Auto-Recuperación Perfecta de Imágenes Naturales utilizando el Dominio de la Frecuencia

por

Fernando Álvarez Villalvazo

Tesis sometida como requerimiento parcial para obtener el grado de

Maestro en Ciencias, en el área de Ciencias Computacionales

por el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Febrero, 2018

Tonantzintla, Puebla

Supervisor:

Dra. Claudia Feregrino Uribe

Dra. Alicia Morales Reyes

Coordinación de Ciencias Computacionales

INAOE

A mi querida esposa.

Agradecimientos

A mi familia y amigos por el apoyo durante el trayecto de mis estudios.

Al Instituto Nacional de Astrofísica, Óptica y Electrónica, por la oportunidad de haber sido uno de sus estudiantes.

A mis asesores Doctora Claudia Feregrino Uribe y Doctora Alicia Morales Reyes por su maravillosa guía, apoyo y generosidad. Ha sido un privilegio trabajar bajo su supervisión, siempre mostrándome la dirección correcta. Al Doctor Sergio Bravo Solorio por su asesoría proporcionada durante el desarrollo de la tesis.

Al Consejo Nacional de Ciencia y Tecnología, por el apoyo económico para estudiar la maestría con la beca número 424153.

Índice general

Índice de figuras

Índice de tablas

Lista de acrónimos

BPP Bits per pixel.

DCT Discrete Cosine Transform.

DFT Discrete Fourier Transform.

DSWT Discrete Stationary Wavelet Transform.

DWT Discrete Wavelet Transform.

IWT Integer Wavelet Transform.

LSB Least significant bit.

MDS Maximum Distance Separable.

MSB Most significant bit.

MSE Mean Squared Error.

PSNR Peak Signal-to-Noise Ratio.

SHA Secure Hash Algorithm.

SSIM Structural Similarity Index.

Resumen

La protección de imágenes digitales es un reto en el ámbito científico debido a los altos requerimientos de poder de cómputo de los algoritmos de recuperación de información actuales y a la variedad de ataques que existen. La auto-recuperación perfecta es uno de los métodos para lograr la protección de información. En este trabajo se desarrolla un método de auto-recuperación perfecta en el dominio de la frecuencia utilizando la Transformada Entera Wavelet como medio de inserción. Con lo anterior se logra superar a los métodos reportados en la literatura con respecto al tiempo de procesamiento y a la cantidad de ataques soportados. Las mejoras alcanzadas en el tiempo de procesamiento se dan gracias a la disminución en el número de operaciones necesarias para recuperar la información perdida, y el aumento en la robustez se logra al soportar los ataques de cropping, tampering y modificación de 2 bits menos significativos.

Capítulo 1

Introducción

Durante las últimas décadas, el uso de los medios digitales de información ha incrementado año con año, debido a la transmisión o almacenamiento de una cantidad mayor de datos. Dentro de la información producida en todo el mundo se encuentran las imágenes digitales, cuyo número aumenta día con día, por ejemplo, en la plataforma Instagram se publican más de 38 mil fotos por minuto [Leboeuf, 2016].

La distribución de una gran cantidad de fotos en la web es posible gracias a la disminución de los costos de adquisición de cámaras fotográficas y/o teléfonos inteligentes. El uso de imágenes digitales no solo aumentó en la vida personal sino también en la laboral, por ejemplo, para documentar la instalación de algún instrumento, documentación de algún tratamiento médico, vigilancia o exploración de terreno.

Al incrementar el número de imágenes digitales, ya sea para uso personal o de trabajo, el interés por la protección de las mismas va en aumento, principalmente en el envío de imágenes por la web. El emisor desea proteger la imagen al ser enviada, con la seguridad de que el receptor reciba la imagen original y no una versión distorsionada. Existen métodos de protección de imágenes que utilizan marcas de

agua, con las cuales se identifica la distorsión provocada y se recupera la información perdida [Hung and Chang, 2007], [Zhang and Wang, 2008]; siendo este enfoque el utilizada en el trabajo de investigación que aquí se presenta.

El interés por la protección de imágenes se ha vuelto indispensable debido en parte al uso de imágenes como medio de información y como identificación. Una de las dificultades de la protección es el fácil acceso a programas de edición de imágenes; las personas usan las imágenes como evidencia de un acontecimiento y al tener un fácil acceso a programas de edición las pueden distorsionar.

Dependiendo del contexto en que se utilicen las imágenes digitales, las consecuencias de ser distorsionadas pueden ser graves, por ejemplo, si estas son evidencia en un juicio, si son soporte para el diagnóstico de algún padecimiento médico, si muestra actividad militar en otros países, etcétera. Al desarrollar un método de protección de imágenes que pueda recuperar la información de la imagen original a pesar de haber sido distorsionada, se lograría el objetivo de la protección.

La protección de imágenes implica identificar las partes dañadas y recuperar la información perdida. En la literatura existen métodos de recuperación de contenido en imágenes, lo cuales es posible clasificarlos por la calidad de la imagen recuperada [Korus and Dziech, 2013]. Esta clasificación es posible hacerla en 2 grupos, métodos de recuperación aproximada y métodos de recuperación perfecta; los primeros obtienen imágenes recuperadas visualmente cercanas a la imagen protegida pero no son iguales, en cambio los métodos de recuperación perfecta recuperan exactamente los mismos valores, por lo tanto las imágenes protegidas y recuperadas son iguales.

Los métodos de recuperación aproximada y perfecta tienen tres pasos en común, la obtención de información de control, autenticación de contenido y recuperación de información perdida, por lo observado en los métodos de la Tabla 3.1 y 3.2 de la sección 3. Estos pasos se deben a que es necesario tener información redundante, identificar qué parte de la imagen está dañada y con esto poder recuperar la

información perdida. La información de control es información parcial y resumida de la imagen original, ésta se puede obtener usando los 5 bits mas significativos (MSB por sus siglas en Inglés), una de las técnicas para resumir la información de control son las funciones Hash [Zhang and Wang, 2008] o promedios de una región [Som et al., 2015].

1.1. Problemática

El problema principal en la protección de imágenes digitales enviadas a través de la web se da durante la transmisión, ya que una tercera persona puede interceptar las imágenes enviadas y modificarlas o el mismo receptor podría hacerlo. Dicha protección se puede lograr de diversas maneras, por ejemplo, utilizando un canal de comunicación oculto, cifrando las imágenes enviadas, añadiendo claves de acceso a las imágenes o utilizando la auto-recuperación de imágenes. Esta última es el área de estudio en este trabajo se investigación, el cual se centra en la recuperación de información perdida por un ataque a la imagen o imágenes enviadas. Dentro de la auto-recuperación de imágenes existe la auto-recuperación perfecta, la cual recupera la imagen protegida sin errores incluso después de sufrir distorsiones, este trabajo se centra en este tipo de auto-recuperación.

En el ámbito científico, militar y médico, existe interés por la protección de imágenes con la característica de pérdida nula de información (auto-recuperación perfecta). Un ejemplo en el área médica es que al perder información en imágenes radiológicas, se puede llegar a un diagnóstico erróneo por parte de los expertos; el diagnóstico es determinante para el tratamiento y la vida del paciente. En el ámbito militar el reconocimiento erróneo de tropas enemigas o aliadas también es determinante en la vida humana.

Dentro de las Ciencias Computacionales se investiga cómo resolver los incon-

venientes de proteger las imágenes al ser enviadas a través de Internet, esto supone un reto debido a la variedad de ataques que puede sufrir una imagen, su severidad y la complejidad de recuperar la información perdida. La cantidad de ataques, así como la severidad de los mismos se debe principalmente a la facilidad de acceso a programas de edición y múltiples herramientas. Los restos específicos observados del problema de la recuperación de información son los siguientes:

- Definición de la información de control para lograr una restauración perfecta.
- Disminución de la distorsión de la imagen una vez insertada la información de control.
- Decremento del costo computacional para la recuperación de información.
- Incremento de robustez ante mayor variedad de ataques.

El reto al definir la información de control radica en la obtención de ésta y en la identificación correcta del valor del pixel; ya que al ser recuperación perfecta, sin errores, considera que solo existe un valor recuperado exitoso y la información de control es el medio para encontrar con éxito el valor del pixel.

Al insertar información a una imagen, ésta sufre un cierto grado de distorsión. Existen distintas métricas para medir la distorsión provocada en las imágenes marcadas, las usadas en este trabajo de investigación son: la relación señal a ruido pico (PSNR) y el índice de similitud estructural (SSIM), por sus siglas en Inglés; estas métricas son las más usadas en la literatura de marcas de agua [Hore and Ziou, 2010], [Barni and Bartolini, 2004]. Para usos comunes de las Es necesario mantener la distorsión en los niveles más bajos y evitar así la pérdida de información al insertar una marca de agua [Korus and Dziech, 2013].

Al ser recuperación perfecta, la búsqueda del valor correcto se vuelve una búsqueda exhaustiva, por lo tanto, el costo computacional es elevado. El reto recae

en disminuir el costo computacional y así poder ser viable la implementación en la vida cotidiana.

La robustez es la capacidad de un método para soportar un mayor número de ataques con cierto grado de severidad. Al ser distintos ataques, la forma de infringir daño es diferente, por ejemplo, un ataque puede ser la modificación de el bit menos significativo (LSB), por sus siglas en Inglés y otro promediar los valores de pixeles vecinos para obtener un nuevo valor del pixel. Esta diferencia de daño crea el reto de diseñar estrategias que soporten el daño a distintos tipos de ataques.

Como resultado del análisis de los problemas mencionados, se presenta un método de auto-recuperación perfecta capaz de aumentar la cantidad de ataques soportados y reducir el costo computacional comparado con el estado del arte. Esto se realiza utilizando la herramienta de la Transformada Entera *Wavelet* (IWT), por sus siglas en Inglés, siendo los coeficientes obtenidos a partir de ella los utilizados como información a proteger y recuperar.

La verificación del funcionamiento de métodos de auto-recuperación de contenido, comúnmente se realiza aplicando ataques controlados a las imágenes de prueba. Existe un problema con esto debido a que cada autor produce los ataques para realizar sus pruebas y estos no representan un conjunto de pruebas estándar para medir el rendimiento de un sistema (*benchmark*). Existen diversos *benchmarks* para evaluar los esquemas de marcas de agua en imágenes [Kutter and Petitcolas, 1999], [Pereira et al., 2001] y [Herrigel et al., 2001]. De los cuales se seleccionará el más adecuado para el esquema propuesto.

1.2. Objetivos

Objetivo general:

El objetivo de este trabajo es desarrollar un método capaz de auto-recuperar una imagen natural, la cual esté libre de error al compararla con la imagen marcada (esquema de restauración perfecta), dicho método debe ser capaz de soportar los ataques de remplazo de contenido, *cropping* y modificación de 2 LSB.

Objetivos específicos:

- Identificar la información de control necesaria para lograr la restauración perfecta de una imagen.
- Desarrollar un método de inserción capaz de soportar *tampering*, *cropping* y modificación de 2 LSB, utilizando el dominio de la frecuencia.
- Identificar y verificar el funcionamiento del método propuesto utilizando el *benchmark* más adecuado.

1.3. Metodología

En la primera etapa se realizará un análisis del estado del arte, explorando los métodos de recuperación perfecta, destacando las ventajas y desventajas de los métodos. Al realizar la comparación entre los métodos, se seleccionará el mejor, tomando en cuenta la robustez ante ataques, la severidad de cada ataque soportado y la información de control utilizada para la auto-recuperación de contenido.

Una vez seleccionado el mejor método, éste se implementará para evaluar los resultados expuestos por el autor, observando ventajas y desventajas. Dentro de

dicha evaluación se comprobará la fiabilidad de la información de control y se identificarán opciones para mejorar el método. Debido a que en el trabajo propuesto se utilizará la información de control para la auto-recuperación de contenido, es de gran importancia asegurar su correcto funcionamiento, identificando las limitaciones de la misma.

Se realizará un análisis del estado del arte, explorando métodos de inserción de información (marcas de agua) que utilicen los coeficientes frecuenciales como medio de inserción y el análisis de los métodos de recuperación aproximada que utilicen el dominio de la frecuencia. Esto para identificar las posibles formas de utilizar el dominio de la frecuencia y seleccionar la más adecuada, observando los requerimientos de la información de control, ya que ésta es la parte medular de la recuperación de imágenes.

Se desarrollará el método de auto-recuperación perfecta utilizando la información de control seleccionada y la forma en que se utiliza el dominio de la frecuencia, comprobando el funcionamiento para los ataques de *cropping* y *tampering*. Adicionalmente se desarrollaran técnicas para la resistencia al ataque de modificación de LSB, para esto se modificará la información de control, adicionando información que ayude a resistir el ataque.

Debido a la falta de un *benchmark* estándar para la prueba de los métodos de recuperación de información, se seleccionará el *benchmark* más adecuado para el método desarrollado. Este *benchamark* se utilizará para verificar la robustez ante otro tipo de ataques no presentes en los métodos de recuperación perfecta. Para la selección del *benchmark* se tomará en cuenta el tipo de dato de entrada, el tipo de ataques que contiene, la cantidad de ataques y el uso en la literatura.

En la Figura 1.1 se muestra el diagrama general de la metodología propuesta.

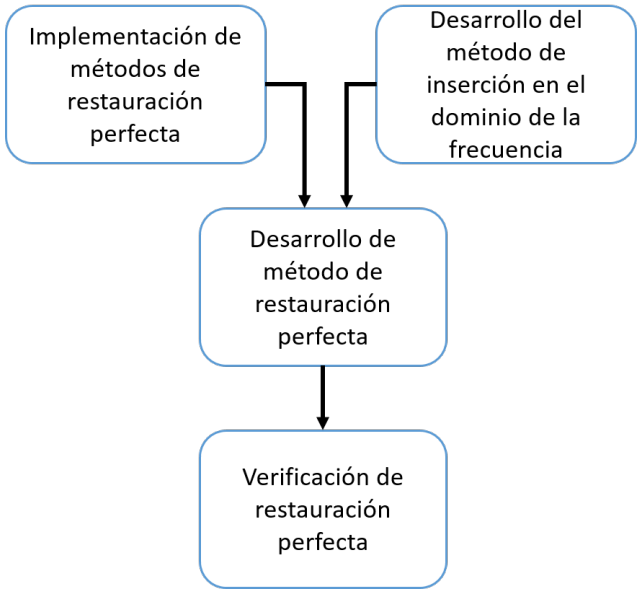

Figura 1.1: Diagrama general de la metodología.

1.4. Organización de la tesis

La organización de esta tesis es la siguiente: en el capítulo 2, marco teórico, se definen y muestran los fundamentos para una mejor compresión del método propuesto. En el capítulo 3, trabajo relacionado, se presenta una revisión detallada del estado del arte que permitió la definición del esquema desarrollado. En el capítulo 4, auto-recuperación perfecta utilizando el dominio de la frecuencia, se define el método desarrollado, que incluye la inserción, autenticación y recuperación de información. En el capítulo 5, experimentos y resultados, se muestra la comparación entre la implementación del método base y sus modificaciones, mostrando los resultados obtenidos del método propuesto con respecto a la capacidad de recuperación de contenido, costo computacional, imperceptibilidad y robustez ante distintos tipos de ataques. En el capítulo 6, conclusiones y trabajo a futuro, se presentan los puntos más importantes de esta tesis y el trabajo futuro identificado.

Capítulo 2

Marco teórico

En este capítulo se presenta una descripción de las herramientas utilizadas para el desarrollo del trabajo de investigación, iniciando con las características de las imágenes digitales utilizadas y las métricas utilizadas para la evaluación de la distorsión en las imágenes. Después se presenta una descripción de las técnicas de marcas de agua, sus aplicaciones, tipos y descripción de los ataques más comunes. Después la auto-recuperación de imágenes, las principales técnicas y vertientes. Por último, una descripción de la Transformada *Wavelet* utilizada en este trabajo.

2.1. Imágenes naturales

Las imágenes naturales en términos simples son aquellas que representan el entorno en el que vivimos [Hyvearinen et al., 2009]. Éstas tienen distintas características que son útiles para el método propuesto:

- Imágenes capturadas por un dispositivo óptico-digital como las cámaras fotográficas digitales.
- Las imágenes no contienen modificaciones digitales, con excepción de la con-

versión de la escala de grises.

- La diferencia promedio de los 5 MSB de cada vecino a 1 pixel de distancia, debe ser menor a 5.

Un ejemplo de imágenes naturales son las imágenes tomadas en una calle en cualquier dirección en donde sea posible apreciar personas, vehículos, plantas, edificios o animales. De acuerdo al estudio [Hyvearinen et al., 2009], este tipo de imágenes tiene una distribución no uniforme; la identificación de la distribución estadística de las imágenes naturales es toda una área de estudio. Pero la característica más importante para este trabajo es la diferencia promedio de los vecinos a 1 pixel de distancia. Esto se debe al método utilizado para realizar la recuperación de información, esto debido a que más del 94 % de los casos la diferencia promedio de los vecinos es menor a 5 [Bravo-Solorio et al., 2012]; el análisis se realizo en imágenes naturales a escala de grises con 8 bits de profundidad de color.

2.2. Métricas de calidad en imágenes

En el ámbito de marcas de agua y en recuperación de imágenes se utilizan diversas métricas para medir la degradación sufrida por una imagen al insertar información en ella. A continuación, se presentan las métricas utilizadas en este trabajo para medir dicha degradación.

