. It is important

to mention that, although the MPEG compression is not considered a temporal attack, internally the MPEG compression performs temporal attacks like frame deleting to reach the bitrate required (the way to perform this depends on the implementation of the rate control unit). Moreover, in MPEG compression the most obstructive video attack appears, the motion prediction attack, that in some cases deletes most of the frame energy.

The attacks described above are considered the most significative temporal attacks in video watermarking schemes.

IV. P

In order to properly evaluate the performance of video watermarking schemes and to allow a fair comparison between different schemes, a benchmark suite must include a set of tests and a way of measuring the results of the tests using controlled conditions. In this way, the tests initial conditions are of high importance. Thus, attack parameters would be selected by analyzing the state of the art in video watermarking schemes that resist temporal attacks.

In watermarking, the tests are oriented to measure the requirements of an application. So, the robustness, the fidelity and the capacity are commonly measured.The term robustness describes the watermark resistance to these attacks and can be measured by the bit correct ratio (BCR), which is defined as the ratio of correctly extracted bits to the total number of embedded bits. The most important performance measure relates the watermark robustness to the attack and shows the overall behavior of the method towards attacks. Additionally, the false probability alarm and false rejection probability tests can be performed. Fidelity is the perceptual similarity between the original and the watermarked video. Fidelity is usually defined in terms of Signal to Noise Ratio (SNR) or Peak Signal to Noise Ratio. The capacity is the maximum amount of information that can be hidden in a medium, due to the robustness is the most important requirement used in watermarking, the capacity is usually not considered.

Unlike the image watermarking, the embedding and detection time is important to evaluate, since the video presents the time dimension.

V. C

A new framework of video watermarking benchmark specialized in temporal attacks, has been described in this paper. Due to the temporal attacks can cause loss of synchronization and this kind of attacks are exclusive for video, the main temporal attacks are detailed and different ways of performing them are described. As future work, the performance measures will be analyzed in depth in order to evaluate the fidelity and robustness, among others. In addition, the construction of a video data set can be performed. Finally, due the module approach of the benchmark framework more video attacks and performance measures can be added.

A

The authors acknowledge to CONACYT the support provided through the grant for PhD studies number 174444 and the project grant CB-2007-1-84668.

R

[1] M. Kutter and F. A. P. Petitcolas, "A fair benchmark for image watermarking systems," *Security and Watermarking of Multimedia Contents*, vol. 3657, pp. 226–239, Abril 1999.

[2] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Attacks on copyright marking systems," in *Information Hiding, Second International Workshop, IH'98*, ser. Lecture Notes in Computer Science, D. Aucsmith, Ed., vol. 1525. Portland, Oregon, U. S. A.: Springer, Abril 1998, pp. 218–238.

[3] F. A. P. Petitcolas, "Watermarking schemes evaluation," *IEEE signal processing*, vol. 17, no. 5, pp. 58–64, Septiembre 2000.

[4] S. Pereira, S. Voloshynovskiy, M. Madueño, S. Marchand-Maillet, and T. Pun, "Second generation benchmarking and application oriented evaluation," in *In Information Hiding Workshop III*, Pittsburgh, PA, USA, Abril 2001.

[5] V. Solachidis, A. Tefas, N. Nikolaidis, S. Tsekeridou, A. Nikolaidis, and I. Pitas, "A benchmarking protocol for watermarking methods," in *2001 IEEE International Conference on Image Processing*, vol. 3, Thessaloniki, Greece, Octubre 2001, pp. 1023–1026.

[6] H. C. Kim, H. Ogunleye, O. Guitart, and E. J. Delp, "The watermark evaluation testbed (wet)," in *Security, Steganography, and Watermarking of Multimedia Contents*, ser. Proceedings of SPIE, E. J. Delp and P. W. Wong, Eds., vol. 5306. SPIE, 2004, pp. 236–247.

[7] C. O. Dumitru, S. Duta, M. Mitrea, and F. Preteux, "Gaussian hypothesis for video watermarking attacks: Drawbacks and limitations," *The International Conference on "Computer as a Tool" EUROCON, 2007*, pp. 849–855, Septiembre 2007.

[8] [Online]. Available: http://ccc.inaoep.mx/~cferegrino/vidmark/

[9] EURECOM, UNIGE, TUD, AUTH, UVIGO, NIM, INA, EPFL, PHILIPS, TCC, IGD, and UCL, "Common data processing and intentional attacks," Certimark, Tech. Rep. D 4.1, Oct. 2001. [Online]. Available: http://www.certimark.org/public/public_deliverable.html

[10] L. Kezheng, Y. Wei, and F. Bo, "Digital watermarking synchronization algorithm of mpeg-4 video," *Computer and Computational Sciences, International Multi-Symposiums on*, vol. 0, pp. 164–167, 2008.

[11] C. Chen, T. gang Gao, and L. zong Li, "A compressed video watermarking scheme with temporal synchronization," *Congress on Image and Signal Processing*, vol. 5, pp. 605–612, 2008.

[12] L. Kezheng, Y. Wei, and L. Pie, "Video watermarking temporal synchronization on motion vector," *3rd International Conference on Intelligent System and Knowledge Engineering. ISKE 2008.*, vol. 1, pp. 1105–1110, Noviembre 2008.

[13] E. T. Lin and E. J. Delp, "Temporal synchronization in video watermarking," *IEEE Transactions on Signal Processing*, vol. 52, no. 10, pp. 3007–3022, Octubre 2004.

[14] L. he Zhang, G. Xu, J. jun Zhou, and L. jie Shen, "A video watermarking scheme resistant to synchronization attacks based on statistics and shot segmentation," *Seventh International Conference on Intelligent Systems Design and Applications. ISDA 2007*, pp. 603–608, Octubre 2007.

[15] L. Yang and Z. Guo, "A robust video watermarking scheme via temporal segmentation and middle frequency component adaptive modification," in *Digital Watermarking*, ser. Lecture Notes in Computer Science, vol. 4283, 2006, pp. 150–161.

[16] J. Sun and J. Liu, "A temporal desynchronization resilient video watermarking scheme based on independent component analysis," in *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 1, Sept. 2005, pp. I–265–8.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING or 371(c) DATE	GRP ART UNIT	FIL FEE REC'D	ATTY.DOCKET.NO	TOT CLAIMS	IND CLAIMS
61/399,945	07/19/2010		110	MX/Inaoc-10-03		

CONFIRMATION NO. 9290

Ekstrom Law Group
Suite 120-248
650 Castro St.
Mountain View, CA 94041

FILING RECEIPT

OC000000042907528

Date Mailed: 08/09/2010

Receipt is acknowledged of this provisional patent application. It will not be examined for patentability and will become abandoned not later than twelve months after its filing date. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please submit a written request for a Filing Receipt Correction. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections**

Applicant(s)

Pedro A. Hernandez-Avalos, Puebla, MEXICO;
Claudia Feregrino-Uribe, Puebla, MEXICO;
Rene Cumplido, Puebla, MEXICO;
Jose Juan Garcia-Hernandez, Puebla, MEXICO;

Power of Attorney:
Carmen Pili Ekstrom--34981

If Required, Foreign Filing License Granted: 08/04/2010
The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is ***US 61/399,945***
Projected Publication Date: None, application is not eligible for pre-grant publication
Non-Publication Request: No
Early Publication Request: No
**** SMALL ENTITY ****
Title

Video watermarking scheme resistant to temporal desynchronization attacks

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same

effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at http://www.uspto.gov/web/offices/pac/doc/general/index.html.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, http://www.stopfakes.gov. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4158).

LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with

respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign AssetsControl, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

VIDEO WATERMARKING METHOD RESISTANT TO TEMPORAL DESYNCHRONIZATION ATTACKS

TECHNICAL FIELD

The present invention relates to multimedia signal processing, temporal synchronization protocols for video and audio, and temporal synchronization in digital watermarking applications and more particularly, it is related to a video watermarking method resistant to temporal desynchronization attacks.

BACKGROUND OF THE INVENTION

The use of video watermarking is focused on protection issues where just a few bits are enough for copyright protection of content. Nevertheless, the use of schemes that embed more information opens the door for the use of the watermarking in emerging rich media applications like indexing, subtitles, hypervideo, interactive video, etc. as disclosed by Dumitru et al. at the EUROCON 2007 conference: “Gaussian Hypothesis for Video Watermarking Attacks: Drawbacks and Limitations".

Nowadays, the protection of videos is usually achieved by hiding a single watermark in a full video such as the scheme disclosed in the PhD thesis of L. Coria-Mendoza “Low-Complexity methods for Image and Video Watermarking”. The watermarking schemes for media protection embed little information to ensure a greater probability of detection. Increasing the capacity of information embedding involves the use of more complex watermarking schemes, mainly to ensure the correct extraction of the watermark in the right order. This implies introducing a temporal synchronization phase, e.g., temporal synchronization is the process of identifying the correspondence between the temporal coordinates of the watermarked signal and the ones for watermark. Furthermore, it is important to mention that the video will be compressed, which causes the watermarking schemes to be robust to video compression attacks.

As it is known, the video compression process removes temporal redundancy by using a motion compensation unit and a rate control unit. These two internal tasks are the main cause of temporal desynchronization on compressed videos. The temporal synchronization is crucial for successfully detecting watermarks. If the detector cannot be synchronized with its input video, an embedded watermark cannot be detected even though it is present in the video. The video compression schemes attack the watermark hidden in each frame and desynchronize in time the detector due to the mechanism used to reduce the redundancy in the video. Thus, if two or more frames hide a single watermark (regardless of the reasons), then a temporal synchronization process is necessary to resist the video compression attack.

In order to achieve temporal synchronization and resistance to video compression, some watermarking schemes, such as the one described by Zhang et al. at the ISDA 2007 conference: "A Video Watermarking Scheme Resistant To Synchronization Attacks Based On Statistics And Shot Segmentation", and Kezheng et al. at the ISKE 2008 conference: "Video Watermarking Temporal Synchronization On Motion Vector", hide information by changing slightly the magnitudes of the motion vectors of an entire scene. To perform this, motion vectors are classified into different groups, then a bit of information is embedded in each one. In this way, a bit is hidden within a group of vectors with similar characteristics. However, when using the internal task of MPEG compression as disclosed by Moving Pictures Expert Group ISO/IEC 14496-2, ISO/IEC 14496-10 and ISO 13818, (the generation of motion vectors), these schemes provide robustness to this attack. This could be seen as a disadvantage because if the video is re-encoded, the watermark can be lost.

C. Chen et al. at the 2008 Congress on Image and Signal Processing, in the document "A Compressed Video Watermarking Scheme With Temporal Synchronization", provides another scheme which hides information using other inner process carried out into the compression process like the use of the DCT coefficients (Discrete Cosine Transform), in the MPEG-4 compression. The spread spectrum is used to hide the same watermark in a subset of DCT blocks per video scene. This scheme depends on the internal task performed by the encoder.

The scheme described by Lin et al. in the documents: "Temporal synchronization in video watermarking" published in the Proceedings of the SPJE International Conference on Security and Watermarking of Multimedia Contents IV: "Temporal synchronization in video watermarking" published in IEEE Transactions on signal processing; and by Delp et al. in the document "Optical and Digital Techniques for Information Security" of 2005, hides information on the raw video, which makes it independent of the encoder. However, the scheme was not resistant to compression attacks.

The scheme above mentioned performs temporal synchronization by manipulating the embedding keys used for watermarking each frame. In this way, the watermark that is embedded in each frame carries along temporal information. The key is changed every β frames using the information of the current frame and the previous key, thereby a new key time dependent of the current and the previous frame is generated.

In US Patent No. 7,567,721 B2 a method for embedding digital watermarks in compressed video include perceptual adapting a digital watermark in predicted and non-predicted data based on block activity derived from the compressed video stream,

embedding in predicted objects in a video stream having separately compressed video objects and bit rate control of watermarked video.

One of the improvements made to the original Lin's scheme algorithm is attributed to a paper published by Delp and Lin in 2004, "Temporal Synchronization In Video Watermarking", the disclosure of which was incorporated in the US Patent No. 7,886,151 which describes a protocol for temporal synchronization of media signals with temporal components to be used in digital watermarking and other applications. This synchronization protocol achieves initial synchronization by finding an initial synchronization key through analysis of a temporal media signal stream. It then uses features of the stream and a queue of one or more keys from previous frames to derive subsequent keys to maintain synchronization. If synchronization is lost due to channel errors or attacks, for example, the protocol uses the initial synchronization key to re-establish synchronization. In digital watermarking applications, the synchronization protocol is agnostic to the watermark embedding and reading functions.

The US Patent No. 7,840,005 B2 relates to synchronization paradigm and applies this paradigm to different forms of synchronization, including both temporal and spatial synchronization of digital watermarks. For spatial synchronization, a paradigm in a spatial coordinate system of a media signal is applied, such as the two-dimensional spatial coordinate system of a digital still image or frame of video. The paradigm is applied to perform spatial synchronization of digital watermarks.

For performing the synchronization, the method of US Patent No. 7,840,005 B2 comprises: detecting peaks due to redundancy of features in a host media signal; wherein the redundancy is controlled via a state machine that repeats a portion of a watermark structure and varies another portion of the watermark structure over a coordinate system of the host media signal; analyzing the peaks to derive estimates of geometric or temporal distortion of the host media signal; computing a histogram of the estimated geometric or temporal distortion parameters; and from the histogram, computing a geometric or temporal distortion of the host media signal.

As we can see from the above, prior methods describe synchronizing a digital watermark by using feature extraction and a key generator. The temporal synchronization embeds a watermark within each frame that carries temporal information. The key is changed every β frames, which is called the local repeat. Keys are based on previous keys and information from the current frame. A finite state machine is used to calculate the keys. It is initiated by a global master key and can be reset every α frames (global repeat rate).

Since the present invention scheme is directed to a scheme comprising MPEG standards, it is quite convenient to define the type of frames included in it. MPEG

standards include three kinds of frames: a) intra picture frames (I-frames); b) forward-predicted frames (P-frames); and c) bidirectional-predicted frames (B-frames). A video stream or recording will always start with an I-frame and will typically contain regular I-frames throughout the stream. These regular I-frames are crucial for the random access of recorded MPEG-4 files, such as with rewind and seek operations during playback. As is mentioned by Smart et al. in IC-COD-REP012 of 2008 “Understanding Mpeg-4 Video”, the main disadvantage of I-frames is that they tend to compress much less than P-frames or B-frames.

I-frames are coded without reference to other frames. They are coded like an image. P-frame applies motion prediction by referencing an I-frame or P-frame in front of it, motion vector points to the block in the referenced frame. B-frame applies motion prediction, referencing a frame in front of it and/or a frame behind it. Each of the two referenced frames may be I-frame or P-frame.

Macro block (MB) in video stream is represented as a 16 x 16 sample area. Each MB contains six 8 x 8 blocks, four for luminance and two for chrominance. A block of an I-frame contains six 8 x 8 block S, four for luminance and two for chrominance. A block of an I-frame contains simply values of luminance or chrominance of its own. A block of a P-frame or B-frame contains the difference between the values of itself and the referenced block. This process is called motion compensation. Each frame is divided into MBs. The coding process of each block includes DCT (Discrete Cosine Transform), quantization, run length encoding and entropy coding in that order. The resulting video stream comprises of entropy codes, motion vectors and control information about the structure of video and characteristics of coding.