MSE

El error cuadrático medio (MSE), por sus siglas en inglés, es de las medidas más utilizadas en la literatura de marcas de agua, éste cuantifica la diferencia entre los valores marcados y originales. El MSE mide el promedio de los cuadrados de cada error, donde el error es la diferencia cuadrática entre el valor original y el marcado.

En imágenes se utiliza la ecuación 2.1, donde N y M representan las dimensiones de la imagen, f representa la imagen marcada y g_{ij} representa la imagen original [Navas et al., 2008].

$$MSEj, g = \frac{1}{N \times M} \sum_{i=1}^{M} \sum_{j=1}^{N} (f_{ij} - g_{ij})^2 \tag{2.1}$$

PSNR

Es la métrica más utilizada en la literatura de recuperación de información, la cual define la relación entre el valor máximo de una señal y la degradación causada a la imagen original por la marca de agua. Debido a los distintos rangos que pueden tener los pixeles de una imagen, [0-255] 8 bits, [0-4095] 12 bits, etc, el PSNR se expresa en escala logarítmica y utiliza el decibel como unidad de medida. Se utiliza la ecuación 2.2, donde MAX_I representa el valor máximo de los pixeles, esto depende de la cantidad de bits con el cual se representan [Navas et al., 2008]. Entre mayor sea el valor del PSNR equivale a una menor distorsión a la imagen original.

$$PSNR = 10 \times \log_{10} \left(\frac{MAX^2{}_I}{MSE} \right) = 20 \times \log_{10} \left(\frac{MAX_I}{MSE} \right) \tag{2.2}$$

SSIM

El Indice de Similitud Estructural (SSIM), por sus siglas en inglés, es un método para medir la similitud entre dos imágenes, tomando en cuenta la luminancia y contraste. Dicho indice de similitud es usado en menor medida por la comunidad científica, pero es una de las medidas que mejor describen la degradación en la imagen marcada ya que no solo mide la diferencia entre los valores de la imagen marcada y original, sino que toma en cuenta aspectos que son importantes para

la percepción del ojo humano (luminancia y contraste). Se utiliza la ecuación 2.3, donde μ es el promedio, σ^2 es la varianza y σ es la covarianza correspondientes a la imagen marcada x y la imagen original y. Los valores de $C_1 = (0.01 \times L)^2$ y $C_2 = (0.03 \times L)^2$, donde L es el valor máximo que puede tener un pixel. Los valores obtenidos de SSIM son entre 0 y 1, donde 1 se obtiene si y solo sí las dos imágenes son exactamente iguales [Wang et al., 2004].

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu^2{}_x + \mu^2{}_y + C1)(\sigma^2{}_x + \sigma^2{}_y + C_2)} \tag{2.3}$$

2.3. Marcas de agua

En la comunicación entre dos personas existe un canal de comunicación, el cual puede ser el aire, correo postal, correo electrónico, mensajería de texto, etc. El canal de comunicación para las marcas de agua digitales comúnmente es la web, esto por los medios digitales usados para la inserción de información y por ser el medio de comunicación más usado en todo el mundo.

El marcado de agua digital es el proceso de insertar una firma o una marca en algún medio digital (imágenes, video, audio, entre otros). La marca o firma se puede definir como un conjunto de bits que representan información y una imagen marcada es un medio digital con información insertada de preferencia de manera imperceptible a los sentidos humanos [Barni and Bartolini, 2004], [Nematollahi et al., 2016].

En la Figura 2.1 se observa el diagrama general del marcado de imágenes, este consta de dos procesos principales que es el método de inserción y extracción. Dentro del método de extracción existen 3 casos:

- La obtención de la marca insertada independientemente de la distorsión provocada a la imagen marcada.

- Recuperación de la imagen marcada independientemente del estado de la marca y/o imagen marcada.
- La obtención de la marca junto con la imagen marcada.

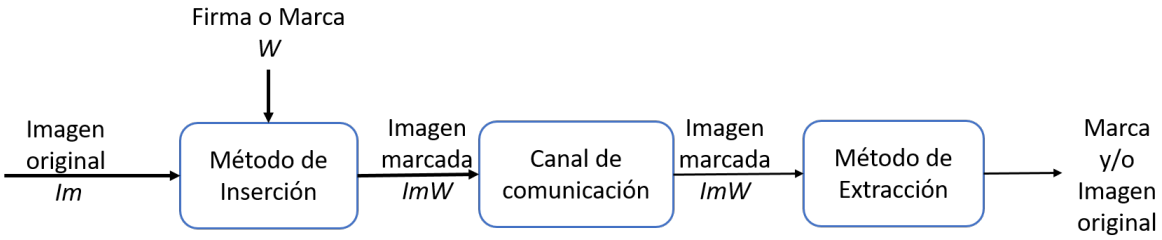

Figura 2.1: Diagrama general de marcas de agua en imágenes.

Los métodos que solo recuperan la marca están enfocados en darle un uso externo a la información insertada, por ejemplo, reclamo de derechos de autor, en este caso no importa si el medio donde se insertó la marca está dañado o no. Aquí lo importante es recuperar la información insertada, los datos del autor, y con eso comprobar la veracidad del producto o en su caso identificar al dueño.

La recuperación de la imagen marcada consiste en tener un medio dañado y con la ayuda de la marca de agua recuperar la información perdida durante el daño. Este uso de la marcas de agua es para proteger la integridad del medio digital [Fridrich and Goljan, 1999].

La obtención de la marca junto con la imagen original es usada principalmente en ambientes médicos, su objetivo es minimizar el daño causado al insertar la marca. Al insertar la marca en una imagen se introduce información redundante de la imagen original y así al extraer la marca se repara el daño causado por la inserción. Por ejemplo, si el medio de inserción son estudios médicos (imágenes), la identidad e historial clínico del paciente se introducen en la imagen, junto con información redundante o realizando operaciones que puedan revertirse; al extraer

dicha información es posible restaurar la imagen a su estado original con ayuda de la información redundante, asegurando así la calidad de los estudios. Cabe mencionar que estos esquemas aún no soportan ataques con un alto grado de severidad [Barni and Bartolini, 2004].

El diagrama general presentado en la Figura 2.1, puede ser modificado para un uso específico de marcas de agua, de acuerdo al propósito del método desarrollado. Un factor para diferenciar el uso de la marca de agua es conocer a partir de donde y cómo se forma la información que se insertará en la imagen, ya que puede ser a partir de la imagen original o ser tomada de otra fuente.

Los usos más comunes de las marcas de agua en imágenes son: la autenticación, la recuperación y el ocultamiento de información, protección de derechos de autor y huellas digitales.

En la autenticación de información, se parte de que la imagen marcada es atacada de alguna manera y en cierta medida. Dicha imagen marcada contiene información insertada, la información es usada para determinar qué pixeles han sufrido cambios (dañados) y qué pixeles están intactos (no dañados). Generalmente dicha autenticación se realiza por regiones, por ejemplo, la imagen puede ser divida en bloques para determinar su estado, dañado o intacto [Rey and Dugelay, 2002].

En la recuperación de información de imágenes se utiliza la información insertada para proteger la imagen en sí, es decir, se inserta información con la cual es posible recuperar la parte de la imagen dañada. Esta información perdida puede tratarse de bits dentro de unos cuantos pixeles o varios pixeles en regiones específicas, esto depende de la modificación realizada a la imagen [Barni and Bartolini, 2004].

El ocultamiento de información es utilizado para mandar mensajes entre dos o más personas, de manera que sea imperceptible la existencia de comunicación entre los participantes. Al enviar información oculta dentro de una imagen, es posible tener un canal de comunicación donde a simple vista solo existe un simple intercambio de

imágenes [Barni and Bartolini, 2004].

La protección de derechos de autor se realiza insertando los datos del autor en la imagen, dichos datos autentican al dueño de la imagen; esto es utilizado para evitar el uso indebido de imágenes por terceras personas. El método tiene como principal cualidad la resistencia a una variedad de ataques y con distintos niveles de severidad [Barni and Bartolini, 2004].

Las huellas digitales son un uso más profundo de los derechos de autor, estas tienen el objetivo de identificar a la persona que distribuyó el material de manera ilegal; se logra personalizando la marca insertada con los datos de la persona a la cual se le entregó la imagen protegida [Barni and Bartolini, 2004].

2.3.1. Tipos de marcas de agua

Las marcas de agua, independiente del uso que se les da, se pueden dividir dependiendo del dominio de inserción de la información. Existen principalmente dos dominios de inserción, el dominio espacial y el dominio de la frecuencia [Barni and Bartolini, 2004]. Una vez insertada la información, las marcas de agua se pueden clasificar por el tipo de información necesaria para extraer la marca del medio. Esto se refiere a la utilización de información que es necesaria para poder extraer la información insertada, sistemas ciegos y no ciegos [Cox et al., 2007].

Inserción en el dominio espacial

La inserción en el dominio espacial se realiza modificando directamente el valor en los pixeles con los que está compuesta la imagen. Este tipo de inserción tiene las características de tener un costo computacional bajo, ya que la inserción es realizada a través de sumas y multiplicaciones; se tiene un alto control de la máxima distorsión agregada a la imagen, ya que se sabe cuál es la diferencia entre el valor del pixel

original y el final; tiene una alta capacidad de inserción; tiene una baja resistencia ante ataques [Nematollahi et al., 2016], [Barni and Bartolini, 2004].

Inserción en el dominio de la frecuencia

Para realizar la inserción en el dominio de la frecuencia es necesario realizar una transformación a la imagen original, las transformaciones más utilizadas son, Transformada Discreta de Wavelet (DWT), Transformada Discreta Coseno (DCT) y Tranformada Discreta de Fourier (DFT) [Korus and Dziech, 2013].

Sistemas no ciegos

Para estos sistemas en la extracción de la marca es necesario contar con información extra de la imagen original, es decir, para la extracción es necesario tener acceso a la imagen original. Al utilizar la imagen original se considera que son sistemas no ciegos, estos sistemas son utilizados principalmente en la protección de derechos de autor ya que el autor cuenta con su imagen original.

Sistemas ciegos

En estos sistemas, el usuario receptor no necesita la imagen original para la extracción, éste solo necesita la imagen marcada y si es necesario, una llave para extraer la información ya que la llave se usa para tener un mayor nivel de seguridad. Estos sistemas son utilizados, por ejemplo, en ocultamiento de información, recuperación de imágenes, autenticación de información etc.

2.3.2. Ataques en imágenes marcadas

Un ataque a una imagen es la modificación de la información con la que está conformada la imagen, dicha modificación tiene un grado de severidad, este grado es medido dependiendo del tipo de ataque, por ejemplo, en porcentaje de daño, el valor de una variable, la cantidad de bits modificados, un porcentaje de calidad, etc. Existen ataques que modifican uniformemente a toda la imagen, cambiando un gran porcentaje de los pixeles presentes, por ejemplo, la adición de ruido aleatorio, compresión JPG o cualquier tipo de filtrado. Existe otro tipo de ataques que solo modifican un área específica de la imagen, por ejemplo, el *cropping* y *tampering*, que solo elimina/modifica una región, dejando intacto el resto de la imagen.

Los ataques pueden ser clasificados debido al motivo del ataque, ataques intencionados y no intencionados [Nematollahi et al., 2016]. Cabe mencionar que esta clasificación de ataques se centra en el motivo por el cual se realiza el ataque y no en el ataque en si, cualquier ataque puede ser intencionado o no intencionado dependiendo del contexto.

Ataques intencionados

Un ataque intencionado tiene el objetivo de borrar una marca de agua de cierta imagen. Al tener un canal de comunicación con transferencia de imágenes un atacante puede interceptar la imagen, el atacante al tener conocimiento de la existencia de una marca de agua en la imagen, este efectuara modificaciones especificas con tal de eliminar la marca. Estas modificaciones especificas pueden ser, por ejemplo, la eliminación de los LSB de cada pixel, el pase de un filtro a la imagen para promediar los valores originales, rotar ligeramente la imagen provocando el cambio de localización de los pixeles, etc. Este tipo de ataques son más efectivos, es decir, provocar el mayor daño posible, al conocer el método utilizado en la inserción de la información.

Ataques no intencionados

Los ataques no intencionados comúnmente los realiza el receptor de la imagen marcada, ya sea por desconocimiento del daño que puede provocar o por tratar de mejorar la calidad de la imagen. Algunos ataques pueden ser, el cambio de formato en la imagen, el cual puede provocar pérdida de información; el ajuste de brillo o contraste; ampliación o reducción de dimensiones.

Principales ataques

A continuación se muestra una lista con los ataques más comunes presentes en la literatura [Petitcolas et al., 1998], [Petitcolas, 2000]:

- *Cropping*, cortado parcial de imágenes, el corte de imágenes se realiza en porcentaje con respecto a las dimensiones (filas y columnas). Se realiza de dos formas, 1) corte de porcentaje en una sola dimensión ya sea en filas o columnas; la otra forma es de porcentaje en todas las direcciones, cortando un porcentaje de columnas de izquierda a derecha y de derecha a izquierda, también quitando un porcentaje de filas de arriba hacia abajo y de abajo hacia arriba.

- *Tampering*, sustitución de regiones en la imagen. La sustitución de regiones se realiza para cubrir información dentro de la imagen o cambiar la situación presentada en la imagen. Este ataque se realiza en porcentajes con respecto al número total de pixeles que forma la imagen, se puede sustituir por pixeles aleatorios, partes de otra imagen o pixeles en ceros.

- Adición de ruido aleatorio, es la adición de valores aleatorios que tienen un rango de $+-n$ en el valor de cada pixel perteneciente a la imagen atacada, donde n representa el nivel de daño.

- Modificación LSB's, es la sustitución de una cierta cantidad de bits en cada

pixel, comenzando por los bits menos significativos (LSB), para el nivel de daño se utiliza el número de bits modificados.

- Filtro promedio, se utiliza una matriz de tamaño $n \times n$, donde n es un número impar, esto para poder tener un centro en la matriz. Los valores dentro de la matriz se promedian para obtener el nuevo valor del pixel central, la matriz realiza un recorrido por toda la imagen. Este ataque genera pérdida de nitidez en la imagen, *blurring*.

- Filtro gaussiano, se aplicar un filtro utilizando una matriz de tamaño $n \times n$ la cual contendrá una distribución gaussiana en sus valores. La matriz realiza un recorrido por toda la imagen para multiplicar los valores de la matriz con los pixeles. Tiene los parámetros del tamaño de la matriz, los cuales deben ser números impares, esto para tener un centro en la matriz; y la desviación estándar será la que determine la distribución de la onda gaussiana representada en la matriz, en la practica entre mayor sea este valor mayor será el daño provocado al pasar el filtro por la imagen.

- Compresión JPG, es aplicar el formato de imagen JPG a la imagen, el cual es un tipo de compresión de imágenes con pérdida de información, éste formato de imagen tiene niveles de calidad de 0 a 100 %, entre menor sea el porcentaje de calidad mayor será la perdida de información.

- *Rescale*, ajuste en el tamaño de la imagen, utiliza interpolaciones y promedios.

- Rotación, rotar la imagen a partir de su centro.

- Rotación con corte, al rotar la imagen cortar las esquinas sobresalientes, dejando una imagen del tamaño original, rellenando con ceros los pixeles sin valor asignado.

- Ruido sal y pimienta, selección aleatoria de pixeles para cambiar su valor al mínimo o máximo.

En la Figura 2.2 se muestran los distintos ataques aplicados a una imagen. En el inciso a) se muestra la imagen original, b) imagen aplicando *cropping* en corte lateral, c) *cropping* en todas las direcciones, d) *tampering*, e) rotación y f) rotación cortando la imagen a las dimensiones originales.

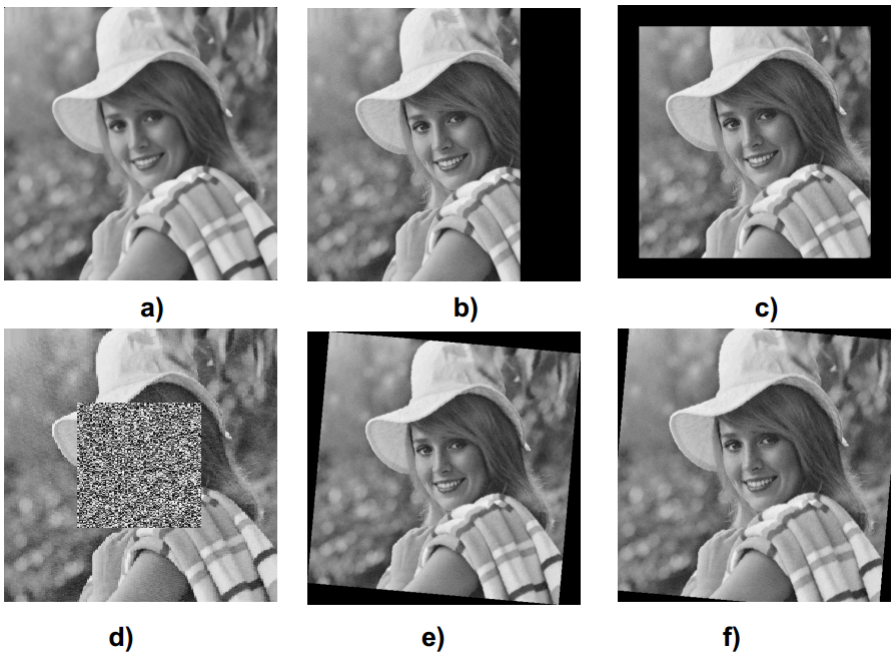

Figura 2.2: Ejemplo de ataques a una imagen.