The present invention scheme provides a more robust video watermarking scheme resistant to various types of attacks. The present invention comprises a scheme which is robust to MPEG-2, MPEG-4 Part 2 and MPEG-4 part 10 compression attacks. Moreover, the scheme provides an improved resistance against the more common temporal synchronization attacks.

OBJECTS OF THE INVENTION

Having in mind the drawbacks of the prior art, it is an object of the present invention to provide a video watermarking method resistant to temporal desynchronization attacks, which can resist MPEG-2, MPEG-4 Part 2 and MPEG-4 part 10 compression attacks

One other object of the present invention is to provide a video watermarking method resistant to temporal desynchronization attacks, which is more robust against

various types of attacks, allowing the owners of digital media to track their copyrighted property more accurately

Another object of the present invention is to provide a video watermarking method resistant to temporal desynchronization attacks, which allows the keys to be less dependent on a single frame, making them more dependent on the averaged features of similar frames instead of the features of a single frame.

It is still another object of the present invention to provide a video watermarking method resistant to temporal desynchronization attacks, which provides a second global period, creating redundancy of keys for each scene shift and making the algorithm more robust against MPEG compression.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features that are considered characteristic of the present invention are set forth with particularity in the appended claims. The invention itself, however, together with other objects and advantages thereof, will be best understood in the following detailed description of certain embodiments, when read in connection with the accompanying drawings, wherein:

FIG. 1 is a schematic diagram of the prior art finite state machine used to generate new keys at Lin et al. scheme.

FIG. 2 is a schematic diagram of an example of the key sequence of Lin et al. scheme shown in **FIG. 1** where a repeating key seed is applied to the FSM every eight frames (α=8) while every two consecutive frames (β=2) the local key is repeated.

FIG. 3 is a schematic diagram of an example of the keys generated using a repetition β adaptive of the present invention scheme.

FIG. 4 is a schematic diagram of an example of the keys generated using a characteristics extractor that use a set of β frame of the present invention scheme**.**

FIG. 5 is a schematic diagram of the key generation redundancy per scene using the present invention scheme**.**

FIG. 6 is a chart of the performance of the present invention scheme against MPEG-2, MPEG-4 part 2 and MPEG-4 part 10 compression.

DESCRIPTION OF PREFERRED EMBODIMENTS

The present invention is directed to provide a video watermarking scheme resistant to temporal desynchronization attacks, increasing robustness in the scheme to temporal desynchronization attacks; however, for a proper understanding of the video watermarking scheme of the present invention, it is important making reference to the

scheme proposed by Lin et al., which is discussed in the background of the invention chapter.

As it was mentioned above, the Lin et al. scheme, performs temporal synchronization by manipulating the embedding keys used for watermarking each frame. So the watermark embedded in each frame carries along temporal information. The key is changed every β frames using the information of the current frame and the previous key, thereby a new key time-dependent of the current and the previous frame is generated.

Regarding **FIG. 1**, key generation in Lin's scheme is achieved by a finite state machine (also known as FSM) following the next steps:

1. The finite state machine is restarted (state 0) and the current embedding key K_i is set to K_e, which is the master key, then the embedding process generates a watermarked frame Y(i).
2. The characteristics X, Y and Z are obtained from Y(i) and they are used to define the next state in the FSM.
3. The next embedding key K_{i+1} (used to watermark the frame X(i + 1)) is computed using the state equations shown in **FIG. 1**.
4. The next frame is obtained and the embedding key K_{i+1} is used to watermark it.

The steps 2, 3, 4 are carried out, using this new frame, until all frames are generated and watermarked.

There is a mechanism for generating a period redundancy α that indicates the maximum number of frames that will be watermarked until the use of K_e, and a repeat redundancy β that indicates the amount of frames watermarked using the same key. In other words, a master key is seeded every α frames. The next key is then calculated from the previous key and feature information from the current frame.

In **FIG. 2** it is shown an example of a key sequence with α = 8 and β = 2. The same tone of gray indicates that the same key was used to hide the watermark. It can be noticed that every two frames the key is changed and every eight frames the key is restarted. Moreover, the keys used in the frames 2 and 3 are not the same as those used in the frames 10 and 11. During a period α, the only key that remains unchanged is the master key K_e. The frames watermarked with the master key are called resynchronization frames, which are used as a reference when the detector loses synchronization. The period given by α controls the degree of overall redundancy and determines how many frames the key generator has restarted.

The repetition β controls the degree of local synchronization. A high degree of repetition β increases the redundancy and decrements the generation of new keys, which is an advantage to resist temporal attacks such as frame dropping, frame transposing and frame averaging. Moreover, if the degree of repetition is small, there is little or no

redundancy, which means that when a single watermark is not detected the synchronization period is completely lost.

In the watermark detection process, the main task is carried out by the finite state machine and by a queue that keeps track of frames not synchronized. A null queue is generated first, and then the key K_e is used for continuously extracting the watermark from the frames.

Once this watermark is correctly detected, the predicted state-keys (using the same FSM as in the embedding as in **FIG. 1**) is pushed into the queue. Then, watermark extraction of each frame is repeated with the keys K_e, and those reserved in the queue. When a watermark is detected, the current state-key will be moved to the front of queue if it does not belong to the initial set. Then the predicted state-keys of current state-key is pushed into the queue. The key generation in the embedding and in the extracting processes depends on the use of the FSM. When the magnitudes of X, Y and Z (from a frame) are close during the embedding process and Y(i) is attacked, the FSM transition (in the extracting process) can be different from the expected one, causing the temporal desynchronization.

Now regarding the present invention, at least three improvements over the prior art schemes and more precisely, over the Lin et al. scheme were made. The new scheme comprises: a) To perform an adaptable β repetition; b) To modify the characteristic extractor in order to consider β frames; and c) To modify the temporal redundancy in order to generate redundancy in I-frames.

In the prior art schemes, a fixed β repetition is used. However, this can be counterproductive due to MPEG-2, MPEG-4 part 2 and MPEG-4 part 10 compressions strongly attack temporal redundancy through motion compensation. If a frame sequence belongs to a scene with little movement, this compensation destroys part of these frames. If frame destruction happens and the magnitude of β is small, the watermark hidden in that repetition will be lost, which causes temporal desynchronization (the following keys will be lost because they are related to the previous keys).

One of the improvements of the present invention over the prior art is to hide the same watermark in a number of frames given by the mutual information between them. So, if in similar frames, the mutual information is high and in disimilar frames, the mutual information is low, then a measure of similarity in time can be utilized. The metric of mutual information is given by equation I:.

$$I(X,Y) = \sum_x \sum_y p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)} \right) \qquad 1)$$

where p(x, y) is the probability mass function or *pmf* of the random variables X and Y (i. e., frame X and frame Y), p(x) and p(y) represents the marginal *pmf* of X and Y, respectively.

To determine the amount of frames watermarked with the same key, a threshold U is used, in this case U = 1. Furthermore, a limit in the number of consecutive frames to be watermarked is established to 20. An example of this embodiment is shown in **FIG. 3** where the values of β are 4, 2, 4, 3, 2 and 2. These values indicate that the first four frames are similar, the next two frames are similar too, the next four frames are similar again and so on. The dissimilarity between the consecutive frames determines the end of each β frames.

Another improvement made by the present invention is directed to the characteristics of the extractor. The extractor in the prior art obtains the characteristics from the watermarked frame (the average of frame set is being used so far) and uses them to decide the route through FSM transitions. It is important to notice that the extractor invokes the last watermarked frame of each β repetition, which generates a dependence between the frame and the generated key. In this manner, if the last frame of repetition is eliminated during detection, the characteristics of another frame will be taken into account to generate the new key; which causes that the new key is not the expected one.

The present invention comprises taking all the frames watermarked with the same key to extract their characteristics. An image that represents all the frames with β repetition is obtained by averaging the low frequencies of the watermarked frames using a Gaussian filter with a $\sigma = 4$. It should be understood that other averaging schemes could also be applied.

By adding the magnitudes of the feature points (using the Harris Detector as described in the Fourth Alvey Vision Conference by Harris et al. "A Combined Comer And Edge Detection”), to the X, Y and Z characteristics another improvement is made wherein the new X, y and Z characteristic are no longer close and prevents the erroneous transition in the FSM as it is shown in **FIG. 4**, In this context, the proposed extractor is not dependent of a previous frame so the present invention averaged the characteristics from all the frames watermarked with the same key allowing a greater robustness against frame dropping.

In relation to **FIG. 5,** another improvement of the invention over the prior art is shown, which comprises modifications to the temporal redundancy control to hide a watermark in each possible I-frame. The present invention provides MPEG compression which encodes some frames internally as if they were images (I-frame), some others using motion estimation taking I-frames as basis (P-frame) and finally some others using motion estimation taking P-frames as basis (B-frame). In this manner, it is pretend to hide

the watermark on each possible frame to avoid the aggressive attack of motion estimation. Also, I-frames are generally created by the codificator when a shift of scene is detected. In this sense, the aim is to generate redundancy of keys utilizing the shift of scene (the end of each scene is obtained using the metric of mutual information from equation 1). In this manner, there is redundancy of scene γ. The idea behind this new redundancy is to guarantee detection of at least one bit for each I-frame. Thus, the scheme of the present invention is able to support MPEG-2, MPEG-4 part 2 and MPEG-4 part 10 compressions.

This new redundancy γ generates a new chain of watermarked frames, i. e. only the frames in the beginning of a scene are watermarked using the key generated in the previous scene change (or by using the master key Ke). This improvement causes that the message will be embedded after the scene changes. When the whole message was embedded, in the new beginning of a scene the message will be embedded again. This is a new period redundancy.

This second period redundancy γ is introduced that occurs with every scene change in the video. With each scene change, the key for the first frame in the scene is generated from the initial frames in the previous scene. This minimizes the degradation from motion estimation by avoiding the use of B-frame and P-frames, which are the ones affected by motion estimation.

The period redundancy α (unknown at the beginning of the embedding process) depends on the amount of information to be embedded and on the size of β (adaptable). For example, if five bits are embedded and the values of β are: 3, 4, 3, 5 and 4, then α = 19.

Another embodiment of the present invention is directed to a scheme which uses two α values, one for embedding the information in a sequential way and the other for embedding the information at the beginning of each scene. The key generation on scheme of the present invention such as embedding and extracting processes, is well known to one of ordinary skilled in the art.

The present invention also is related to modules in the key generation process comprising: a) temporal redundancy control; b) feature extraction; c) a scene detector and d) β repetition detector. This peculiarity of the invention provides the object of making the watermarking scheme more robust against temporal attacks and video compression. Resistance against frame dropping, frame transposition, frame decimation and frame average are significantly improved with the modified algorithm.

As we can see from the above, the present scheme comprises an algorithm which makes the schematic method resistant to video compression (MPEG-2, MPEG-4 part 2, MPEG-4 part 10), change of frame rate and two other ways of implementing frame average and frame transposition. These types of attacks occur inadvertently or deliberately during digital transmission, and making the watermark more robust against these attacks would allow the digital media to be more robust through more transfers.

The prior art used a feature extractor and key generator to embed a time-dependent key into each frame. A finite state machine is employed to generate each key based on the previous key and frame features. Redundancy is provided across β consecutive frames (at a previously determined value) and also the FSM is re-seeded every α frames.

EXPERIMENTAL RESULTS

In order to evaluate the resistance against compression attacks, the performance of the method of the present invention is compared to the prior art, for such purpose, seven videos of 352 x 288 pixels with a range of 870 to 2101 frames at 25 frames per second (fps) were used. The conducted experiments comprises testing the videos with different bit rates and detecting the hidden watermark, measuring the bit correct ratio or BCR. The parameters used for Lin et al.'s scheme were α= 150 and β= 10, i.e., 15 bits were hidden and the size of the queue was 10 entries.

FIG. 6 shows the performance of both schemes: prior art (Lin et al.) scheme and the present invention against compression attacks. The performance of the present invention overcomes the prior art scheme in all aspects except for the case of encrypted video al 400 Kbps bit rate where both performed similarly. Both schemes are affected by MPEG-4 part 10 compression, since this achieves a better reduction of temporal and spatial redundancy mechanism (bit rate control).

However, the detection of the watermark using the present invention is up to 40% better than the scheme of the prior art against the MPEG-4 part 10 compression at 200 Kbps. The bit rate influences the impact of temporal and spatial attacks during compression. The higher the bit rate, the less aggressive the spatial and temporal attacks, and the lower the bit rate, the less severe the spatial and temporal attacks. The bit rate also affects the watermark detection as shown in **FIG. 6**, where the detection percentage improves as the bit rate increases.

The scheme of the present invention and the prior art's scheme utilize a basic spread spectrum technique to embed the watermark as indicated by equation 2.

$$y_i = x_i + sw_i \qquad 2)$$

where *y*, is the watermarked i frame, x_i is the original i frame, s is a robustness factor and w_i is a watermark. The watermark is a random variable with normal distribution N(0,1). Detection is performed by using a low pass filter to reduce the effect of the original image, as suggested by Kalker et al in SPIE Security and Watermarking of Multimedia Content: "A Video Watermarking System For Broadcast Monitoring", followed by correlation with the watermark signal. The correlation is carried out as shown in equation 3:

$$c = E\{y \bullet w\} \qquad 3)$$

if c ≈ 0, the watermark does exist

else c ≈ 0, the watermark does not exist

The present invention is directed to state transformation scheme of key sequence for video watermarking robust to MPEG-2, MPEG-4 part 2 and MPEG-4 part 10. The present scheme comprises redundancy mechanism in embedding process which increases the resistance to the main widely used video compression schemes. The experimental results indicate that the scheme of the present invention has unexpectedly better resistance to MPEG compression as compared to scheme known in the art.

In relation to an analysis of the influence of the watermark embedding in the bit rate control mechanism, as well as, embedding multiple watermarks in a frame to increase the capacity scheme, the algorithm of the present invention provides increased robustness against other types of attacks. Testing shows improved response to frame dropping, frame transposition, frame decimating, and frame averaging. A summary table is shown in Table 1 which compares BCR performance of the improved scheme against the prior art scheme using various types of attacks.

While the invention has been illustrated and described as embodied in the temporal synchronization in video marking scheme, it is not intended to be limited to the details shown, since it is well understood that various omissions, modifications, substitutions and changes in the forms and details of the scheme illustrated and in its operation can be made by those skilled in the art without departing in any way from the spirit of the present invention.