2.4. Auto-recuperación de imágenes

En la auto-recuperación de imágenes se utiliza la autenticación de contenido y la recuperación de información, estos sistemas son sistemas ciegos. Al autenticar los pixeles, se obtienen dos conjuntos, los pixeles dañados y los pixeles sin daño. Es posible recuperar la información perdida a partir de la información que se reconoce como no dañada.

En la Figura 2.3 se observa el esquema general del método de auto-recuperación de imágenes. Este diagrama difiere del diagrama general de marcas de agua 2.1, esto

por la forma de obtener la marca a insertar, esta marca es obtenida a partir de la imagen original. El uso de la marca es recuperar las partes de la imagen que han sido dañadas.

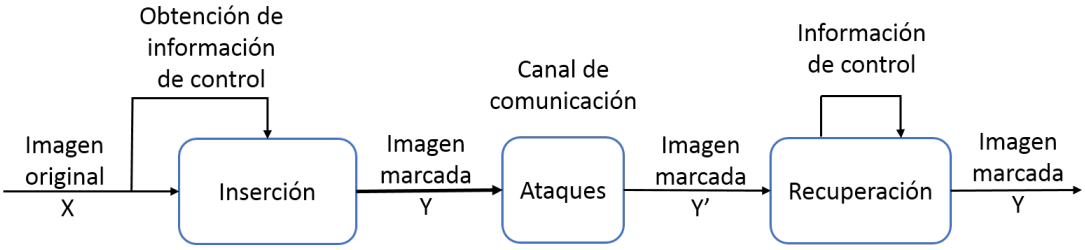

Figura 2.3: Diagrama general para la auto-recuperación de imágenes.

Dentro de la auto-recuperación de imágenes existe una clasificación en cuanto a qué información es la que será protegida, es posible proteger la información espacial (pixeles) o la información en frecuencia (coeficientes). También existe la clasificación con respecto al tipo de recuperación que se realiza, recuperación aproximada y recuperación perfecta.

2.4.1. Aproximada

La recuperación aproximada, tiene como objetivo recuperar la mayor cantidad de pixeles dañados, pero el pixel recuperado es una aproximación del pixel original ya que están enfocados en recuperar imágenes con una gran cantidad de daño, en algunos casos este daño puede ser hasta del 60 % de la imagen, con PSNR variable de 20dB-40dB [Korus and Dziech, 2013]. Los valores aproximados se pueden obtener insertando promedios de una cierta región, sustituyendo los LSB's por MSB's de otros pixeles/coeficientes o interpolación. Existen métodos que protegen los pixeles en el dominio espacial o los coeficientes en el dominio de la frecuencia.

2.4.2. Perfecta

La auto-recuperación perfecta se refiere a la restauración sin errores de la imagen dañada con respecto a la imagen marcada. La imagen marcada puede ser dañada en cierto grado y ser recuperada sin errores, con ello, al ser recuperada puede volverse a utilizar sin necesidad de tener la imagen original. Hasta la fecha solo se han propuesto métodos de recuperación perfecta que protegen los pixeles en el dominio espacial.

2.5. Transformada *Wavelet*

La Transformada *Wavelet* es ampliamente utilizada en las marcas de agua, dadas sus diferentes versiones como lo son la DWT, Transformada Entera Wavelet (IWT), Transformada Discreta Wavele Estacionaria (DSWT), por sus siglas en inglés; al tener distintas variantes, la cantidad de aplicaciones es amplia. En general se observa que en los métodos de marcas de agua que utilizan el dominio de la frecuencia, se obtienen buenos resultados en cuanto a robustez pero bajo nivel de inserción.

En este trabajo de investigación, debido a la naturaleza del método base [Bravo-Solorio et al., 2012] y dadas las características de representación entera de los coeficientes, información implícita de tiempo y frecuencia, la IWT fue seleccionada como medio de transformación del dominio espacial al dominio de la frecuencia. Los coeficientes resultantes de la aplicación de la IWT, son tomados como información a proteger y recuperar, dichos coeficientes son análogos a los pixeles en el dominio espacial pero en menor cantidad.

2.5.1. Análisis $Wavelet$

La Transformada $Wavelet$ al trabajar en los dominios temporal y de frecuencia, ofrece ventajas de preservación de mayor información con una sola transformada, el predecesor directo es la Transformada de $Fourier$. Al analizar una señal, ya no era suficiente contar con información de la frecuencia sino también era necesario saber en qué tiempo de la señal se encontraban las altas o bajas frecuencias. La Transformada $Wavelet$ soluciona este problema y con el paso de los años se han desarrollado las variantes mencionadas.

Esta transformada es eficiente para el análisis local de señales no estacionarias y de rápida transitoriedad; al igual que la Transformada de $Fourier$ con Ventana, mapea la señal en una representación de tiempo-escala. El aspecto temporal de las señales es preservado. La diferencia está en que la Transformada $Wavelet$ provee análisis de multiresolución con ventanas dilatadas. El análisis de las frecuencias de mayor rango se realiza usando ventanas angostas y el análisis de las frecuencias de menor rango se hace utilizando ventanas anchas [Amandí and Campo, 2006].

Las $Wavelets$, funciones base de la Transformada $Wavelet$, son generadas a partir de una función $Wavelet$ básica, mediante traslaciones y dilataciones. Estas funciones permiten reconstruir la señal original a través de la Transformada $Wavelet$ inversa.

La Transformada $Wavelet$ no es solamente local en tiempo, sino también en frecuencia. Dentro de los usos de esta herramienta podemos nombrar, además del análisis local de señales no estacionarias, el análisis de señales electrocardiográficas, sísmicas, de sonido, de radar, así como también es utilizada para la compresión, procesamiento de imágenes y reconocimiento de patrones. [Amandí and Campo, 2006]

De manera formal, la Transformada $Wavelet$ de una función $f(t)$ es la descomposición de $f(t)$ en un conjunto de funciones $\psi_{s,\tau}(t)$, que forman la base y son

llamadas $Wavelets$. La Transformada $Wavelet$ continua se define como:

$$W_f(s,\tau) = \int_{-\infty}^{\infty} f(t)\psi^*{}_{s,\tau}(t)dt \tag{2.4}$$

donde $W_f(s,\tau)$ es la señal transformada de $f(t)$ que es una función de dos variables "s,τ", los parámetros de escala y traslación respectivamente. Las $wavelets$ son generadas a partir de la traslación y cambio de escala de una misma función $Wavelet$ $\psi(t)_{s,\tau}$, llamada la $Wavelet$ madre, y se define como:

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right) \tag{2.5}$$

donde s es el factor de escala y τ es el factor de traslación.

Las $wavelets$ $\psi_{s,\tau}(t)$ generadas de la misma función $wavelet$ madre $\psi(t)$ tienen diferente escala s y ubicación τ, pero tienen la misma forma. Se utilizan siempre factores de escala $s > 0$. Las $Wavelets$ son dilatadas cuando la escala es $s > 1$, y son contraídas si $s < 1$. Así, cambiando el valor de s se cubren un amplio rango de frecuencias. Valores grandes del parámetro s corresponden a frecuencias de menor rango, o una escala grande $\psi_{s,\tau}(t)$. Valores pequeños de s corresponden a frecuencias de menor rango o una escala muy pequeña de $\psi_{s,\tau}(t)$ [Amandí and Campo, 2006], [Poularikas, 2010].

2.5.2. Transformada Discreta $Wavelet$

La DWT nació por la necesidad de calcular las $Wavelets$ de una señal de manera discreta, principalmente usada en cómputo, esto también para poder hacer cálculos más sencillos aunque se sacrifique precisión. En las señales continuas existe información redundante que al pasar a un medio discreto se pierde, la forma de representarlas es una aproximación pero en la mayoría de los casos es suficiente

trabajar con señales discretas. La ecuación para la DWT es la siguiente [Diego, 2008]:

$$DWT[m,n](x) = \frac{1}{\sqrt[2]{a_0^m}} \Sigma_n f(n) g\left(\frac{k - nb_0 a_0^m}{a_0^m}\right) \tag{2.6}$$

donde $f(t)$ es la $Wavelet$ madre, k es una variable entera que indica el número de muestra en la señal de entrada y los factores de escalado a y de traslación b se convierten en funciones discretas del parámetro entero m en la forma $a = a_0^m$ y $b = nb_0 a_0^m$. De este modo se origina una familia de funciones denominadas $wavelets$ hijas; cada una de ellas es la $wavelet$ madre con un determinado escalado y traslación.

A pesar de que es posible implementar la ecuación 2.6 de manera digital el número de operaciones sigue siendo elevado dado que depende del número de muestras con que está constituida la $Wavelet$ madre y el número de posibles valores que se le otorga a escalamiento. La implementación digital que convierte la DWT en una descomposición de la señal mediante un filtro pasa altas HP, y un filtro pasa bajas LP, fue propuesto por [Mallat, 1989].

A partir de la señal original $x(k)$ obtenemos:

$$c_1(n) = \Sigma_k h(k - 2n) x(k) \tag{2.7}$$

$$d_1(n) = \Sigma_k g(k - 2n) x(k) \tag{2.8}$$

donde $c_1(n)$ es la salida del filtro pasa bajas y $d_1(n)$ es la salida del filtro pasa altas.

Esta descomposición mediante los filtros pasa altas HP, y pasa-bajas LP, fracciona el espectro en dos bandas. A la salida del filtro LP se tiene una señal en el dominio del tiempo cuyo espectro está confinado entre 0 Hz y la mitad del espectro de la señal analizada. A su vez, la salida del filtro HP es una señal en el dominio del tiempo cuyo espectro está confinado entre la mitad del espectro y la frecuencia máxima de la señal. Este límite superior es la mitad de la frecuencia de muestreo,

[Diego, 2008]. En la Figura 2.4 se observa el diagrama del uso de filtros pasa bajas y pasa altas.

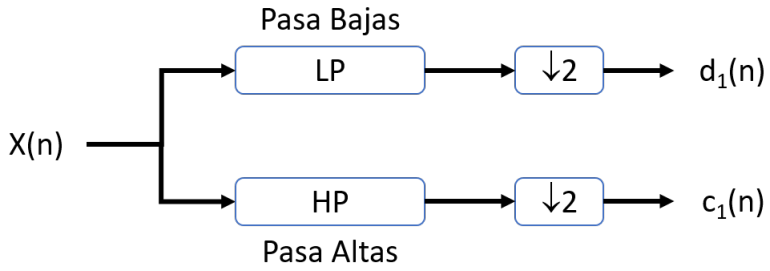

Figura 2.4: Diagrama general del uso de filtros para la DWT.

La salida del filtro pasa-altas da los detalles de las componentes de alta frecuencia, mientras que la salida del filtro pasa-bajas da las componentes de baja frecuencia. Esta salida puede ser de nuevo descompuesta en el siguiente nivel. De este modo se obtiene lo que se denomina árbol simple de descomposición *Wavelet*.

Con este conjunto de descomposiciones sucesivas se obtiene una secuencia de filtros *Wavelet* denominada árbol simple, por su representación gráfica típica que se muestra en la Figura 2.5

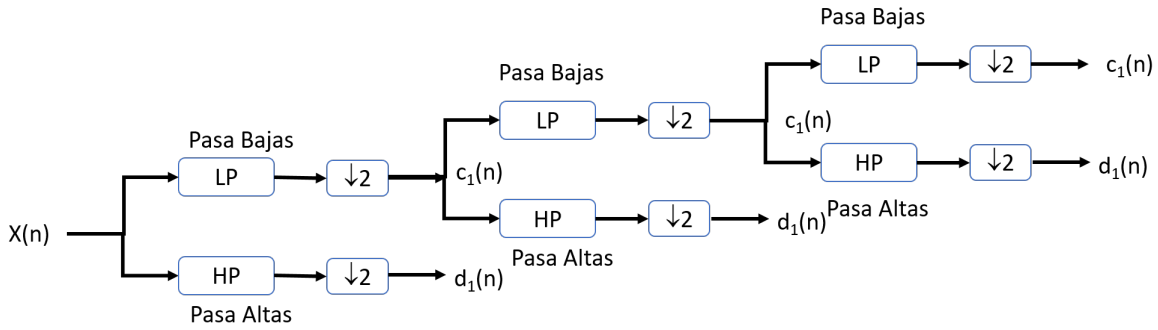

Figura 2.5: Diagrama general de descomposiciones sucesivas.

A la salida de cada filtro y antes del siguiente nivel de descomposición se eliminan uno de cada dos coeficientes, lo que se denomina *downsampling*. Estos

coeficientes son eliminados dado que no aportan información adicional en relación con el Principio de Incertidumbre, ya que al aumentar la resolución en frecuencia, se disminuye la resolución en tiempo.

2.5.3. Transformada $Wavelet$ en 2D

En la aplicación de la DWT en imágenes, se utiliza el esquema de filtrado, esto por las ventajas de disminución de operaciones y aplicación trivial de filtros, hablando en términos computacionales. Dado que digitalmente una imagen es una matriz de $n \times m$ donde cada valor dentro de la matriz contiene información, la aplicación de la DWT se realiza en dos etapas, 1) se aplica el filtrado en una dimensión, filas; 2) se aplica el filtrado en la otra dimensión, columnas.

Dado que son 2 filtros, se obtiene un total de 4 matrices de coeficientes llamadas, LL, matriz resultante de la aplicación del filtro pasa bajas en filas y columnas; LH, matriz resultante de la aplicación de filtro pasa bajas en filas y pasa altas en columnas; HL, matriz resultante de la aplicación del filtro pasa altas en filas y pasa bajas en columnas; HH, matriz resultante de aplicar el filtro pasa altas en las 2 dimensiones. En la Figura 2.6, se observa la aplicación de la DWT en 2D.

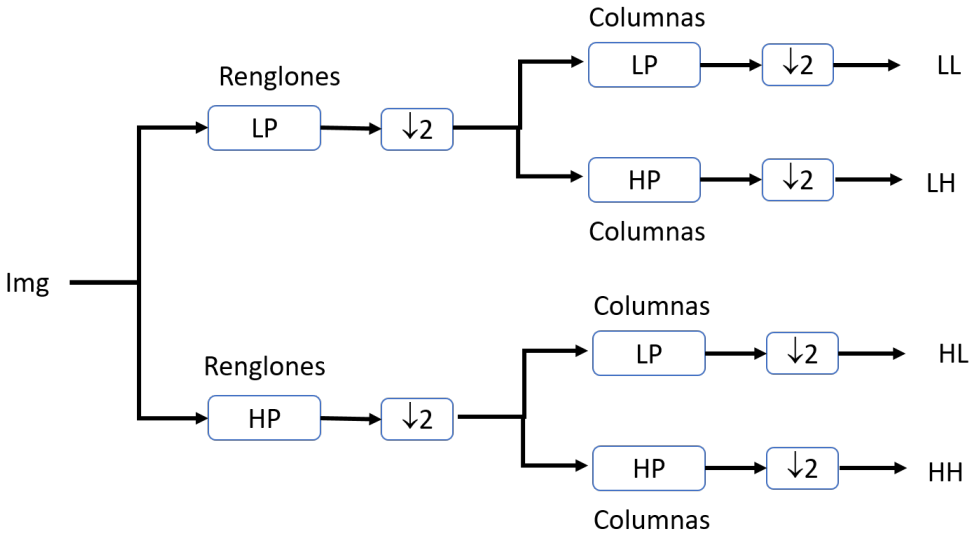

Figura 2.6: Diagrama general DWT en 2D.

2.5.4. Transformada Entera *Wavelet*

En muchas aplicaciones los datos de entrada para las Transformadas *Wavelet* corresponden a datos enteros, por ejemplo en imágenes. Pero la mayoría de las Transformadas *Wavelet* suponen que los datos de entrada son de punto flotante, regresando los coeficientes en valores de punto flotante. Redondear los coeficientes resultantes no es buena opción, ya que esto provoca una pérdida a la hora de reconstruir la señal original. Para solucionar esto se selecciona la IWT, la cual cumple con las características de mapear la señal entrante de enteros a coeficientes enteros, y a su vez, estos coeficientes reconstruyen la señal original sin pérdida de información, es decir, cumple con la reversibilidad requerida.

Esquema *lifting*

La IWT utiliza una modificación del esquema *lifting*, que utiliza la Transformada Haar con redondeos entre las operaciones. *Lifting* es un esquema que reduce

el número de operaciones necesarias para obtener los coeficientes de la Transformada $Wavelet$. Consta de 3 pasos principales para la transformación los cuales se muestran en la Figura 2.7, [Sweldens et al., 1995].

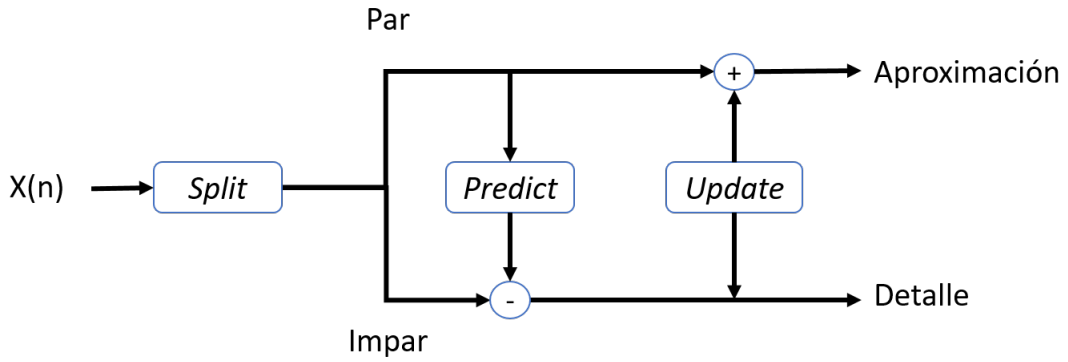

Figura 2.7: Representación del esquema $lifting$.

- *Split*: se realiza una división de la señal en dos subconjuntos, pares e impares.
- *Predict*: se utiliza un operador P, que predice el valor de cada muestra *par* y los valores impares son "predecidos" de los valores pares:

$$D_{i,j} = impares_{i,j} - P(pares_{i,j})$$

  donde $D_{i,j}$ es el conjunto de muestras de detalle de la señal $X(n)$.

- $Update$: en este paso se actualizan las muestras pares con la ayuda de los valores calculados en el paso de *predict*, $D_{i,j}$, las muestras pares serán reemplazadas con nuevos valores utilizando el operador de actualización U, aplicado a los valores $D_{i,j}$:

$$A_{i,j} = pares_{i,j} + U(D_{i,j})$$

  donde $A_{i,j}$, es el conjunto de muestras de aproximación de la señal $X(n)$.

El proceso para obtener la inversa del esquema $lifting$, es realizar las mismas operaciónes en orden inverso. En la Figura 2.8 se muestra la representación de la inversa

del esquema $lifting$, obsérvese que los signos de $update$ y $predic$ son invertidos, así como la inversa del $slip$ es el $merge$.

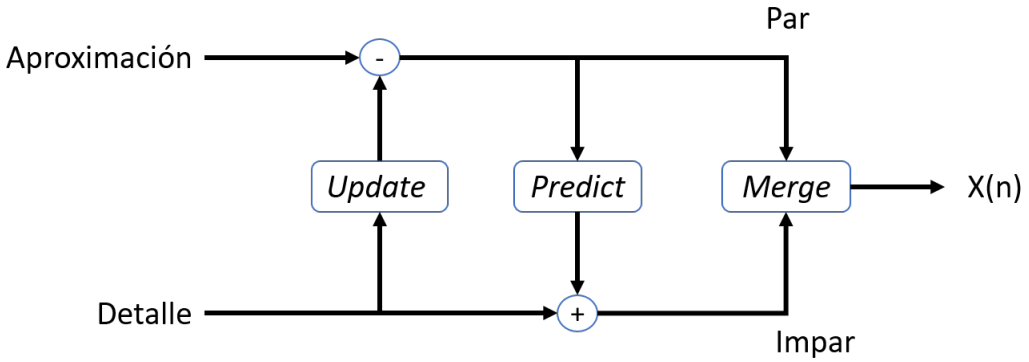

Figura 2.8: Representación del esquema inverso de $lifting$.

Transformada Haar

La Transformada Haar se conoce como la primera Transformada $Wavelet$, esta transformada es una de las más sencillas, fue propuesta por [Haar, 1910]. La Transformada Haar realiza promedios y restas entre valores vecinos que son las muestras tomadas de la señal de entrada. En la Figura 2.9 se muestra una representación de forma simple del método que utiliza la Transformada Haar, donde solo se toma en cuenta la diferencia de los valores en vez de utilizar el valor completo.

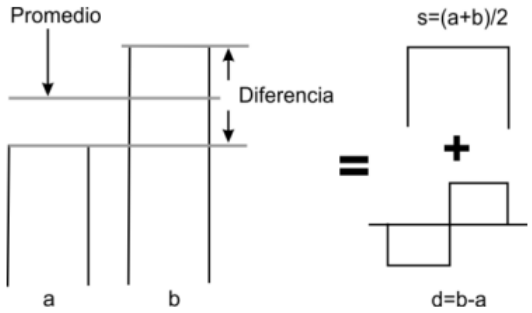

Figura 2.9: Representación de la Transformada Haar.

Las ecuaciones que utiliza la Transformada Haar en el esquema $lifting$ fueron propuestas por [Calderbank et al., 1998], en su documento detalla la obtención de las mismas y la adaptación que se tuvieron que hacer para el correcto funcionamiento en el esquema $lifting$, de manera gráfica la *wavelet* madre que corresponde a la transformada *haar* es una onda cuadrada, en comparación con otras ondas madre utilizadas esta es la más simple y se puede observar esta simplicidad en la figura 2.9 con las operaciones de suma y resta.

La aplicación de la IWT utilizada en este trabajo de investigación es la función propuesta por el software Matlab, $lwt2()$, la cual tiene como entrada la imagen original y el tipo de *wavelet* a utilizar, se utiliza la *wavelet haar* junto con el esquema $int2int$ el cual devuelve los coeficientes en formato entero a 8 bits. Se decidió utilizar dicha función ya que utiliza el esquema $lifting$, como ya se mencionó antes este tiene una eficiente implementación por la utilización de filtros y junto con la *wavelet haar* nos proporciona una transformada con pocas operaciones y con coeficientes enteros; otra característica importante es que dicha transformada es reversible, es decir los cambios en los coeficientes se ven reflejados en los pixeles y estos cambios se pueden observar a pesar del cambio de domino en repetidas ocasiones.

Capítulo 3

Trabajo relacionado

En esta sección se presentan los métodos de recuperación aproximada y recuperación perfecta que fueron base para la investigación presentada. De dichos métodos se exponen las fortalezas y debilidades de cada uno, así como las técnicas utilizadas para lograr el objetivo de la protección de imágenes digitales.

3.1. Recuperación de información aproximada

La recuperación de imágenes se remonta a finales de la década de 1990, con los trabajos propuestos por [Fridrich and Goljan, 1999], [Lee and Won, 1999], en dichos trabajos la distorsión soportada es un porcentaje bajo ($< 5\,\%$) y la recuperación es aproximada, es decir, la imagen protegida y la imagen recuperada no son exactamente iguales pero sí muy similares; con el tiempo se ha llegado a soportar un porcentaje de daño cercano al $90\,\%$ [Lee and Lin, 2008], las técnicas utilizadas han ido mejorando junto con la tecnología empleada para lograr que el costo computacional sea aceptable y viable de realizar.

Debido a que la auto-recuperación de imágenes se inició con la recuperación

aproximada se realizó un análisis de dichos métodos. En la Tabla 3.1 se muestran algunos de los esquemas que realizan auto-recuperación aproximada, se observa la existencia de métodos que utilizan el dominio de la frecuencia y otros el dominio espacial.

Dentro de los métodos que utilizan el dominio espacial, se observó que en estos métodos la imagen marcada tiene una baja distorsión, en comparación con los métodos que utilizan el dominio de la frecuencia, esto se debe al alto control de distorsión que se tiene al modificar bits en el dominio espacial. La distorsión en la imagen marcada es un punto importante en los esquemas de recuperación aproximada ya que la imagen recuperada tendrá un cierto porcentaje de error al compararla con la imagen marcada, si esta distorsión se mide con la imagen original la distorsión será mayor.

En cuanto a los ataques soportados, la gran mayoría de los métodos de auto-recuperación se centran en el ataque de *Tampering*, este ataque consiste en remplazar un cierto porcentaje de la imagen por otra imagen, se mide el porcentaje de pixeles sustituidos, el mayor porcentaje de daño es aproximadamente el 90 %. En segundo lugar está el JPEG, en el cual la calidad de compresión es de aproximadamente del 25 %.

Tabla 3.1: Métodos de recuperación aproximada.

Autor	Dominio de la inserción	PSNR de imagen marcada	PSNR de imagen recuperada	Ataques
[Hung and Chang, 2007]	DCT	[33.0,42.2]dB	[21.17,41.18]dB	$Tampering$, JPEG
[Zhang et al., 2010]	DCT	42.0 dB	[36.9,44.1]dB	$Tampering$, JPEG
[Phadikar et al., 2012]	IWT	35.27	[27.35,33.57]dB	$Tampering$, JPEG
[Noriega et al., 2011]	DCT,IWT	[22.5,42.0]dB	[22.5,44.2]dB	$Tampering$, Ruido sal y pimienta.
[Wang et al., 2011]	DCT	[36.5,39.12]dB	[20.78,27.09]dB	$Tampering$, JPEG.
[Wang et al., 2013]	DCT	[42.0,44.04]dB	[24.6,51.0]dB	$Tampering$.
[He et al., 2012]	Espacial	[37.92,51.14]dB		$Tampering$, promedio constante.
[Chang and Tai, 2013]	Espacial	[44.24]dB	[45.0,52]dB	$Tampering$, VQ, promedio constante.
[He et al., 2009]	Espacial	44dB	[40,42]dB	$Tampering$.
[Som et al., 2015]	DWT	[40,42] dB	[23.76-37.57]dB	$Tampering$.

El método propuesto por [Som et al., 2015] de recuperación aproximada, utilizando la Transformada Discreta $Wavelet$ (DWT), realiza la protección en los coeficientes de la transformada. El proceso es el siguiente: a partir de la imagen original se calculan los coeficientes con la DWT, de las matrices LL, HL, LH y HH se utiliza solo la matriz LL que contiene los coeficientes de bajas frecuencias. La matriz LL es dividida en 4 regiones A, B, C y D, esto se puede observar en la Figura 3.1, dentro de cada sección se forman bloques de 2×2, el bloque A1 de la matriz LL, es decir el bloque 1 de la sección A, junto con 2 bits de autenticación son insertados en los 3 LSB de los pixeles pertenecientes al bloque C1, y así con todos los bloques. La información de la imagen está distribuida en posiciones espejo, por lo tanto, al perder una sección entera es posible recuperar los valores de los coeficientes de la matriz LL solo con extraerlos de los pixeles en los bloques espejo. Este método reporta recuperar la imagen original con alta calidad teniendo un ataque de *tampering* del 60 % y con una calidad media-baja con ataque de *tampering* hasta de un 95 %.

30	58	62	64	65	57	55	56
37	119	114	115	115	116	111	106
38	121	115	109	112	110	114	104
37	108	121	109	114	113	105	109
38	115	124	118	110	118	106	112
37	114	118	106	113	109	113	111
36	110	107	113	103	114	110	112
36	110	115	103	110	113	113	102

Figura 3.1: Representación de dividir matriz LL y la correspondencia de los bloques se observa en los bloques marcados de color naranja.

En general, el comportamiento de los métodos de recuperación aproximada son similares, no hacen una búsqueda exhaustiva para los valores recuperados sino que se extraen parcialmente y los valores faltantes son calculados con valores aleatorios o por alguna técnica de reconstrucción de señales.

3.2. Recuperación de información perfecta

En la Tabla 3.2 se presentan los métodos de recuperación perfecta más importantes de la literatura. Dichos métodos tienen en común la utilización de las funciones Hash en la autenticación de pixeles. En la recuperación de información se utilizan principalmente 3 técnicas, uso de Hash, MDS *codes* y transformaciones utilizando matrices pseudoaleatorias. Los esquemas presentados dicha tabla, recuperan los 5 MSB de cada pixel dañado y utilizan los 3 LSB de cada pixel para insertar la información de control y autenticación.

Tabla 3.2: Métodos de restauración perfecta

Autor	PSNR de imagen marcada	*Tampering*	*Cropping*
Zhang 2008[Zhang and Wang, 2008]	28.7 dB	<3.2 %	No
Zhang 2009 [Zhang and Wang, 2009]	37.9 dB	<6.6 %	No
Zhang 2011[Zhang et al., 2011]	37.9 dB	<24 %	No
Dongmei 2015 [Niu et al., 2015]	37.9 dB	<33 %	No
Bravo 2012 [Bravo-Solorio et al., 2012]	37.9 dB	<25 %	<25 %

En general los métodos de recuperación perfecta se dividen en 3 pasos, inserción, autenticación y recuperación. En la inserción se obtienen la información de control a utilizar y los bits de autenticación, luego se insertan en la imagen. La autenticación es la sección en donde se identifican los pixeles dañados. La recuperación es la sección en donde a partir de la información de control extraída se recuperan los pixeles marcados como dañados.

El método de [Bravo-Solorio et al., 2012] utiliza 2 bits como información de control y 1 bit para autenticación, teniendo una distorsión en la imagen marcada con un PSNR promedio de 37 dB, soportando los ataques de *cropping* y *tampering*. Este método también se divide en 3 secciones:

- La sección de inserción forma subconjuntos pseudoaleatoriamente de m pixeles, a partir de ellos se obtiene 1 bit de información de control mediante la utilización de una función Hash de los 4 MSB de cada pixel por subconjunto, después este bit es insertado en el bit 3 de cada pixel por subconjunto; para la obtención del segundo bit de control se utilizan los 5 MSB de cada subconjunto, formados pseudoaleatoriamente y es insertado en el bit 2 de cada pixel.Para la obtención del bit de autenticación utiliza una función Hash con los 7 MSB de cada pixel.

- La autenticación se realiza por bloques de 8×8, por lo que si un pixel es dañado dentro de bloque 1, todo el bloque será tomado como dañado. Se utiliza un prefijo para identificar a los bloques dañados, dentro del prefijo se inserta información necesaria para soportar el ataque de *cropping*.

- En la recuperación de información se realiza una búsqueda exhaustiva de todos los posibles valores de los pixeles, se realiza en dos etapas, en la etapa 1 se reduce el número de posibles valores y en la segunda etapa se identifica el posible valor correcto.

El método de [Zhang et al., 2011], utiliza 2.5 bits para realizar la recuperación de información y 0.5 bits para la autenticación. El porcentaje de daño soportado por *tampering* es menor al 24 %, teniendo la imagen marcada un PSNR de 37.9 dB con respecto a la original.

- En la inserción, se toman los 5 MSB de cada pixel presente en la imagen, teniendo un total de N pixeles. Se forma un vector con dimensiones [1,5N], el cual es multiplicado por una matriz A_m para obtener $5N/2$ bits, con estos bits se forman N/64 subconjuntos de manera pseudoaleatoria. Se forman bloques de 8×8 pixeles, de cada bloque se introducen los 5 MSB en una función Hash para obtener 32 bits más los bits de algunos de los N/64 subconjuntos, se insertan de manera pseudoaleatoria a los 3 LSB de cada pixel por bloque de 8×8 pixeles.

- Para la autenticación se verifica si el n-ésimo bloque al calcular la función Hash de los 5 MSB corresponde a los bits insertados en dicho bloque, en caso de no coincidir el bloque se considera como dañado y por lo tanto cada bit también.

- En la parte de recuperación se utiliza la misma expresión que al realizar la inserción, pero en esta ocasión se conoce cuales bits de los vectores están dañados, se forman matrices con los bits dañados para obtener un sistema de ecuaciones

con n incógnitas con $5N/2 - n$ ecuaciones; para resolver este sistema se utiliza la eliminación Gauss Jordan.

Los métodos de [Zhang and Wang, 2008] y [Zhang and Wang, 2009] son precursores de [Zhang et al., 2011] por lo tanto trabajan de manera similar, pero con una mayor limitación en cuanto a porcentaje de daño.

El método de [Niu et al., 2015] es el que mayor porcentaje de daño soporta llegando al 33 %, soportando el ataque de *cropping*, teniendo en la imagen marcada un PSNR de 37.9 dB con respecto a la imagen original.

- En la inserción se forman bloques de 8×8 pixeles, donde por cada bloque se obtienen 160 bits de referencia, se introducen a los MDS codes 5 MSB de cada pixel por bloque, esto para reducir el número de bits. Para los bits de autenticación se utilizan los 5 MSB de cada pixel por bloque para ser introducidos en una función Hash y obtener 32 bits. Los 182 bits por bloque con insertados en los 3 LSB de cada pixel.
- Para la autenticación se comparan los bits Hash calculados a partir de los 5 MSB por bloques de 8×8, y los bits extraídos. Al no ser iguales, el bloque se considera un bloque dañado junto con todos sus bits.
- Para la recuperación se forman las matrices MDS *codes* con los bits extraídos, siendo identificados los bits dañados y no dañados. Se utiliza el proceso inverso de los MDS *codes* para poder encontrar los bits dañados.

3.3. Esquemas actuales en el dominio de la frecuencia

Dentro de las marcas de agua digitales existe un conjunto de métodos que utilizan el dominio de la frecuencia. Dado que en este trabajo de investigación se

utiliza la IWT para la protección de imágenes, se realizó un análisis de los métodos que insertan en el dominio de la frecuencia en la literatura. Se encontraron principalmente dos maneras de hacerlo que aplicados a la problemática presentada en este trabajo difieren en cuanto a la información que se protege y en cuanto al lugar de inserción de la información de control. A continuación, se presentan enfoques del uso del dominio de la frecuencia, dichos enfoques se obtuvieron a partir de los esquemas de inserción de información en frecuencia y métodos de recuperación aproximada en frecuencia.

3.3.1. Inserción en el dominio de la frecuencia

Usar al dominio de la frecuencia como medio de inserción. Para este enfoque se obtiene la información de control a partir de los valores de los pixeles en el espacio, dicha información es vista en formato de bits para poder ser insertada en el dominio de la frecuencia. En la Figura 3.2 se ilustra el enfoque uno, en donde el cambio de dominio representa cualquier transformada que cambie del dominio espacial al dominio de la frecuencia.