CLAIMS

1. A video watermarking method resistant to temporal desynchronization attacks, of the type that comprises a watermark embedding stage where the watermark is constructed and then embedded into an original signal to produce a watermarked signal using a finite state machine; an attack stage where the watermarked signal is subjected of different types of attacks; a watermark detection stage where a watermark extractor using a finite state machine examines and reports the watermarked signal and the initial synchronization is achieved by finding an initial synchronization key though analysis of the temporal media signal stream; characterized in that during the watermarking embedding stage, the same watermark is hidden in a number of frames given by the mutual information between them and a temporal redundancy control is modified to hide a watermark in each possible I-frame; and, during the watermark detection stage, the watermark extractor takes all the frames watermarked with the same key to extract its characteristics

2. A video watermarking method resistant to temporal desynchronization attacks according to claim 1, further characterized in that for hiding the same watermark in a number of frames given, the metric of mutual information is given by equation I:.

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)} \right) \qquad \text{I}$$

where p(x, y) is the joint probability mass function or *pmf* of the random variables X and Y (i. e., frame X and frame Y), p(x) and p(y) represents the marginal *pmf* of X and Y, respectively. Mutual information measures the information that frames *X* and *Y* share. This information is used like a simmilarity measure.

3. A video watermarking method resistant to temporal desynchronization attacks according to claim 1, further characterized in that for determining an amount of frames watermarked with the same key a threshold U is used.

4. A video watermarking method resistant to temporal desynchronization attacks according to claim 3, further characterized in that the amount of consecutive frames to be watermarked is established up to 20.

5. A video watermarking method resistant to temporal desynchronization attacks according to claim 1, further characterized in that an image that represents all the frames with "β" repetition is obtained by averaging the low frequencies of the watermarked frames using a Gaussian filter.

6. A video watermarking method resistant to temporal desynchronization attacks according to claim 5, further characterized in that the Gaussian filter used has a "σ" from 3 to 6.

7. A video watermarking method resistant to temporal desynchronization attacks according to claim 6, further characterized in that the Gaussian filter used has an "σ" of 4.

8. A video watermarking method resistant to temporal desynchronization attacks according to claim 5, further characterized by a new X, Y and Z characteristic no longer close to a previous frame, which are obtained adding the magnitudes of the feature points, using a Harris Detector, to an initial X, Y and Z characteristics.

9. A video watermarking method resistant to temporal desynchronization attacks according to claim 1, further characterized in that it provides MPEG compression which encodes some frames internally as if they were images (I-frame), in order to hide the watermark on each possible frame to avoid the aggressive attack of motion estimation.

10. A video watermarking method resistant to temporal desynchronization attacks according to claim 9, further characterized in that the MPEG compression which encodes some frames internally uses motion estimation taking I-frames as basis (P-frame), in order to hide the watermark on each possible frame to avoid the aggressive attack of motion estimation.

11. A video watermarking method resistant to temporal desynchronization attacks according to claim 9, further characterized in that the MPEG compression which encodes some frames internally uses motion estimation taking P-frames as basis (B-frame) in order to hide the watermark on each possible frame to avoid the aggressive attack of motion estimation.

12 A video watermarking method resistant to temporal desynchronization attacks according to claim 9, further characterized in that I-frames are created in the watermark embedding stage when a shift of scene is detected and the end of each scene is obtained using the metric of mutual information from equation I.

13. A video watermarking method resistant to temporal desynchronization attacks according to claim 1, further characterized in that a redundancy period α depends on the amount of information to be embedded and on the size of β, which is adaptable.

14. A video watermarking method resistant to temporal desynchronization attacks according to claim 13, further characterized in that two values of period redundancy (α) are used, one for embedding the information in a sequential way and the other for embedding the information at the beginning of each scene.

Índice de figuras

1.1. Diagrama general de un esquema de marcas de agua 5
1.2. Ejemplo de ataque espacial 15
1.3. Ejemplo de ataque temporal 15

2.1. Redundancia generada por el esquema de Lin 25
2.2. Patrón 3D utilizado para sincronización 26

3.1. Diagrama general del VidMark 35
3.2. Diagrama general, marcas de agua para video 37
3.3. Ataque temporal de eliminación de cuadros 39
3.4. Ataque temporal de inserción de cuadros 40
3.5. Ataque temporal de intercambio simple de cuadros 41
3.6. Ataque temporal de decimación de cuadros 42
3.7. Ataque temporal de promedio por ventana 43
3.8. Ataque temporal de promedio simple 44
3.9. Ataque temporal de cambio de framerate por duplicación y eliminación . . 45
3.10. Ataque temporal de cambio de framerate por fusión 45
3.11. Diagrama de cuadros MPEG 47

4.1. Diagrama general, marcas de agua para video 52
4.2. Proceso de Incrustación 55
4.3. Redundancia de repetición adaptable 57
4.4. Generación de llaves del esquema propuesto 58
4.5. Coeficientes DCT de una imagen 59
4.6. Patrón en zig-zag de coeficientes DCT 60
4.7. Clasificación de bloques en la extracción de características 65
4.8. FSM usado como generador de llaves 67
4.9. Proceso de decodificación de la marca 68

5.1. Tamaño promedio de las repeticiones β contra U 84
5.2. Características dominantes de las repeticiones β 85
5.3. Características obtenidas al atacar las repeticiones β 86
5.4. PSNR del esquema propuesto 89
5.5. Número de las repeticiones β 99

5.6. Numero de bits por cuadro 100
5.7. Número de cambio de escenas 101
5.8. Capacidad del esquema 102
5.9. Duración promedio de escenas 103

Índice de tablas

2.1. Comparación de los principales esquemas de marcas de agua que contemplan la sincronización temporal 24

3.1. Características de los ataques temporales reportados en los principales esquemas de marcas de agua 34

4.1. Tabla comparativa entre el esquema propuesto y el esquema de Lin 78

5.1. Rendimiento del esquema propuesto ante los ataques del VidMark 92

5.2. Comparación del esquema propuesto contra el esquema de Lin ante los ataques del VidMark 95

Glosario

Ataques temporales Son distorsiones hechas en un video, las cuales cambian el orden o la cantidad de cuadros provocando modificaciones temporales en el video. 15, 16, 38

Benchmark Es una herramienta para evaluar sistemas, ésta obtiene índices de rendimiento a través de la aplicación de pruebas estandarizadas. Estos índices son usados como medida de comparación entre sistemas. 17

CIF Formato intermedio común, es formato de video plano que se caracteriza por la resolución de 352×288 píxeles. 36

Close caption Es el nombre en inglés del sistema de subtítulos para programas de televisión y video destinado a permitir que las personas sordas o con dificultades para captar la señal de audio, puedan comprender lo que se dice en televisión o en los videos. 7, 8

Colusión La colusión o confabulación es un ataque en marcas de agua cuyo objetivo es obtener o eliminar la marca de un medio marcado. Este tipo de ataque consiste en que dos usuarios con medios marcados confabulen para analizar sus medios con el fin de obtener o eliminar la marca de sus medios. 20, 74

Copyright Derechos de autor. 2

F_β El vector de características F_β está formado por tres números $F1$, $F2$ y $F3$ los cuales representan características de β cuadros marcados. Este vector es usado por el generador de llaves para crear una nueva llave. 64, 66

$\hat{F_{\beta_i}}$ El vector de características $\hat{F_{\beta_i}}$ está formado por tres números $F1$, $F2$ y $F3$ los cuales representan características de los β cuadros anteriores marcados con la misma marca. Este vector es extraído durante el proceso de detección. Este vector es usado por el predictor de estados para crear un nuevo estado y una nueva llave. i es el índice de la llave detectada, iniciando con $i = 1$ cuando la llave K_E es detectada. 69

FFmpeg Es una colección de software libre que puede grabar, convertir y hacer streaming de audio y vídeo. 47

Fingerprinting Técnica usada para identificar a cada copia de un mismo medio, esto es análogo a una huella digital humana que identifica a una persona. 8

Framerate razón de cuadros por segundo (fps). 21

Hashing Función o método para generar claves o llaves que representen de manera casi unívoca a un documento, registro, archivo, etc. 26