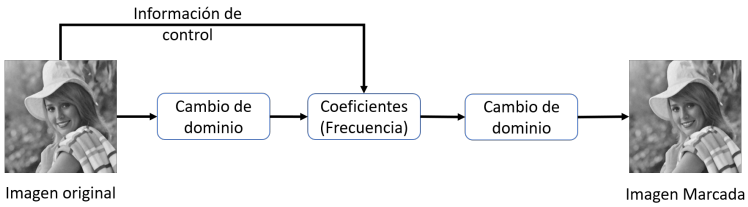

Figura 3.2: Diagrama de enfoque 1 de inserción.

Este enfoque conlleva a la solución de distintos problemas: 1) tener un método de inserción reversible, es decir, la información insertada pueda ser recuperada sin tener valores aproximados sino exactos; 2) al calcular la información de control de los 5 MSB de cada pixel, éstos no deben sufrir modificaciones al insertar información en el dominio de la frecuencia; 3) la cantidad de bits por pixel insertados en la

imagen, siendo necesarios 3 bits por pixel (bpp), al pasar la imagen al dominio de la frecuencia es necesario insertar 3 bits por cada coeficiente disponible.

El primer problema está parcialmente resuelto al utilizar una transformada entera, como lo es la Transformada Wavelet Entera, como los métodos propuestos por [Hernández, 2013], [Lee et al., 2007]. El segundo problema de evitar la modificación en los 5 MSB y el tercero de tener un mínimo de inserción de 3 bpp, están estrechamente relacionados. La relación entre estos dos problemas es que el método de inserción debe tener una baja distorsión en la imagen y al mismo tiempo una alta razón de inserción. Hasta la fecha no se encontró ningún método que utilice una transformada entera en la frecuencia que sea capaz de controlar la modificación en la imagen en los bits menos significativos y que al mismo tiempo inserte más de 2.5 bpp.

3.3.2. Protección y recuperación en el dominio de la frecuencia

Realizar la protección y recuperación de información en el dominio de la frecuencia, esto es, protegiendo los valores de los coeficientes aplicando el método propuesto por [Bravo-Solorio et al., 2012]. Transformar la imagen al dominio de la frecuencia, a partir de los coeficientes calcular la información de control y realizando la inserción de información en la frecuencia dentro de los LSB de cada coeficiente. En la Figura 3.3 se ilustra el enfoque 2.

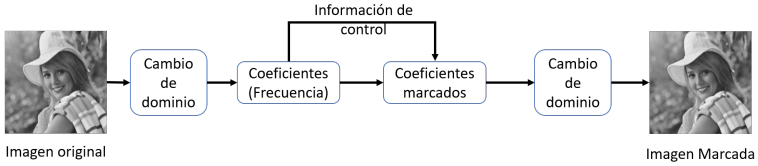

Figura 3.3: Diagrama de enfoque 2 de inserción.

Este enfoque conlleva a la solución de varios problemas: 1) el cálculo de la

información de control a partir de los coeficientes; 2) insertar 3 bits de información en los coeficientes; 3) la extracción de la información insertada debe ser sin errores.

Para el problema uno se debe utilizar una transformada entera, debido a que la representación de los coeficientes es similar a la de los pixeles, debido a que utilizan 9 bits en total, uno para el signo y 8 para el valor absoluto; al ser la representación similar al de los pixeles es posible obtener la información de control de los coeficientes.

El segundo problema tiene solución debido a que al modificar los 3 LSB de cada coeficiente es posible insertar la información de control sin realizar modificaciones a los bits de los coeficientes protegidos.

En el tercer problema, al utilizar una transformada entera se tiene la ventaja de que ésta es reversible, es decir, al calcularla y volver al dominio espacial no se pierde información espacial, todos los pixeles conservan el valor original. Y al realizar las modificaciones, éstas quedan guardadas implícitamente en los valores de los pixeles y podrán ser extraídas sin errores.

3.4. Discusión

Si bien un método resistente a un mayor porcentaje de pérdida de información es un método deseable, el aumentar la variedad de ataques también es necesario debido a la gran cantidad de ataques existentes. De los métodos presentados, se seleccionó el método de [Bravo-Solorio et al., 2012] como método base, esto por las ventajas que presenta al tener un bit por pixel dedicado exclusivamente a la autenticación, el control que se tiene sobre el orden de los bits y la resistencia ante ataques que posee.

Una ventaja marcada del método de [Bravo-Solorio et al., 2012] con respecto al método de [Zhang et al., 2011] y [Niu et al., 2015] es la menor cantidad de infor-

mación de control requerida para la restauración, además del soporte al ataque de *cropping* sin necesidad de realizar algún tipo de modificación. Gracias a la disminución de información requerida para la recuperación, es posible agregar información extra como el número de bloque y dimensiones de la imagen. Esta información es útil para la restauración de las dimensiones originales de la imagen en caso de sufrir el ataque de *cropping*.

Capítulo 4

Auto-recuperación utilizando el dominio de la frecuencia

El método propuesto por [Bravo-Solorio et al., 2012] es usado como método base para este trabajo. En la sección anterior se explican de manera general las técnicas utilizadas y en el apéndice A se extiende la explicación y se presenta la metodología utilizada por el autor.

La modificación principal al método base fue el cambio del dominio espacial al dominio de la frecuencia, esto referente a la información a proteger. El método base protege la información en el dominio espacial, mientras que el método propuesto protege los coeficientes de la transformada IWT en el dominio de la frecuencia. En el método propuesto se obtuvieron dos versiones, la primera versión soporta un ataque extra (modificación de LSBs) y la segunda versión soporta un mayor porcentaje de daño $\approx 1-2\,\%$ en comparación a la primera. Las modificaciones del método base se realizaron en la inserción de información y autenticación de información, se utilizó información adicional para las mejoras mencionadas.

Para trasladar la protección de información del dominio espacial a la protec-

ción de información en el dominio de la frecuencia, se realizó un análisis detallado, llegando a la conclusión de que era posible aplicar la técnica de recuperación de información en el dominio espacial del método base en el dominio de la frecuencia; esto debido a que la distribución de información que está presente en los pixeles es análoga a una de las matriz de coeficientes de la IWT. Se dice que son análogos debido a que la distribución de información en los 5 MSB en ambos casos es la misma, una de características de las imágenes naturales y que es utilizada por el método base, además de la posibilidad de representar los coeficientes y los pixeles de la misma manera (8 bits).

A partir lo mostrado en la sección 3.3, se seleccionó el enfoque dos, el cual consiste en la transformación de la imagen al dominio de la frecuencia para obtener la información de control de los coeficientes e insertar dicha información en los LSBs de los coeficientes. Dicho enfoque es utilizado en los métodos de protección y recuperación de imágenes aproximada en el dominio de la frecuencia. Con esto se controla la distorsión provocada a los coeficientes protegidos y debido al uso de la IWT es posible extraer la información insertada sin errores.

Al modificar el método base se desarrollaron 2 métodos diferentes:

- Método 1 BS_{Robust}, resiste un ataque extra y mejora la complejidad computacional con respecto al método base.
- Método 2 BS_{Damage}, aumenta en $\approx 1-2\,\%$ el daño que soporta la imagen en comparación con el método 1 y mejora la complejidad computacional con respecto al método base.

La solución propuesta al igual que el método de [Bravo-Solorio et al., 2012], se compone de 3 fases principales, en la Figura 4, se muestra un diagrama general de las tres fases.

- Inserción de información, en ésta se transforma la imagen del dominio espa-

cial al dominio frecuencial, usando la IWT, con los coeficientes se obtiene la información de control, la cual es insertada en los coeficientes, se obtienen los bits de autenticación y también se insertan, con esto se tienen los coeficientes marcados, ya con los coeficientes marcados se aplica la inversa de la IWT y se obtiene la imagen marcada.

- Autenticación de información, con la imagen marcada posiblemente dañada se aplica la IWT para obtener los coeficientes marcados, se extraen los bits de autenticación y con ellos se identifica a los coeficientes dañados y no dañados.

- Recuperación de información, con los coeficientes previamente identificados, es utilizada la información de control insertada y se aplica el método de recuperación de información, con esto se recuperan los 5 MSB de cada coeficiente dañado, se vuelve a aplicar la fase de inserción y así se obtienen los coeficientes marcados, con la matriz de coeficientes recuperada se aplica la inversa de la IWT y se obtiene la imagen marcada.

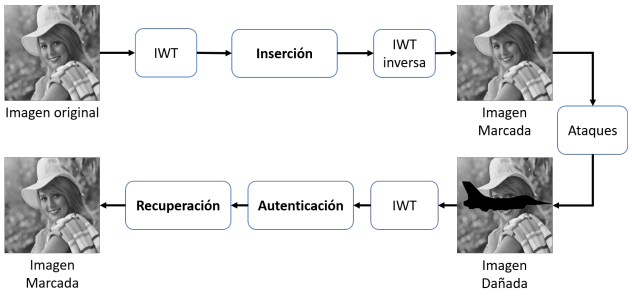

Figura 4.1: Diagrama general de la solución propuesta.

A continuación, se muestra el funcionamiento detallado de los dos métodos propuestos. Cabe mencionar que su funcionamiento está basado en el método de [Bravo-Solorio et al., 2012], se indica explícitamente las secciones en donde se realizaron las modificaciones y las diferencias con el método original. Se utiliza un pseudocódigo de cada fase para facilitar el seguimiento del método propuesto.

4.1. Método 1, BS_{Robust}: incremento de robustez y reducción de complejidad computacional

4.1.1. Inserción

Se consideran imágenes digitales naturales de 8 bits de profundidad de color, en escala de grises . La imagen de entrada $ImgOrig$ tiene dimensiones $n_1 \times n_2$, donde n_1 y n_2 son múltiplos de 8, siendo un total de N pixeles, $N = n_1 \times n_2$. A la imagen $ImgOrig$ se le aplica la IWT, línea 1 Algoritmo 1, obteniendo así 4 matrices de coeficientes: LL, LH, HL, HH cada una de las matrices son de tamaño $\frac{n_1}{2} \times \frac{n_2}{2}$.

De las 4 matrices de coeficientes disponibles, solo se toma en cuenta la matriz LL, esto debido a que la forma de representar esos coeficientes es análoga a la representación de los pixeles en el dominio espacial. Los coeficientes de LL, todos sus valores son positivos, se pueden representar en 8 bits, la distribución de los valores es similar a la distribución espacial y por lo tanto no es necesario hacer modificaciones extras al método de [Bravo-Solorio et al., 2012]. Las matrices HL, LH, HH contienen valores positivos y negativos, la distribución de los valores es diferente a la de los pixeles, necesitan un total de 9 bits para ser representados, dadas estas condiciones no es posible utilizar el método base.

Cada coeficiente se representa con 8 bits $b1, b2, \cdots, b8$ donde $b1$ es el bit menos significativo y $b8$ es el bit más significativo. Se eliminan los 3 LSB de cada coeficiente de la matriz LL. Las llaves k_1 y k_2 son formadas a partir de las dimensiones de la imagen, línea 3 y 4 del Algoritmo 1.

En la función $PseudoRandom$, línea 5 Algoritmo 1, se usa la llave k_1 para permutar de manera pseudoaletoria el orden de los coeficientes pertenecientes a la matriz LL. Para dicha permutación se utiliza el algoritmo Key Scheduling Algorithm

(KSA), el cual es utilizado en el cifrador RC4 [Rivest, 1987]. Ya permutados los coeficientes, se forman subconjuntos de $m = 16$ coeficientes, formando un total de $ns = N/m$ subconjuntos. Donde $x_{i1}, x_{i2}, \cdots x_{im}$ denota los coeficientes dentro del i-ésimo subconjunto.

Para cada subconjunto de coeficientes, líneas 6-9 Algoritmo 1, se calculan $m - bits$ de referencia por subconjunto, utilizando la ecuación 4.1, que usa los bits $b4, \cdots, b7$ de cada coeficiente perteneciente al subconjunto para el cálculo de los $m - bits$ de referencia.

$$r_i = H(\hat{x}_{i1}, \cdots, \hat{x}_{im}) \tag{4.1}$$

donde $H()$ es una función hash, usando el Algoritmo de Hash Seguro 2 (SHA-2), por sus siglas en ingles y $\hat{x}_{ij} \in [0, 15]$ los bits $b4, \cdots, b7$ de cada coeficiente es dado por:

$$\hat{x}_{ij} = \lfloor x_{ij}/8 \rfloor mod 16 \tag{4.2}$$

donde $\lfloor\ \rfloor$ es la función piso de x, devolviendo el valor entero mas grande menor que o igual a x y mod es la operación modulo. Después los bits de referencia r_ij son insertados en el $b2$ de cada coeficiente perteneciente al subconjunto de coeficientes x_{ij}, utilizando la siguiente ecuación.

$$x^{w}{}_{ij} = x_{ij} + r_{ij} \times 2 \tag{4.3}$$

Una vez que se tienen los coeficientes con los bits de referencia insertados en el $b2$, se ordenan los coeficientes a su posición original, línea 10 del Algoritmo 1.

Algoritmo 1: Inserción de información (parte 1 de 3)

entrada: ImOrig, I (Identificador)

salida : ImgMark

```
1  ( LL, HL, LH HH ) = IWT( ImgOrig );
2  LL = DeleteLSB( LL, [1-3] );
3  K1 = Concatenate( LL.Rows, LL.Cols );
4  K2 = K1 + 1;
5  SubsetCof = PseudoRandom( K1, LL );
6  for Index ← 1 to leng( SubsetCof ) do
7      MSB_b4b7 = SubsetCof[Index].ExtractBits( [4-7] );
8      ReferentBits = SHA2( MSB_b4b7 );
9      SubsetCof[Index].Insert( ReferentBits, 2 );
10 LL.updata(SubsetCof);
   /* LL actualizado con los ReferentBits insertados en bit 2     */
```

Los coeficientes actualizados que contienen los bits de referencia en el $b2$ son utilizados en una segunda etapa, se realiza una segunda formación de ns subconjunto, se usa la llave $k2$ y método KSA para la formación de subconjuntos de coeficientes permutados, línea 11 del Algoritmo 1. Por cada subconjunto de coeficientes, líneas 12-15 del Algoritmo 1, se toman los 5 MSB de cada coeficiente para introducirlos en la función hash $H()$ y obtener $m = 16$ bits de referencia, para ello se utiliza la siguiente ecuación:

$$r_i = H(\breve{x}_{i1}, \cdots, \breve{x}_{im}) \tag{4.4}$$

donde $\breve{x}_{ij} \in [0, 31]$ es dado por:

$$\breve{x}_{ij} = \lfloor x_{ij}/8 \rfloor \tag{4.5}$$

después, los bits de referencia resultantes se insertan en el $b1$ de cada coeficiente

perteneciente al i-ésimo subconjunto.

$$x^{w}{}_{ij} = x^{w}{}_{ij} + (r_{ij}) \tag{4.6}$$

Con esto se ordenan los coeficientes a su posición original, línea 16 del Algoritmo 1.

Algoritmo 1: Inserción de información (parte 2 de 3)

```
11 SubsetCof = PseudoRandom( K2, LL );
12 for Index ← 0 to leng(SubsetCof) do
13     MSB_b4b8 = SubsetCof[Index].ExtractBits( [4-8] );
14     ReferentBits = SHA2( MSB_b4b8 );
15     SubsetCof[Index].Insert( ReferentBits, 1 );
16 LL.updata( SubsetCof );
   /* LL actualizado con los ReferentBits insertados en bit 1     */
```

En la sección de la obtención de los bits de autenticación se realizó la primera modificación con respecto al método de [Bravo-Solorio et al., 2012], al modificar esta sección se obtuvo el método BS_{Robust}, la cual soporta un ataque extra.

Para obtener el bit de autenticación de cada coeficiente, se divide de la matriz LL con coeficientes marcados con los bits de referencia ($b1$ y $b2$), se forman bloques de 8×8 coeficientes, línea 17 del Algoritmo 1, en total se tiene $n_b = N/64$ bloques. Por cada bloque se obtiene un código $CodBlock$ distinto de 64 bits de longitud, líneas 18-25 del Algoritmo 1, el cual se construye de la siguiente manera:

$$Cod = I||BitsAutentic||p \tag{4.7}$$

donde $||$ indica la concatenación de los valores; I es un identificador de la imagen a marcar, puede o no ser único; p es el índice del bloque, denota el p-ésimo bloque;

$BitsAutentic$ son los bits de autenticación de los dos bits de referencia ($b1$ y $b2$), para la obtención de $BitsAutentic$ es necesario extraer los bits de referencia de cada coeficiente por bloque e introducirlos a la función hash $H()$ y así obtener 24 $BitsHash$. Obsérvese que Cod contiene un $prefijo$ común, I, el cual tiene una longitud γ de 20 bits, el $prefijo$ tiene la función de identificar los bloques que han sido dañados y los que no.

Para autenticar los 5 MSB de cada coeficiente se obtienen los $BitsHash$, línea 19 y 21 del Algoritmo 1, los 5 MSB se introducen a la función hash $H()$ y se obtienen 64 bits hash. Para la obtención de los bits de autenticación del bloque completo es necesario realizar una función XOR entre el $CodBlock$ del bloque y los $BitsHash$, Se utiliza la siguiente ecuación.

$$a_{p,j} = w_{p,j} \bigoplus h_{p,j} \tag{4.8}$$

donde $p = 1, 2, \cdots, n_b$; $j = 1, 2, \cdots, 64$, $a_{p,j}$ los bits de autenticación, $w_{p,j}$ los bits de $CodBlock$ y $h_{p.j}$ los $BitsHash$ de los 5 MSB del bloque.

Los bits de autenticación son insertados en el $b3$ de cada coeficiente perteneciente al bloque. En esta sección de la inserción se tienen los coeficientes de LL con sus 5 MSB intactos, el $b1$, $b2$ con información de control y $b3$ con información de autenticación, llamada matriz LL marcada. Después se aplica la inversa de la IWT, línea 27 del Algoritmo 1, para esto necesitamos las matrices LH,HL y HH, donde éstas contendrán valores de cero en cada coeficiente. La razón de tener las matrices que contienen altas frecuencias en cero es para poder realizar una recuperación libre de errores, ya que no es posible protegerlas utilizando el método de [Bravo-Solorio et al., 2012], y al no estar protegidas si se pierde algún coeficiente no sera posible saber el valor correcto, pero al colocar los coeficientes en cero es posible saber cual es su valor correcto.

El resultado de aplicar la IWT inversa es la imagen marcada, la cual tiene la información insertada para soportar los ataques de *cropping*, *tampering* y sustitución de LSB's. La imagen marcada es el objeto de retorno de la ejecución del Algoritmo 1.

Algoritmo 1: Inserción de información (parte 3 de 3)

```
17 Blocks = CreateBlocks( LL );
18 for Index ← 0 to leng(SetBlocks) do
19     MSB_b4b8 = SetBlocks[Index].ExtractBits( [4-8] );
20     LSB_b1b2 = SetBlocks[Index].ExtractBits( [1-2] );
21     BitsHash = SHA2(MSB_b4b8);
22     AutenticBitsLSB = SHA2( LSB_b1b2 );
23     CodBlock = Concatenate( I, AutenticBitsLSB, Index ) ;
24     AutenticBitsMSB = Xor( BitsHash, CodBlock );
25     SetBlocks[Index].Insert( AutenticBitsMSB, 3 );
26 LL.updata( SetBlocks ) LoadZero( HL, LH, HH );
27 ImageMark = InverseIWT( LL, HL, LH, HH );
28 Return ImageMark;
```

4.1.2. Autenticación

La fase de autenticación de información da como resultado la identificación de los coeficientes dañados y no dañados, además de restaurar las dimensiones originales en caso de estar presente el ataque de *cropping*. Los ataques (*cropping*, *tampering* o sustitución de LSBs) son realizados durante la transmisión de imágenes, una tercera persona intercepta la imagen la modifica y re-envía, dando al receptor la impresión de que la imagen no ha sufrido daños.

A partir de la imagen marcada dañada o no ($ImgMarkModif$) se aplica la IWT, línea 1 del Algoritmo 2, con la cual se obtienen las matrices LL, LH, HL, HH, de éstas solo se procesa la matriz LL; en caso de que las otras matrices tengan

valores diferentes a cero, se colocan ceros en cada coeficiente. Para la autenticación de información se utilizan los bits de autenticación, presentes en el bit 3 de cada coeficiente, debido a que dicha información esta insertada por bloques, la matriz LL se divide en bloques de 8x8.

Los bloques se construyen a partir de un coeficiente de inicio de formación de bloques, debido al ataque de *cropping* se tienen 64 posibles maneras de formar los bloques, desplazando el coeficiente de inicio en $i = [0, 1, \cdots, 8]$ y $j = [0, 1, \cdots, 8]$, donde $i = filas$ y $j = columnas$; ciclos mostrados en el Algoritmo 2 en las líneas 2 y 3. En la figura 4.2 se muestra un ejemplo del ataque de *cropping*, resaltando la división de los coficientes por bloques, en caso de realizar el corte de la matriz a mitad de un bloque, se deberá buscar el inicio de los bloques en una de las 64 posibles posiciones. Esto debido a que, si se inicia la formación de bloques en el lugar incorrecto no se encontrarán los bits de autenticación correctos. Cabe mencionar que cada bloque contiene sus propios bits de autenticación por esto no es posible autenticar la fusión de partes de dos bloques, solo bloques completos.

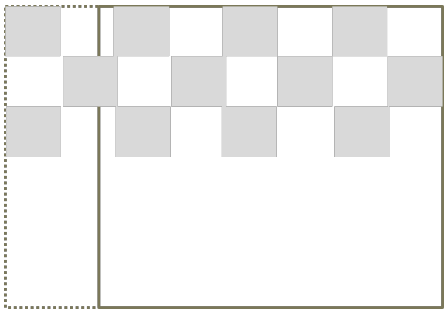

Figura 4.2: Ejemplo del ataque de *cropping*.

Por cada inicio de búsqueda de bloques se extraen los bits de autenticación, de ellos se obtiene el $CodBlock$ y se comparan entre sí para verificar si mínimo un 20 % de éstos son iguales, solo se comparan los MSB de cada coeficiente, los que corresponden al prefijo. Para obtener el $CodBlock$ se extrae el bit de autenticación correspondiente al LSB de cada coeficiente por bloque, a $a_{p,j}$, donde p es el numero de

bloque y j el numero de coeficiente perteneciente al bloque; se calculan los bits hash $h_{p,j}$ utilizando los 5 MSB de cada coeficiente, se operan los bits de autenticación y bits hash mediante una operación lógica XOR, obteniendo así el $CodBlock$ del p-ésimo bloque; se utiliza la siguiente ecuación.

$$w_{p,j} = a_{p,j} \bigoplus h_{p,j} \tag{4.9}$$

A partir de $CodBlock$ se obtienen el $prefijo$ de longitud $\gamma = 20$ bits, se agrupan los $prefijos$ iguales en un conjunto $\mathbf{A}$, si $|\mathbf{A}| > TL$, donde $TL = 20\,\%$ que es un umbral obtenido empíricamente por el autor del método base, si se cumple la condición anterior se considera que la matriz LL contiene información insertada.

Las dimensiones originales de la imagen están implícitas en la llave $k1$, en caso de *cropping* se comparan dichas dimensiones con las dimensiones de la matriz LL, en caso de ser diferentes se restaura la matriz a sus dimensiones originales, línea 14 del Algoritmo 2; para saber la dirección en que fue cortada la imagen y por lo tanto la matriz, se utiliza el número de bloque p, identificando la dirección de los bloques que faltantes.

Al tener el $prefijo$ correcto se procede a autenticar cada bloque, si el p-ésimo bloque tiene el mismo prefijo que el seleccionado como correcto se considera un bloque reservado o bloque sin daño, en caso contrario, el bloque se considera un bloque dañado y todos los coeficientes pertenecientes al bloque también.

Algoritmo 2: Autenticación de información

Data: ImgMarkModif, K1

Result: ImgMark

```
( LL, HL, LH HH ) = IWT( ImgMarkModif );
for I_x ← 1 to 8 do
    for I_y ← 1 to 8 do
        SetBlocks = ExtracBlocks( LL, [I_x, I_y] );
        ListCods = [ ];
        for Index ⇐ 1 to leng( SetBlocks ) do
            AutenticBitsMSB = SetBlocks[Index].ExtracBits( 3 );
            MSB_b4b8 = SetBlocks[Index].ExtracBits( [4-8] );
            BitsHash = SHA2( MSB_b4b8 );
            CodBlock = Xor( BitsHash, AutenticBitsMSB );
            ListCods.append( CodBlock );
        InfoCods = ComparatorCods( ListCods );
        if InfoCods.Percent > 20 % then
            CodCorrect = InfoCods.Correct;
            [LL, HL, LH, HH] = RestoreDimensions( LL, k1 ) ;
            SetBlocks = ExtracBlocks( LL, [I_x, I_y] );
            for Index ⇐ 1 to leng( SetBlocks ) do
                AutenticBitsMSB = SetBlocks[Index].ExtracBits( 3 );
                MSB_b4b8 = SetBlocks[Index].ExtracBits( [4-8] );
                BitsHash = SHA2( MSB_b4b8 );
                CodBlock = Xor( BitsHash, AutenticBitsMSB );
                AutenticBitsLSB = CodBlock.ExtracBits( [21-44] );
                SetBlocks[Index]=Autentication(SetBlocks[Index],
                  AutenticBitsLSB);
            LL.updata( Blocks );
            GoTo Fin ;
/* Si se encuentran más del 20% de CodBloks iguales; la matriz
   LL contiene información insertada                          */
Fin;
LLAutentic = Autentication( LL, CodCorrect );
Return LLAutentic;
```

Una segunda autenticación es realizada después de identificar los coeficientes dañados, ésta se realiza a los bits de referencia con la ayuda de $AutenticBitsLSB$, los cuales son obtenidos a partir del código $CodBlock$. En la primera autenticación solo se verifican los 5 MSB y ésta es la que dicta si el coeficiente es dañado o no. En la segunda autenticación se verifica si los bits de referencia están dañados o no; en caso de perder los 2 LSB, que pertenecen a los bits de referencia, la matriz LL se considera sin daños, ya que a partir de los 5 MSB se pueden obtener los bits $AutenticBitsMSB$ que contienen los $AutenticBitsLSB$.

El algoritmo 2 devuelve una matriz LL autenticada, marcando cada coeficiente con una bandera de coeficiente dañado o no, además de identificar si alguno de los bits de referencia (bits [1-2]) están dañados o no.

4.1.3. Recuperación de información

Para la recuperación de información se utiliza el mismo algoritmo que utiliza [Bravo-Solorio et al., 2012] aplicado a los coeficientes protegidos.

El proceso de recuperación es el siguiente: se permuta pseudoaleatoriamente la matriz LL utilizando la llave $k1$, formando así los subconjuntos de coeficientes de tamaño $m = 16$. De todos los subconjuntos formados están aquellos que no pueden ser procesados, los que están formados solo con coeficientes reservados y los que contienen más de 3 coeficientes dañados. Para la recuperación se procesan los subconjuntos que tienen máximo 3 coeficientes dañados y mínimo 1, esto para evitar una búsqueda en un espacio mayor a $16^3 = 4096$ combinaciones, son 16 debido a que por coeficiente tenemos 16 posibles combinaciones (4 bits) y elevado a la 3 potencia por el número de coeficientes dañados en un mismo subconjunto.

Si el subconjunto cumple con estas condiciones, el subconjunto se convierte en un subconjunto tratable y se realiza una búsqueda exhaustiva para encontrar los

posibles valores correctos. Por subconjunto tratable, líneas 6-10 del Algoritmo 3, se extraen los bits de referencia del i-ésimo subconjunto, que está en el $b2$ de cada coeficiente utilizando la siguiente ecuación.

$$r'_{ij} = \lfloor (y_{ij}/2) \rfloor mod 2 \tag{4.10}$$

donde r'_{ij} representa a los bits de referencia $ReferentBits$ y y_{ij} los coeficientes pertenecientes al j-ésimo subconjunto. Se calculan los bits de referencia a partir de los valores de los coeficientes existentes en el subconjunto, para esto se introducen en la función $H(\hat{y}_{i1}, \cdots, \hat{y}_{im})$ donde $\hat{y}_{ij} = (\lfloor y_{ij}/8 \rfloor mod 16)$, donde $\hat{y}_{ij}$ representa a los bits MSB_{b4b7} utilizados en la fase de inserción para obtener los bits de referencia . Si y_{ij} pertenece a un coeficiente reservado, éste se introduce en la función hash $H(\cdot)$, de lo contrario se realiza una serie de iteraciones para introducir los 16 posibles valores de los 4 bits desconocidos para cada coeficiente dañado. Al comparar los bits de referencia extraídos y calculados, se podrá conocer aquellos valores que sean candidatos a ser el valor correcto. La comparación se realiza bit a bit exceptuando los bits pertenecientes a coeficientes dañados.

Al terminar el proceso de búsqueda se obtiene un conjunto R ($Values$), línea 8 del Algoritmo 3, de posibles valores, los posibles valores son obtenidos con la comparación anterior de los bits de referencia calculados y extraídos. Los valores en el conjunto R son expandidos a 5 bits, línea 9 Algoritmo 3, se utilizan a los vecinos del coeficiente dañado para verificar si el valor expandido es candidato a recuperación o no. Los valores expandidos verificados son ligados a los coeficientes dañados, esto nos da un conjunto R con las posibles soluciones al coeficiente buscado.

Algoritmo 3: Recuperación de información

```
Data: LLAutentic, K1, K2
Result: ImgMark
NumCoefDamage = Damage( LLAutentic );
while NumCoefDamage > 0 do
    SetCoef = PseudoRandom(K1, LLAutentic);
    for Index ← 1 to leng(SetCoef) do
        NumCoefDamageSets = Damage( SetCoef[Index] );
        if NumCoefDamageSets <= 3 and NumCoefDamageSets > 0 then
            ReferentBits = SetCoef[Index].ExtractBits( 2 );
            Values = FindValues( SetCoef[Index], ReferentBits );
            ValuesVerif = ExpansionValues( Values.Neighbors, Values);
            SetCoef[Index].insert( ValuesVerif );
    LLAutentic.updata(SetCoef);
    SetCoef = PseudoRandom(K2, LLAutentic);
    for Index ← 0 to leng( SetCoef ) do
        NumCoefDamageSets = Damage( SetCoef[Index] );
        if NumCoefDamageSets <= 3 and NumCoefDamageSets > 0 then
            ReferentBits = SetCoef[Index].ExtractBits( 1 );
            Values = FindValues( SetCoef[Index], ReferentBits );
            SetCoef[Index].insert( Values );
    LLAutentic.updata(SetCoef);
    /* Aquellos Coeficientes con un solo valor en values es
       restaurado                                          */
    RestoreCoef(LLAutentic);
    NumCoefDamage = Damage( LLAutentic );
ImgMark = InverseIWT( LLAutentic, HL, LH, HH );
Return ImgMark;
```

Por ejemplo, si el i-ésimo coeficiente tiene un conjunto $R = \{7, 1\}$, la expansión sería $R = \{7, 23, 1, 17\}$, para la verificación de los valores se utiliza la vecindad espacial de un coeficiente de distancia, obteniendo un máximo de 8 vecinos, tomando en cuenta el valor de los vecinos solo si estos son no dañados. La diferencia entre el valor del coeficiente y el promedio de los vecinos debe ser menor a 5 para poderse tomar en cuenta como valor candidato a recuperar, en caso contrario se rechaza el valor. Si el valor promedio es 20, el conjunto quedaría como, $R = \{23, 17\}$, los valores con una mayor diferencia de 5 son eliminados del conjunto R, esto para reducir el número de búsquedas en la siguiente etapa.

Posteriormente, se permutan los datos de la matriz LL utilizando la llave $k2$, formando subconjuntos de $m = 16$ coeficientes. Se toman solo los subconjuntos con un máximo de 4096 posibles combinaciones, teniendo en cuenta que habrá coeficientes con pocos posibles valores en el conjunto R que fueron encontrados en la etapa anterior. Se extraen los bits de referencia ubicados en el bit 1, usando la siguiente ecuación.

$$r'_{ij} = (y_{ij}) mod 2 \tag{4.11}$$

donde r'_{ij} representa a los bits de referencia $ReferentBits$ y y_{ij} los coeficientes pertenecientes al j-ésimo subconjunto.

Los bits referencia calculados, se obtienen a partir de los coeficientes pertenecientes al i-ésimo subconjunto; se calculan los bits de referencia con: $H(\hat{y}_{i1}, \cdots, \hat{y}_{im})$ donde $\hat{y}_{ij} = \lfloor y_{ij}/8 \rfloor$. Si y_{ij} pertenece a un coeficiente reservado, éste se introduce en la función Hash $h(\cdot)$, si el conjunto R está vacío, se realiza una serie de iteraciones para introducir los 32 posibles valores de los 5 bits desconocidos,en caso de que el conjunto R contenga valores, la iteración se realiza únicamente con estos valores.

Al finalizar la búsqueda, por coeficientes solo quedan un número limitado de

valores, por lo que si un coeficiente se asocia a un único valor, este se sustituye y es marcado como recuperado, en caso de que existan más de un solo valor se utiliza al promedio de los vecinos espaciales del coeficiente en cuestión y si la diferencia entre estos es mayor a 5 es eliminado; si aún existe más de un valor estos son eliminados y el coeficiente vuelve a tomar el estado de dañado.

El proceso se repite en n-iteraciones de la permutación con $k1$ y después con $k2$, hasta que el número de coeficientes dañados sea 0, cabe mencionar que debido a que por iteración no todos los subconjuntos formados son tratables por lo que no es posible recuperar el 100 % de los coeficientes dañados en una sola iteración. El método funciona debido a que la información de control está distribuida en los dos subconjuntos, por ejemplo, si en la formación de los primeros subconjuntos utilizando $k1$ un coeficiente X pertenece a un subconjunto con 6 coeficientes dañados, este subconjunto no tratable, pero en la formación de los segundos subconjuntos utilizando $k2$, puede pertenecer a un subconjunto con solo 2 coeficientes dañados y éste se convierte en un subconjunto tratable, por lo cual el valor verdadero del coeficiente puede ser recuperado sin problemas.

El peor caso sucede cuando el i-ésimo coeficiente pertenece a un subconjunto en $k1$ con mas de 3 coeficientes dañados y también en $k2$ pertenece a otro subconjunto con muchos coeficientes dañados. En estos casos es posible que los valores verdaderos no sean recuperados, debido al constante reinicio de los subconjuntos o el no tratarlos.

4.1.4. Costo computacional

El costo computacional del método propuesto se puede dividir en las 3 fases, 1) inserción, 2) autenticación, 3) recuperación de información. Debido al alto coste que representa la recuperación de información se puede despreciar el costo computacional de las otras dos fases. Cabe mencionar que al comparar el método original y el

propuesto la mayor diferencia de costo computacional radica en la recuperación, a continuación se presenta el análisis del mismo.