Hipervideo Es una secuencia de video que contiene enlaces incrustados, los cuales permiten la navegación entre el mismo video y otros elementos multimedia. 4, 9

JPEG Grupo Conjunto de Expertos en Fotografía, es el nombre de un comité de expertos que ha creado estándares de compresión y codificación de archivos de imágenes fijas. El estándar más utilizado es JPEG. 59

MPEG Grupo de Expertos en Imágenes Móviles, es el nombre de un comité de expertos que ha creado estándares de compresión y codificación de videos. Los estándares más conocidos son MPEG, MPEG-2 y MPEG-4. 12

$\mathbf{K}$ Es una secuencia de llaves usada para marcar al video $\mathbf{Y}$. $K(t)$ corresponde a la llave usada para generar la marca $W(t)$ que está incrustada en el cuadro $Y(t)$. 52

K_E Es la llave de incrustación que será usada para genera una marca, la cual será incrustada en un cuadro específico. 37

$\hat{\mathbf{K}}$ Es una secuencia esperada de llaves necesarias para detectar las marcas incrustadas en un video $\hat{\mathbf{Y}}$. $\hat{K}(t)$ corresponde a la llave necesaria para detectar la marca en el cuadro $\hat{Y}(t)$. 52

λ Describe la función salida de la FSM, esta función realiza un mapeo de cada estado a un valor de salida. Después de una transición de estado, se obtiene una llave $K(t+1)$ mediante $K(t+1) = \lambda(s(t+1))$, la función $\lambda(\cdot)$ es una función uno a uno. 65

NTSC Es un sistema de codificación y transmisión de televisión en color analógico desarrollado en Estados Unidos, y que se emplea en la actualidad en la mayor parte de América y Japón, entre otros países. 15

PAL Es un sistema de codificación utilizado en la transmisión de señales de televisión analógica en color en la mayor parte del mundo. Se utiliza en la mayoría de los países africanos, asiáticos y europeos, además de Australia y algunos países americanos. 15

ϕ Describe la función transición de estados de la FSM, un ejemplo de esta función de transición es $s(t+1) = \phi(s(t), K_E, F_\beta)$ aquí las entradas a esta función son el estado actual $s(t)$, la llave de incrustación K_E y el vector de características F_β, la salida es el nuevo estado $s(t+1)$. 65

Piratería Acto de copiar o distribuir medios sin consentimiento de su propietario. 1

QCIF Formato intermedio común aun cuarto, es formato de video plano que se caracteriza por la resolución de 176×144 píxeles. Corresponde a una cuarta parte de la resolución CIF. 36

Redundancia de escenas La redundancia de escenas γ, también llamada redundancia de inicio de escenas, es la redundancia que se crea al incrustar el mensaje en el inicio de cada escena. 54

Redundancia de periodo La redundancia de periodo α, es la redundancia que se crea al incrustar el mismo mensaje de forma consecutiva a lo largo del video. 54

Redundancia de repetición La redundancia de repetición β, es la redundancia que se crea al incrustar la misma marca en cuadros consecutivos, lo que corresponde a una repetición β. 54

Repetición β Una repetición β es un conjunto de cuadros consecutivos marcados con la misma marca. 54

Sincronización Es el proceso de identificar la correspondencia entre las coordenadas de la señal marcada y las coordenadas de la marca incrustada. De esta forma, la sincronización temporal consiste en encontrar en dónde se ocultó la marca a través del tiempo. 19

Streaming Técnicas usadas para distribuir video por Internet. 14

$\mathbf{X}$ Es un video original, es decir que no ha sido marcado. $X(t)$ corresponde a al cuadro de $\mathbf{X}$ mostrado en el tiempo t. 37

$\mathbf{Y}$ Es un video marcado sin distorsiones. $Y(t)$ corresponde a al cuadro marcado de $\mathbf{Y}$ mostrado en el tiempo t. 37

$\hat{\mathbf{Y}}$ Es un video marcado posiblemente distorsionado. $\hat{Y}(t)$ corresponde a un cuadro de $\hat{\mathbf{Y}}$ mostrado en el tiempo t. 37

YUV 4:2:0 YUV es un método que define a una señal de video que separa los componentes de luminancia (Y) y crominancia (UV). 4:2:0 es el método específico para codificar dichos componentes. 36

Siglas

2DPCA Análisis en dos Dimensiones de los Componentes Principales. 23

AACS Sistema Avanzado de Acceso a Contenido. 2

BCR Razón de bits Correctamente Detectados. 36

CSS Sistema de Codificación de Contenido. 2

DCT Transformada Discreta de Coseno. 14, 26
DRM Administración de Derechos Digitales. 1, 2
DT CWT Transformada Wavelet Compleja de Árbol Dual. 15

EC Extractor de Características. 63

FPS Cuadros por segundo. 21
FSM Máquina de Estados Finitos. 65

HD-DVD DVD de alta definición. 2

ISO/IEC Organización Internacional para la Estandarización/Comisión Electrotécnica Internacional. 12

MPAA Asociación Cinematográfica de Estados Unidos. 3
MSE Error Cuadrático Medio. 87

PDM Proceso de Decodificación de la Marca. 67
PIM Proceso de Incrustación de la Marca. 53
PSNR Relación Señal a Ruido Pico. 36

SNR Relación Señal a Ruido. 36

VRML Lenguaje de Modelado de Realidad Virtual. 13

Referencias

[Anó98] Anónimo, "THE DIGITAL MILLENNIUM COPYRIGHT ACT OF 1998," *U.S. Copyright Office Summary*, Octubre 1998, http://www.copyright.gov/legislation/dmca.pdf. [Citada en p. 1]

[Anó06] Anónimo, "Advanced Access Content System (AACS): Introduction and Common Cryptographic Elements," Informe técnico, Intel Corporation, International Business Machines Corporation, Matsushita Electric Industrial Co. Ltd, Microsoft Corporation, Sony Corporation, Toshiba Corporation, The Walt Disney Company, Warner Bros, Febrero 2006, http://www.aacsla.com/specifications/. [Citada en p. 2]

[BB04] Mauro Barni y Franco Bartolini, *Watermarking Systems Engineering, Enabling Digital Assets Security and Other Applications*, Signal Processing and Communication Series, Marcel Dekker, New York, NY, USA, 2004, ISBN 0824748069. [Citada en p. 87]

[BCK+99] Jeffrey A. Bloom, Ingemar J. Cox, Ton Kalker, Jean-Paul M. G. Linnartz, Matthew L. Miller y C. Brendan S. Traw, "Copy Protection for DVD Video," en "Proceedings of The IEEE," tomo 87, Julio 1999. [Citada en p. 2]

[BCM02] Patrick Bas, Jean-Marc Chassery y Benoit Macq, "Geometrically invariant watermarking using feature points," en "IEEE Transactions on Image Processing," tomo 11, páginas 10–14, 2002. [Citada en p. 22]

[BKS01] Nikolaos V. Boulgouris, Filippos D. Koravos y Michael G. Strintzis, "Self-synchronizing watermark detection for MPEG-4 objects," en "The 8th IEEE International Conference on Electronics, Circuits and Systems. ICECS.", tomo 3, páginas 1371–1374, Octubre 2001. [Citada en p. 22]

[CgGzL08] Chao Chen, Tie gang Gao y Li zong Li, "A Compressed Video Watermarking Scheme with Temporal Synchronization," *Congress on Image and Signal Processing*, páginas 605–612, 2008, doi:http:

//doi.ieeecomputersociety.org/10.1109/CISP.2008.194. [Citada en p. 14, 22, 24, 27, 34]