El costo computacional de la recuperación es elevado dado que es una búsqueda exhaustiva. El costo computacional del método original está dado principalmente por el número de pixeles presentes en la imagen, esto es debido a que a partir del número de pixeles se obtiene el número máximo de subconjuntos y el número máximo de subconjuntos define el número máximo de búsquedas que se realizan en subconjuntos $k1$ y $k2$.

El número máximo de búsquedas se obtiene a partir del número de pixeles $NPix$ y el número de subconjuntos $Numsub$, que estan dados por:

$$NPix = n1 \times n2$$

$$NumSub = NPix/16$$

N es dividido entre 16 debido a que cada subconjunto contiene 16 pixeles. Para el máximo número de búsquedas en los subconjuntos formados utilizando $k1$:

$$MaxNumBusquedas_{k1} = \sum_{1}^{Numsub} 16^{PixDam}$$

donde $PixDam$ es el número de pixeles dañados que contiene cada subconjunto. Se toman en cuenta únicamente los subconjuntos con $PixDam > 0$ y 16 debido a que cada pixel dañado dentro de los subconjuntos tiene 16 posibles valores y la búsqueda se hace en relación a estos. Para el máximo número de búsquedas en los subconjuntos formados utilizando $k2$:

$$MaxBusqueda_{k2} = \sum_{1}^{Numsub} 32^{PixDam}$$

se utiliza 32 debido a que en este caso el número de posibles valores por pixel es 32. Para este cálculo se toma el número máximo de operaciones por este motivo no se toma en cuanta la reducción de búsquedas utilizando el conjunto R, entonces el número máximo de operaciones por iteración es:

$$MaxBusqueda_{k1} + MaxBusqueda_{k2}$$

Esto es por cada iteración, debido a que dentro de cada iteración se recupera un porcentaje de pixeles perdidos, en la siguiente otro porcentaje y así sucesivamente hasta que no existan pixeles dañados. La causa de que solo recupere un porcentaje es que por iteración no todos los subconjuntos son tratables, la gran mayoría son no tratables y solo de los tratables es posible realizar la recuperación de pixeles.

Para el calculo del costo computacional del método propuesto se sustituye la variable N (Número de pixeles) por $NCoef$ (Número de coeficientes), dando como resultado un costo computacional menor, se reduce el costo un 75 % en comparación con el costo del método original. Esto debido a que el número de coeficientes a recuperar es 75 % menos que el número de pixeles a recuperar, y al tener tal cantidad de coeficientes, el número máximo de subconjuntos disminuye de igual manera, así como el número búsquedas por iteración.

La reducción del número de coeficientes en comparación al número de pixeles es debido a que al aplicar la IWT a la imagen, con la cual se obtienen 4 matrices de tamaño $\frac{n_1}{2} \times \frac{n_2}{2}$, donde n_1 y n_2 son las dimensiones de la imagen.

4.2. Método 2, BS_{Damage}: mejora ante incremento de daño y reducción de complejidad computacional

4.2.1. Inserción

Para el cálculo de los bits de información de control se utiliza el mismo método que en el método propuesto BS_{Damage}, parte uno y dos del Algoritmo 1, la diferencia recae en el lugar de inserción de los mismos, siendo insertados en el $b3$ y $b2$.

Para obtener el bit de autenticación de cada coeficiente, se parte de la matriz LL con coeficientes marcados con los bits de referencia (bits 2 y 3), se forman bloques de 8×8 coeficientes, línea 1 del Algoritmo 4, siendo un total de $n_b = N/64$ bloques. A cada bloque se le insertará un código $CodBlock$ distinto de 64 bits de longitud, línea 2 a 8 del Algoritmo 4 el cual esta formado de la siguiente manera:

$$CodBlock = I||CoefExtras||p \tag{4.12}$$

donde $I =$ Indice de la matriz, es un identificador por matriz LL y por consecuencia es un identificador de imagen; $CofExtras$ son 5 MSB de 5 coeficientes pertenecientes a un bloque opuesto; $p =$ es el índice de bloque, denota el p-ésimo bloque. Obsérvese que $CodBlock$ contiene un $prefijo$ común, I, donde la longitud γ es de 20 bits. El $prefijo$ tiene la función de identificar los bloques que han sido dañados y los que no.

Para la obtención de los $CoefExtras$ y del bloque opuesto, líneas 5 y 6 del Algoritmo 4, se considera la formación de secciones del método propuesto por [Som et al., 2015]. La matriz LL es dividida en 4 secciones, A, B, C, D, como se muestra en la figura 4.3 a), el bloque $(1, 1)$ en la sección A, contendrá $CoefExtras$ de su bloque opuesto $(1, 1)$ de la sección D y el bloque $(1, 1)$ en B contendrá los

$CofExtras$ de su bloque opuesto en C. Como se observa los bloques en la sección A son los opuestos a la sección D y los bloques de la sección B son los opuestos de la sección C. Debido a que solo se seleccionan 5 coeficientes de los 64 pertenecientes al bloque opuesto, se utilizó la distribución representada en la figura 4.3 b) para la selección de coeficientes.

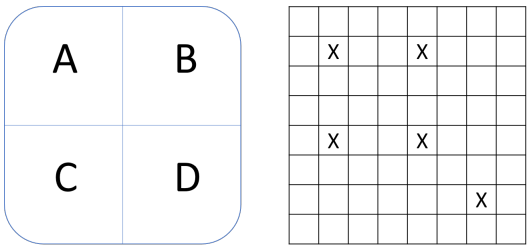

Figura 4.3: a) representa la división de la matriz LL para separa a los bloques en 4 secciones diferentes, b) representa la distribución de los coeficientes seleccionados para los $CofExtras$.

Algoritmo 4: Inserción de información (obtención de bit de Autenticación)

```
1  SetBlocks = CreateBlocks( LL );
2  for Index ← 0 to leng(SetBlocks) do
3      MSB_b2b8 = SetBlocks[Index].ExtractBits( [2-8] );
4      BitsHash = SHA2(MSB_b2b8);
5      CoefExtras = ExtractCoef( SetBlocks[Index].Opposite );
6      CodBlock = Concatenate( I, CoefExtras, Index ) ;
7      AutenticBits = Xor( BitsHash, CodBlock );
8      SetBlocks[Index].Insert( AutenticBits, 1 );
9  LL.updata( SetBlocks ) LoadZero( HL, LH, HH );
10 ImageMark = InverseIWT( LL, HL, LH, HH );
11 Return ImageMark;
```

Por bloque se obtienen los bits hash, líneas 3 y 4 del Algoritmo 4, para esto se extraen los 7 MSB de cada coeficiente y son introducidos a la función $SHA2()$, obte-

niendo así 64 $BitsHash$. Se realiza una función XOR, línea 7 del Algoritmo 4, entre el $CodBlock$ del bloque y los $BitsHash$, obteniendo así los bits de autenticación, se utiliza la siguiente ecuación:

$$a_{p,j} = w_{p,j} \bigoplus h_{p,j} \tag{4.13}$$

donde $p = 1, 2, \cdots, n_b$ (número de bloques) ; $j = 1, 2, \cdots, 64$(número de coeficientes por bloque); $a_{p,j}$ los bits de autenticación; $w_{p,j}$ los bits de $CodBlock$ y $h_{p.j}$ los $BitsHash$ de los 5 MSB del bloque.

Los bits de autenticación son insertados en el $b1$ de cada coeficiente perteneciente al bloque, línea 8 del Algoritmo 4. Al final de la inserción se tiene la matriz LL marcada, teniendo coeficientes con 5 MSB intactos, el $b3$, $b2$ con información de control y $b1$ con información de autenticación. Al tener la matriz LL marcada se aplica la inversa de la IWT y así obteniendo la imagen marcada, siendo ésta el resultado de la aplicación del Algoritmo 4.

4.2.2. Autenticación

Al igual que en el método BS_{Robust} a partir de la imagen dañada o no, se aplica la IWT, obteniendo la matriz LL, línea 1 del Algoritmo 5. La matriz LL se divide en bloques de 8x8 para extraer el $CodBlock$ de cada uno y verificar si existe o no información insertada en la matriz. Debido a que existen 64 posibles lugares de inicio para la formación de bloques se iteran cada una de las posiciones en búsqueda de $CodBlock$, líneas 2 a 17 del Algoritmo 5.

Por bloque formado se extrae el bit 1 de cada coeficiente, $AutenticBits$, después se calculan los $BitsHash$ utilizando los 7 MSB de cada coeficiente, para la obtención del $CodBlock$ se realiza una XOR entre los bits de autenticación y $BitsHash$, se

utiliza la siguiente ecuación.

$$w_{p,j} = a_{p,j} \bigoplus h_{p,j} \tag{4.14}$$

donde $a_{p,j}$ son los bits de autenticación; $h_{p,j}$ son los bits hash de los 5MSB y $w_{p,j}$ es el $CodBlock$ del p-ésimo bloque.

A partir de $CodBlock$ se obtiene el $prefijo$ de longitud $\gamma = 20$ bits, se agrupan los $prefijos$ iguales en un conjunto A, si $|A| > TL$, donde TL es igual al 20 % se considera que la matriz LL contiene información insertada. Las dimensiones de la matriz son recuperadas en caso de *cropping* con la ayuda de la llave $k1$.

Al tener el prefijo correcto se procede a autenticar cada bloque, si el p-ésimo bloque tiene el mismo prefijo que el seleccionado se considera un bloque reservado o bloque sin daño, en caso contrario el bloque se considera un bloque dañado y todos los coeficientes pertenecientes al bloque también.

Los coeficientes extras que se encuentran en los bloques no dañados son utilizados para recuperar los 5 MSB de aquellos coeficientes pertenecientes al bloque opuesto en caso de ser un bloque dañado. Con esto se recupera $\approx 10\,\%$ de los coeficientes dañados, con respecto al total de coeficientes dañados.

Algoritmo 5: Autenticación de información

Data: ImgMarkModif, K1

Result: ImgMark

```
( LL, HL, LH HH ) = IWT( ImgMarkModif );
for I_x ← 1 to 8 do
    for I_y ← 1 to 8 do
        SetBlocks = ExtracBlocks( LL, [I_x, I_y] );
        ListCods = [ ];
        for Index ⇐ 1 to leng( SetBlocks ) do
            AutenticBits = Blocks[Index].ExtracBits( 1 );
            MSB_b2b8 = Blocks[Index].ExtracBits( [2-8] );
            BitsHash = SHA2( MSB_b2b8 );
            CodBlock = Xor( BitsHash, AutenticBits );
            ListCods.append( CodBlock );
        InfoCods = ComparatorCods( ListCods );
        if InfoCods.Percent > 20 % then
            CodCorrect = InfoCods.Correct;
            [LL, HL, LH, HH] = RestoreDimensions( LL, k1 ) ;
            SetBlocks = ExtracBlocks( LL, [I_x, I_y] );
            for Index ⇐ 1 to leng( SetBlocks ) do
                AutenticBits = Blocks[Index].ExtracBits( 3 );
                MSB_b2b8 = Blocks[Index].ExtracBits( [4-8] );
                BitsHash = SHA2( MSB_b4b8 );
                CodBlock = Xor( BitsHash, AutenticBits );
                CoefExtras = CodBlock.ExtracBits( [20-44] );
                RestoreCoef(SetBlocks[Index].Opposite, CoefExtras);
            LL.updata( Blocks );
            GoTo Fin ;
/* Si se encuentran mas del 20% de CodBloks iguales; la matriz
   LL contiene información insertada                          */
Fin;
LLAutentic = Autentication( LL, CodCorrect );
Return LLAutentic;
```

4.2.3. Recuperación de información

Para la recuperación de información se utiliza el algoritmo 3, con la diferencia que al iniciar este proceso ya se cuenta con aproximadamente un 10 % de coeficientes dañados recuperados con los $CoefExtras$, esto es una ventaja en cuanto el tiempo de recuperación de información, en comparación con el método 1 BS_{Robust}.

4.2.4. Costo computacional

El costo computacional del método 2 BS_{Damage} es como máximo el mismo del método 1 BS_{Robust}, esto debido a el número de coeficientes a recuperar son aproximadamente 10 % menos en comparación con dicho método, debido a la recuperación de coeficientes utilizando los $CoefExtras$, que se encuentran dentro del $CodBlock$, en la fase de autenticación.

Capítulo 5

Evaluación empírica y análisis de resultados

En las siguiente sección se describen a detalle los datos de prueba utilizados, además de la cama de prueba de ataques que se aplicó a los datos para la validación de los métodos propuestos. Se incluye un análisis comparativo con otros métodos propuestos en el estado del arte y un análisis puntual de las mejoras alcanzadas por los esquemas propuestos.

5.0.1. Base de datos

Con el fin de evaluar el desempeño de los métodos de recuperación perfecta propuestos método 1 (BS_{Robust}), método 2 (BS_{Damage}) y el método base, se utilizaron las bases de datos de Inria [Dalal and Triggs, 2005], Pasadena House [Helle and Perona, 2000] y USC-SIPI [Weber, 1997]. En la tabla 5.1 se muestra un resumen de la descripción y tipo de contenido de imágenes contenidas en cada una de las base de datos.

Tabla 5.1: Base de datos usadas para evaluar la solución propuesta.

Base de datos	Número de imágenes	Dimensiones	Descripción
Inria	1313	640×480	Imágenes de personas y objetos cotidianos en calles.
Pasadena House	270	1760×1168	Fachadas de casas sin personas presentes.
USC-SIPI	145	255×255, 512×512, 1024×1024	Imágenes estándar, texturas y aéreas.

Para analizar del funcionamiento de la información de control utilizada en el método base [Bravo-Solorio et al., 2012] se utilizó la base de datos USC-SIPI [Weber, 1997], esto por contener imágenes naturales, no naturales y con distintas dimensiones, pudiendo observar así las ventajas y restricciónes de utilizar dicha información de control; se observo a un mal funcionamiento con imágenes no naturales, más adelante se presenta el funcionamiento de los métodos propuestos en imágenes naturales que al utilizar la información de control del método base los resultados no son favorables.

La razón por la cual la información de control del método base no funciona correctamente en imágenes no naturales, es debido a que la diferencia entre los 5 MSB de un pixel determinado y el valor promedio de los 5 MSB de sus vecinos a un pixel de distancia es mayor a 5, característica que solo presentan las imágenes naturales. Esta diferencia provoca que la información de control del método base no pueda determinar cual de los posibles valores obtenidos en la fase de recuperación de información es el correcto.

Las bases de datos, Pasadena House e Inria fueron utilizadas para evaluar el comportamiento de los métodos propuestos ya que éstas contienen solo imágenes naturales y contienen imágenes con distintas dimensiones, con lo cual se tienen imágenes de un mayor tamaño lo que conlleva a tener una mayor cantidad de coeficientes a proteger.

5.0.2. *Benchmark* utilizado

El *benchmark* o cama de pruebas seleccionado fue Stirmark 4.0 propuesto por [Petitcolas et al., 1998] y [Petitcolas, 2000], el cual contiene 15 diferentes ataques, de los cuales 9 están presentes en los ataques más comunes mencionados en la sección 2.3.2. Otra razón para la selección de este *benchmark* es por ser desarrollado para imágenes, tener una gran variedad de ataques y ser el más utilizado en la literatura [Barni and Bartolini, 2004]; sin embargo, este no cuentan con todos los ataques necesarios para validar los métodos de auto-recuperación perfecta (*cropping* circular, *tampering* y sustitución de LSBs) debido a que esta enfocado a la prueba de métodos de marcas de agua para transmisión de información oculta o derechos de autor, usos de marcas de agua mencionados en la sección 2.3.

Para tener el *benchmark* adecuado a la evaluación de los métodos propuestos se realizó la simulación de los ataques no presentes en el Stirmark utilizaron funciones de Matlab [Thompson and Shure, 1995] y también se realizó la simulación de los ataques del Stirmark [Petitcolas et al., 1998], [Petitcolas, 2000] en Matlab; obteniendo así un tener un solo *benchmark* y poder facilitar la utilización de las bases de datos. Con esto se cumple el propósito de tener un benchmark adecuando a las necesidades del método a evaluar.

Se aplicó el *benchmark* desarrollado a los dos esquemas propuestos: método 1 BS_{Robust} y método 2 BS_{Damage}. La hipótesis inicial del proyecto de investigación era que al insertar información en el dominio de la frecuencia esta información podría

resistir una variedad de ataques por ejemplo, filtro gaussiano, *rescale*, rotación, adición de ruido aleatorio, compresión JPG, solo por mencionar algunos, por lo visto en los métodos que insertaban información en el dominio espacial tabla 3.1; pero los resultados de aplicar el *benchmark* desarrollado no fueron los esperados, dado que los métodos propuestos (1 y 2) soportaron 3 ataques y 4 ataques respectivamente de los 18 simulados. Esto nos llevó a disminuir la lista de ataques simulados para la evaluación exhaustiva de los métodos propuestos y a descartar la hipótesis formulada con anterioridad.

5.1. Evaluación de esquemas propuestos

La evaluación de los métodos propuestos se realizo utilizando el *benchmark* desarrollado en Matlab el cual simula los ataques de *cropping*, *cropping* circular, *tampering* y sustitución de LSBs, los demás ataques mencionados anteriormente no serán utilizados por no ser soportados por el método base ni por los métodos propuestos.