[CHO+98] Tae-Yun Chung, Min-Suk Hong, Young-Nam Oh, Dong-Ho Shin y Sang-Hui Park, "Digital watermarking for copyright protection of MPEG2 compressed video," *IEEE Transactions on Consumer Electronics*, tomo 44(3):páginas 895–901, 1998. [Citada en p. 14]

[CKLS97] Ingemar J. Cox, Joe Kilian, F. Thompson Leighton y Talal Shamoon, "Secure spread spectrum watermarking for multimedia," en "IEEE Transactions on Image Processing," tomo 6, páginas 1673–1687, 1997. [Citada en p. 6, 74]

[CM08] Lino Evgueni Coria-Mendoza, *Low-Complexity methods for image and video watermarking*, Tesis Doctoral, The University of British Columbia, Vancouver, Canada, Marzo 2008. [Citada en p. 13, 15, 28]

[CMB+08] Ingemar J. Cox, Matthew L. Miller, Jeffrey A. Bloom, Jessica Fridrich y Ton Kalker, *Digital Watermarking and Steganography*, Morgan Kaufmann, San Francisco, California, second edición, 2008. [Citada en p. 5, 7, 61]

[CT91] Thomas M. Cover y Joy A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991. [Citada en p. 63]

[DCRP99] Fréderic Deguillaume, Gabriela Csurka, Joseph J. K. Ó Ruanaidh y Thierry Pun, "Robust 3D DFT video watermarking," en Ping Wah Wong y Edward J. Delp, editores, "Security and Watermarking of Multimedia Contents," tomo 3657, páginas 113–124, The Society for Imaging Science and Technology (IS&T) and the International Society for Optical Engineering (SPIE), SPIE, San Jose, California, U.S.A., 25–27 enero 1999, ISBN 0-8194-3128-1, ISSN 0277-786X. [Citada en p. 22]

[DD03] Gwenaël Doer y Jean-Luc Dugelay, "A guide tour of video watermarking," *Signal Processing: Image Communication*, tomo 18(4):páginas 263–282, Abril 2003. [Citada en p. 15]

[DDMP07] Corneliu O. Dumitru, Sorin Duta, Mihai Mitrea y Francoise Preteux, "Gaussian Hypothesis for Video Watermarking Attacks: Drawbacks and Limitations," *The International Conference on Çomputer as a Tool.EUROCON, 2007*, páginas 849–855, Septiembre 2007, doi:10.1109/EURCON.2007.4400522. [Citada en p. 4, 9]

[DHZQ08] Jing Dong, Hua Huang, Quan Zhou y Chun Qi, "Modified key sequence-based video watermarking scheme resistant to temporal

synchronization attacks," *International Conference on Audio, Language and Image Processing. ICALIP 2008*, páginas 1698–1702, Julio 2008. [Citada en p. 13, 23, 24, 25, 34]

[DL05] Edward J. Delp y Eugene T. Lin, *Optical and Digital Techniques for Information Security*, tomo 1, capítulo Watermarking Streaming Video: The Temporal Synchronization Problem, páginas 135–153, Springer New York, 2005. [Citada en p. 23, 24, 34, 75, 81, 82]

[DTSP08] Lambros Drossos, Dimitrios Tsolis, Spyros Sioutas y Theodore Papatheodorou, *Digital Rights Management for E-Commerce Systems*, Idea Group Inc, 2008. [Citada en p. 1]

[EUT+01] EURECOM, UNIGE, TUD, AUTH, UVIGO, NIM, INA, EPFL, PHILIPS, TCC, IGD y UCL, "Common data processing and intentional attacks," Informe Técnico D 4.1, Certimark, Oct. 2001. [Citada en p. 20]

[GH08] José Juan García-Hernández, *Sistema de protección de derechos de autor en señales de audio*, Tesis Doctoral, Intituto Politecnico Nacional, México, 2008. [Citada en p. 7]

[GLM05] Fabrizio Guerrini, Riccardo Leonardi y Pierangelo Migliorati, "A new video authentication template based on bubble random sampling," en "13th European Signal Processing Conference," Antalya, Turquia, Sep. 2005. [Citada en p. 22, 24, 25, 34]

[HAFUCGH09] Pedro A. Hernandez-Avalos, Claudia Feregrino-Uribe, Rene Cumplido y Jose Juan Garcia-Hernandez, "Video watermarking scheme resistant to MPEG compression," *Circuits and Systems, Midwest Symposium on*, tomo 0:páginas 853–858, 2009. [Citada en p. 111]

[HAFUCGH10] Pedro A. Hernandez-Avalos, Claudia Feregrino-Uribe, Rene Cumplido y Jose Juan Garcia-Hernandez, "Towards the Construction of a Benchmark for Video Watermarking Systems: Temporal Desynchronization Attacks," *Circuits and Systems, Midwest Symposium on*, tomo 0:páginas 628–631, 2010. [Citada en p. 111]

[HAFUCPGH65] Pedro Aaron Hernandez-Avalos, Claudia Feregrino-Uribe, René A. Cumplido-Parra y Jose Juan Garcia-Hernandez, "VIDEO WATERMARKING METHOD RESISTANT TO TEMPORAL DESYNCHRONIZATION ATTACKS," *United States Provisional Patent Application*, 19/07/2011 Application number: PCT/IB2011/001665. [Citada en p. 111]

[HG97] Frank Hartung y Bernd Girod, “Digital Watermarking of MPEG-2 Coded Video in the Bitstream Domain,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, tomo 4:página 2621, 1997, doi:http://doi.ieeecomputersociety.org/10.1109/ICASSP.1997.595326. [Citada en p. 14]

[HG98] Frank Hartung y Bernd Girod, “Watermarking of Uncompressed and Compressed Video,” *Signal Processing*, tomo 66(3):páginas 283–301, 1998. [Citada en p. 21]

[HG06] Amir Houmansadr y Shahrokh Ghaemmaghami, “A Novel Video Watermarking Method Using Visual Cryptography,” *International Conference on Engineering of Intelligent Systems*, páginas 1–5, 0-0 2006, doi:10.1109/ICEIS.2006.1703171. [Citada en p. 13]

[HKM05] Oztan Harmanci, Mehmet Kucukgoz y Mehmet Kivanc Mihcak, “Temporal synchronization of watermarked video using image hashing,” en Edward J. Delp y Ping W. Wong, editores, “Security, Steganography, and Watermarking of Multimedia Contents VII,” tomo 5681, páginas 370–380, SPIE, 2005, doi:10.1117/12.587437. [Citada en p. 22]

[HMU06] John E. Hopcroft, Rajeev Motwani y Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006, ISBN 0321455363. [Citada en p. 65]

[HS88] Chris Harris y Mike Stephens, “A Combined Corner and Edge Detector,” en “Proceedings of The Fourth Alvey Vision Conference,” páginas 147–151, Manchester, 1988. [Citada en p. 64]

[hZXjZjS07] Li he Zhang, Gang Xu, Ji jun Zhou y Lin jie Shen, “A Video Watermarking Scheme Resistant to Synchronization Attacks Based on Statistics and Shot Segmentation,” *Seventh International Conference on Intelligent Systems Design and Applications. ISDA 2007*, páginas 603–608, Octubre 2007, doi:10.1109/ISDA.2007.151. [Citada en p. 14, 22, 24, 27, 34]

[KDHM99] Ton Kalker, Geert Depovere, Jaap Haitsma y Maurice Maes, “A Video Watermarking System for Broadcast Monitoring,” en “In Proc. SPIE Security and Watermarking of Multimedia Contents I,” tomo 3657, páginas 103–112, San Jose, California, U.S.A., Ene. 1999. [Citada en p. 13, 22, 60]

[KKHW07] Sung Min Kim, Sang Beom Kim, Youpyo Hong y Chee Sun Won, “Data Hiding on H.264/AVC Compressed Video,” en M. Kamel

y A. Campilho, editores, “Image Analysis and Recognition,” tomo 4633 de *Lecture Notes in Computer Science*, páginas 698–707, Springer-Verlag, Berlin, 2007. [Citada en p. 14]

[KOGD04] Hyung Cook Kim, Hakeem Ogunleye, Oriol Guitart y Edward J. Delp, “The Watermark Evaluation Testbed (WET),” en “Security, Steganography, and Watermarking of Multimedia Contents,” páginas 236–247, 2004. [Citada en p. 31]

[KP99] Martin Kutter y Fabien A. P. Petitcolas, “A Fair benchmark for image watermarking systems,” *Security and Watermarking of Multimedia Contents*, tomo 3657:páginas 226–239, Abril 1999. [Citada en p. 31]

[KWP08] Lin Kezheng, Yang Wei y Liu Pie, “Video watermarking temporal synchronization on motion vector,” *3rd International Conference on Intelligent System and Knowledge Engineering. ISKE 2008.*, tomo 1:páginas 1105–1110, Noviembre 2008. [Citada en p. 14, 22, 24, 27, 34]

[LD02] Eugene T. Lin y Edward J. Delp, “Temporal Synchronization in Video Watermarking,” en “Proceedings of the SPIE International Conference on Security and Watermarking of Multimedia Contents IV,” tomo 4675, páginas 20–25, San Jose, California, U.S.A., 2002. [Citada en p. 23, 24, 34, 66, 75, 81, 82]

[LD04] Eugene T. Lin y Edward J. Delp, “Temporal synchronization in video watermarking,” *IEEE Transactions on Signal Processing*, tomo 52(10):páginas 3007–3022, Octubre 2004. [Citada en p. 23, 24, 34, 53, 75, 81, 82]

[LMT02] Rosa Lancini, Ferdinando Mapelli y Stefano Tubaro, “A robust video watermarking technique in the spatial domain,” en “4th EURASIP-IEEE Region 8 International Symposium on Video/Image Processing and Multimedia Communications,” páginas 251–256, Zadar, Croacia, Junio 2002. [Citada en p. 22]

[LSH06] Hongmei Liu, Fenglian Shao y Jiwu Huang, “A MPEG-2 Video Watermarking Algorithm with Compensation in Bit Stream,” en “Digital Rights Management. Technologies, Issues, Challenges and Systems,” tomo 3919, páginas 123–134, Springer-Verlag, Agosto 2006. [Citada en p. 14]

[LSKL03] Jeroen Lichtenauer, Iwan Setyawan, Ton Kalker y Reginald Lagendijk, “Exhaustive Geometrical Search and the False Positive Watermark Detection Probability,” en “Proceedings of SPIE Security,

Steganography, and Watermarking of Multimedia Contents V," tomo 5020, páginas 203–214, Santa Clara, CA, 2003. [Citada en p. 22]

[MPA09] "Motion Picture Association of America," Febrero 2009, http://www.mpaa.org/piracy.asp. [Citada en p. 3]

[MvOV96] Alfred J. Menezes, Paul C. van Oorschot y Scott A. Vanstone, *Handbook of applied cryptography*, CRC Press, 1 edición, 1996. [Citada en p. 12]

[NSB02] Xiamu Niu, Martin Schmucker y Christoph Busch, "Video Watermarking Resisting to Rotation, Scaling, and Translation," en "In Proc. 8th. IEEE SPIE Security and Watermarking of Multimedia Contents IV," tomo 4675, páginas 512–519, San Jose, California, U.S.A., Ene. 2002. [Citada en p. 22]

[PAK98] Fabien A. P. Petitcolas, Ross J. Anderson y Markus G. Kuhn, "Attacks on Copyright Marking Systems," en David Aucsmith, editor, "Information Hiding, Second International Workshop, IH'98," tomo 1525 de *Lecture Notes in Computer Science*, páginas 218–238, Springer, Portland, Oregon, U. S. A., Abril 1998. [Citada en p. 31]

[PBBC97] A. Piva, M. Barni, F. Bartolini y V. Cappellini, "DCT-based watermark recovering without resorting to the uncorrupted original image," en "Image Processing, 1997. Proceedings., International Conference on," tomo 1, páginas 520 –523 vol.1, oct 1997, doi: 10.1109/ICIP.1997.647964. [Citada en p. 61]

[Pet00] Fabien A. P. Petitcolas, "Watermarking schemes evaluation," *IEEE signal processing*, tomo 17(5):páginas 58–64, Septiembre 2000. [Citada en p. 31]

[PVM+01] Shelby Pereira, Sviatoslav Voloshynovskiy, Maribel Madueño, Stéphane Marchand-Maillet y Thierry Pun, "Second generation benchmarking and application oriented evaluation," en "In Information Hiding Workshop III," Pittsburgh, PA, USA, Abril 2001. [Citada en p. 31]

[Rho07] Geoffrey B. Rhoads, *Method for monitoring internet dissemination of image, video and/or audio files*, Digimarc Corporation, United States, Patent number 7171016, Enero 2007. [Citada en p. 3]

[SLY06] Jiande Sun, Ju Liu y Hua Yan, "A 2DPCA-Based Video Watermarking Scheme for Resistance to Temporal Desynchronization," en "Advances in Neural Networks - ISNN 2006," tomo 3973 de *Lecture Notes in Computer Science*, páginas 312–317, Springer, 2006. [Citada en p. 22, 23, 24, 34]

[STN+01] V. Solachidis, A. Tefas, N. Nikolaidis, S. Tsekeridou, A. Nikolaidis y I. Pitas, "A benchmarking protocol for watermarking methods," en "2001 IEEE International Conference on Image Processing," tomo 3, páginas 1023–1026, Thessaloniki, Greece, Octubre 2001. [Citada en p. 31]

[TC11] Yuri Tsivian y Gunars Civjans, "CineMetrics.lv - Movie measurement and study tool database," Agosto 2011, http://www.cinemetrics.lv/. [Citada en p. 33, 102]

[TKANR05] Mercan Topkara, Ashish Kamra, Mikhail J. Atallah y Cristina Nita-Rotaru, "Viwid: Visible watermarking based defense against phishing," tomo 3710:páginas 470–483, 1 2005. [Citada en p. 4]

[WIP96] WIPO, "WIPO Copyright Treaty," *World Intellectual Propierty Organization*, Diciembre 1996, http://www.wipo.int/treaties/en/ip/wct/trtdocs_wo033.html. [Citada en p. 1]

[WZWP05] Guomin Wu, Yueting Zhuang, Fei Wu y Yunhe Pan, "A Novel Watermarking Scheme Based on Video Content," en Z. Chen, editor, "Digital Libraries: International Collaboration and Cross-Fertilization," tomo 3334 de *Lecture Notes in Computer Science*, páginas 449–252, Springer-Verlag, Berlin, 2005. [Citada en p. 14]

[YG06] Liesen Yang y Zongming Guo, "A Robust Video Watermarking Scheme Via Temporal Segmentation and Middle Frequency Component Adaptive Modification," en "Digital Watermarking," tomo 4283 de *Lecture Notes in Computer Science*, páginas 150–161, 2006. [Citada en p. 22, 24, 26, 34]

[YL95] Boon-Lock Yeo y Bede Liu, "Rapid scene analysis on compressed video," *Circuits and Systems for Video Technology, IEEE Transactions on*, tomo 5(6):páginas 533 –544, diciembre 1995. [Citada en p. 57]