La razón de no poder soportar la mayoría de los ataques utilizados en el Stirmark 4.0 fue la distribución de los pixeles dañados. Al dañar pixeles distribuidos en toda la imagen, en la autenticación de contenido más de la mitad de los bloques son marcados como dañados, por lo tanto el método no puede recuperar tal grado de daño, cabe mencionar que se utiliza la información de control y esta como máximo soporta un 25 % de daño en los pixeles presentes en la imagen.

Al utilizar la información de control para la recuperación de información propuesta por el método base, se pueden destacar algunos aspectos importantes:

- Los métodos propuestos protegen imágenes del tipo natural ya que son el tipo de imágenes que puede proteger el método base.

- Los ataques soportados por el método base son de igual manera soportados por los métodos propuestos, aunque se protejan los coeficientes y no los pixeles.
- Los ataques que no son soportados por el método base tampoco serán soportados por los métodos propuestos, esto por la similitud entre los pixeles y los coeficientes de la matriz LL.

La dependencia de ataques soportados y no soportados, se debe a que existe una similitud entre los pixeles en el dominio espacial y los coeficientes de la frecuencia (solo matriz LL); los dos se pueden representan utilizando 8 bits, la diferencia entre los 5 MSB de un pixel/coeficiente hacia los 5 MSB de sus vecinos es menor a 5.

5.1.1. Evaluación empírica

La evaluación empírica se refiere a la utilización de imágenes prueba, aplicarles los métodos propuestos, simular los ataques del *benchmark* desarrollado y recuperar la imagen marcada. Con esto se evalúan los limites de los métodos propuestos. Debido a que se utiliza la información de control del método base ya se cuenta con el limite superior de los métodos propuestos, que es máximo $25\,\% - 26\,\%$ de pixeles/coeficientes dañados, pero no se tiene con certeza el limite inferior.

Se utilizaron las imágenes de las bases de datos Pasadena House e Inria, ya que contienen solo imágenes del tipo natural y contiene imágenes con dimensiones mayores a 512×512, cabe mencionar que al evaluar el método base con la base de datos de USC-SIPI se observó que trabaja mejor con imágenes con dimensiones mayores a 512×512.

Los resultados se dividieron por base de datos y por tipo de ataque, para visualizar los resultados dependiendo de la cantidad de pixeles dañados en las imágenes mejorando así la visualización de los resultados.

En las figuras 5.1 a la 5.7 se muestran los resultados de los ataques de *cropping*, *cropping* circula, *tampering* y modificación LSB's respectivamente, aplicados a la base de datos Pasadena House e INRIA. En color azul se muestran los resultados del método base, en naranja el método propuesto BS_{Damage} y en gris el método propuesto BS_{Robust}.

Las gráficas muestran en el eje de las x la severidad del ataque; en el caso de *cropping* y *tampering* la severidad equivale al porcentaje de pixeles dañados, en el caso de *cropping* circular la severidad equivale al porcentaje de pixeles quitados en los 4 lados de la imagen, de la parte externa a la interna; y en modificación de LSB's corresponde al número de LSB's a los cuales se les sustituyó el valor. El eje y muestra el promedio del porcentaje de recuperación de pixeles.

Dentro de la evaluación empírica se observó que existen imágenes en las cuales no es posible recuperar el 100 % de los coeficientes sino ≈ 99.8 %, esto sucede tanto en el método base (en este caso serían pixeles) como en los métodos propuestos. La razón de esto es que al aumentar el número de pixeles dañados aumenta la posibilidad de tener subconjuntos no tratables y también de encontrar más de una posible solución del valor final por coeficiente dañado, al tener varios posibles valores y no es posible decidir cual de ellos es el correcto con la ayuda de los coeficientes vecinos el valor correcto no es recuperado; esto sucede cuando el porcentaje de pixeles dañados es mayor a 21 %. Cabe mencionar que esta perdida de coeficientes es en promedio menos del 0.4 % y dichos coeficientes pueden ser estimados con la ayuda de los coeficientes vecinos; aun así la imagen recuperada se puede considerar como una imagen marcada ya que los coeficientes perdidos pueden ser estimados y obtener una imagen marcada de la misma calidad que en la primera ocasión que se aplicó el método.

Al aplicar los métodos propuestos se modifica la imagen original, esta modificación se mide con el PSNR al comparar la imagen marcada y la imagen original, en promedio los métodos propuestos obtienen un PSNR de 28-30 dB, en comparación

con el método base que obtiene 37 dB este PSNR es bajo pero suficiente para pasar desapercibido al ojo humano.

A continuación se presentan los resultados separados por ataque y dentro de cada sección se observa la utilización de las distintas bases de datos.

Resultados aplicando ataque de *Cropping*

En las figuras 5.1 y 5.2 se muestran las gráficas de la simulación del ataque de *cropping* en las bases de datos Pasadena House e Inria. Las gráficas no muestran porcentajes de daño menores a 22 %, ya que en estos la recuperación es del 100 % y tampoco mayores a 26.5 % ya que con los métodos no se recupera algo significativo.

El eje x de las figuras 5.1 y 5.2 corresponde a la severidad del ataque de *cropping*, éste se mide en % de las columnas de la imagen cortadas.

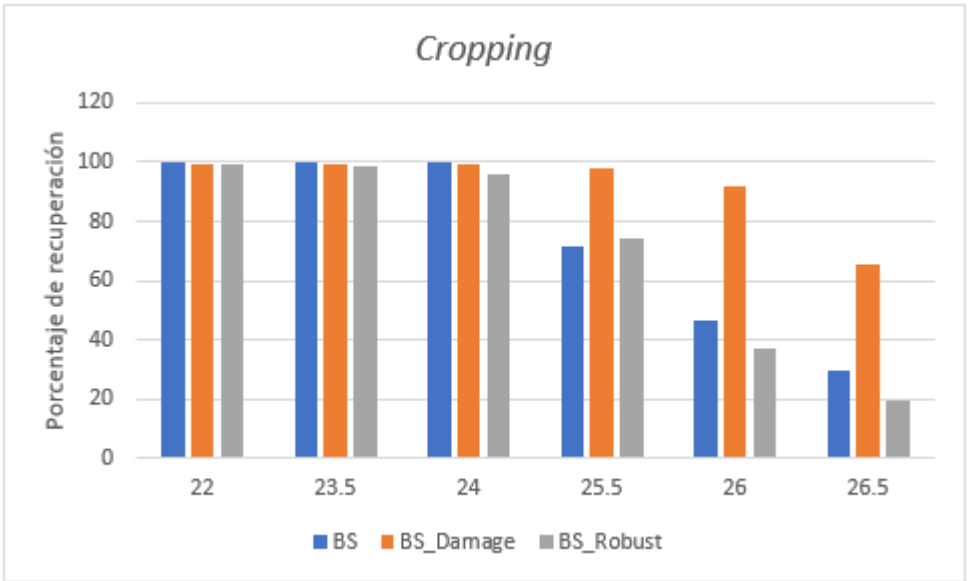

Figura 5.1: Gráficas de resultados del ataque *cropping* en la base de datos Pasadena House

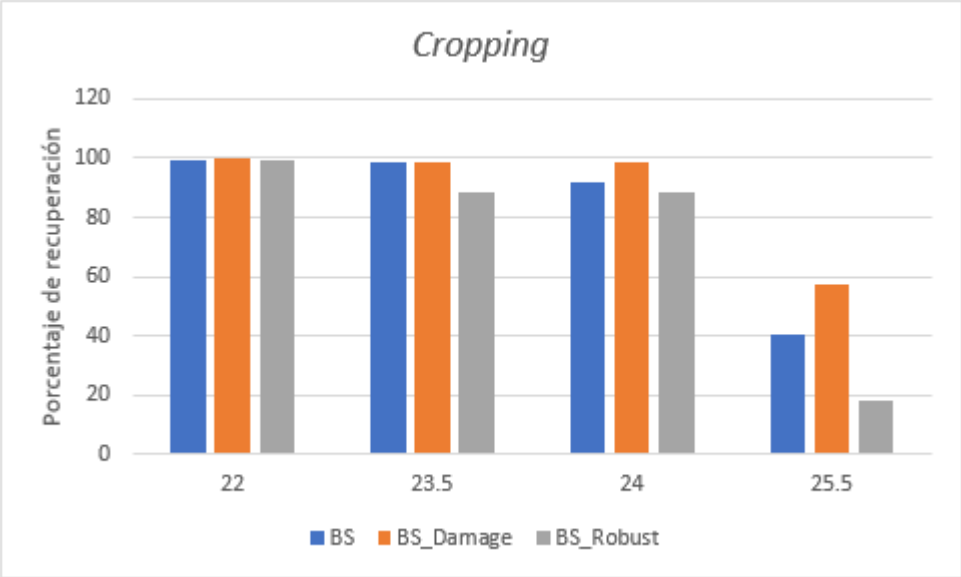

Figura 5.2: Gráficas de resultados del ataque *cropping* en la base de datos Inria

- Se observa que el método base recupera toda la información perdida cuando el cropping afecta hasta el 24 % de la imagen; en daños mayores la recuperación del método base BS es menor al 70 %.

- En método BS_{Damage} recupera toda la información perdida cuando el cropping afecta hasta el 23 % de la imagen; en daños mayores, la recuperación de este método obtiene en promedio porcentajes de recuperación cercanos al 100 %. Comparado con el método base, la caída en los porcentajes de recuperación es más suave.

- El comportamiento del método de BS_{Robust} con un daño menor de 22 % logra recuperar el 100 % de la imagen afectada pero a partir del 22 % de daño éste comienza a obtener recuperaciones menores al 100 %, bajando de manera abrupta al llegar al 25 % de daño.

Resultados aplicando el ataque de *Cropping* circular

En las figuras 5.3 y 5.4 se muestran las gráficas de la simulación del ataque de *cropping* circular en las bases de datos Pasadena House e Inria. Las gráficas no muestran una severidad del ataque menor a 5 ni mayor a 7, por recuperar el 100 % y no recuperar de manera perfecta, respectivamente.

El eje x de las figuras 5.3 y 5.4 corresponde a la severidad del ataque *cropping* circular, este se mide en % de columnas y filas cortadas en las cuatro direcciones, ejemplo de este ataque se muestra en la subsección 2.3.2.

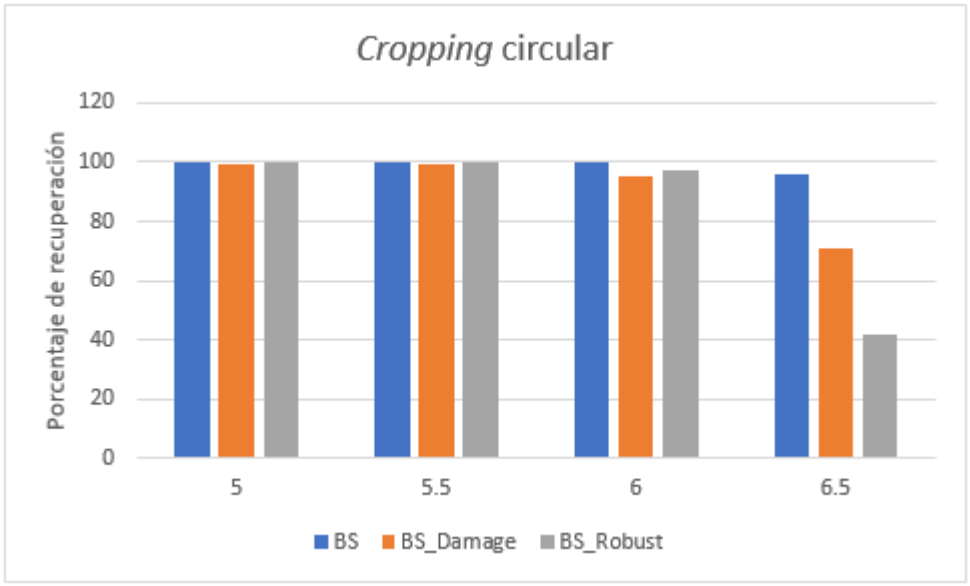

Figura 5.3: Gráficas de resultados del ataque *cropping* circular en la base de datos Pasadena House

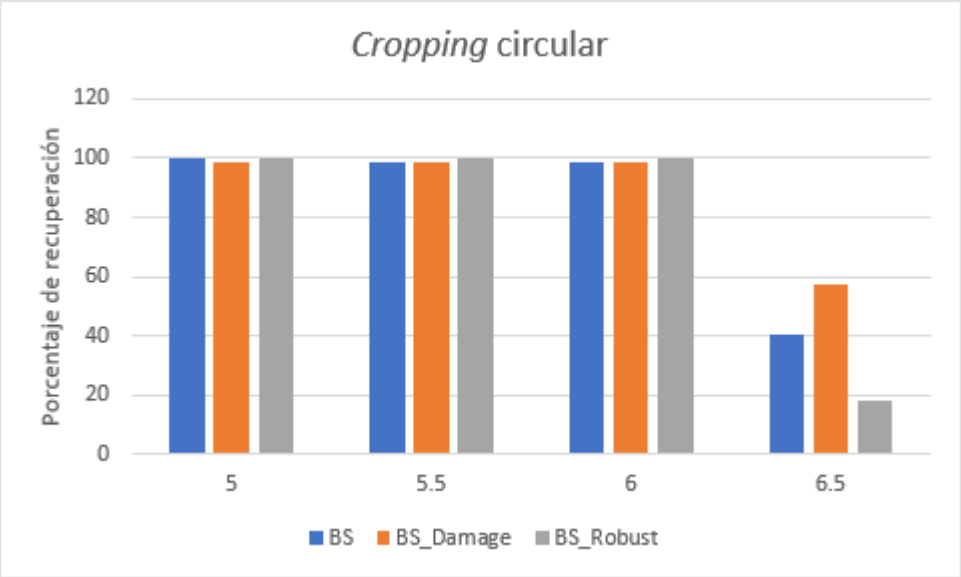

Figura 5.4: Gráficas de resultados del ataque *cropping* circular en la base de datos Inria

- Se observa que cuando el método base BS es afectado por el ataque de cropping circular y éste tiene una severidad 5 o menor, los porcentajes de recuperación se mantienen en 100 %. A partir de esta severidad el porcentaje de recuperación en algunas imágenes es $\approx 99.9\,\%$, llegando a severidad 6.5 la recuperación cae abruptamente al $20 - 40\,\%$.

- Cuando el método BS_{Damage} es afectado por el ataque de *cropping* circular y éste tiene una severidad 5.5 o mayor el porcentaje de recuperación se mantiene en $\approx 99\,\%$, al tener una severidad 6, cae su porcentaje de recuperación a $\approx 90 - 95\,\%$ y cuando el daño llega al 6.5 se severidad el porcentaje de recuperación desciende $\approx 50 - 70\,\%$.

- El comportamiento del método BS_{Robust} es similar al método base, con la distinción de que este método al tener una severidad 6 comienza a decender el porcentaje de recuperación hasta llegar 20 %.

Resultados aplicando el ataque de *Tampering*

Las figuras 5.5 y 5.6 muestran las gráficas de los resultados de simulación del ataque de *tampering* circular en las bases de datos Pasadena House e Inria. Las gráficas no muestran severidades del ataque menores a 21 ni mayores a 27, ya que los métodos recuperan el 100 % de los pixeles o no recuperan de manera perfecta, respectivamente.

El eje x de las figuras 5.5 y 5.6 corresponde a la severidad del ataque *tampering*, este se mide en porcentaje de píxeles sustituidos por valores aleatorios. La sustitución de pixeles se realiza en el centro de la imagen y en forma de cuadrado, ejemplo de este ataque se muestra en la subsección 2.3.2.

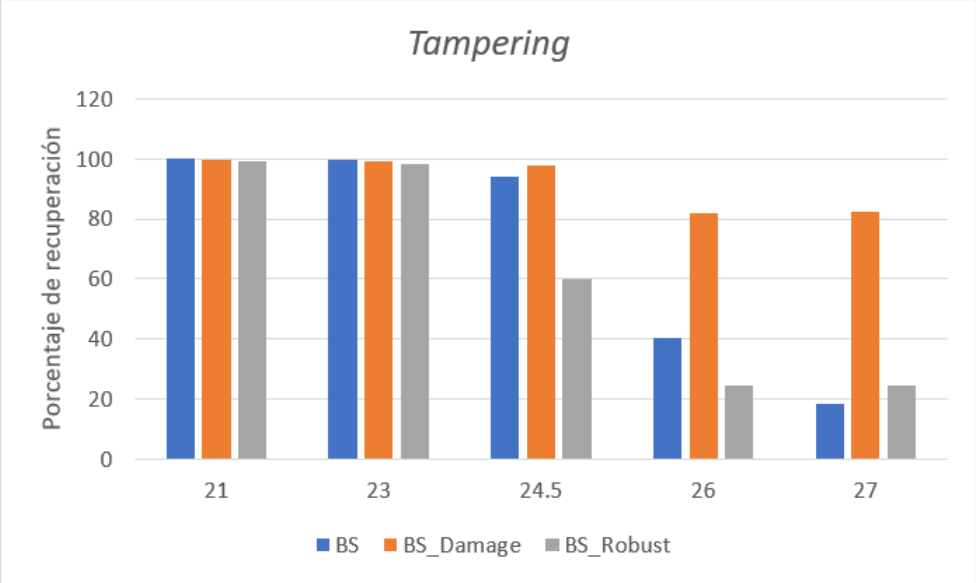

Figura 5.5: Gráficas de resultados del ataque *tampering* en la base de datos Pasadena House

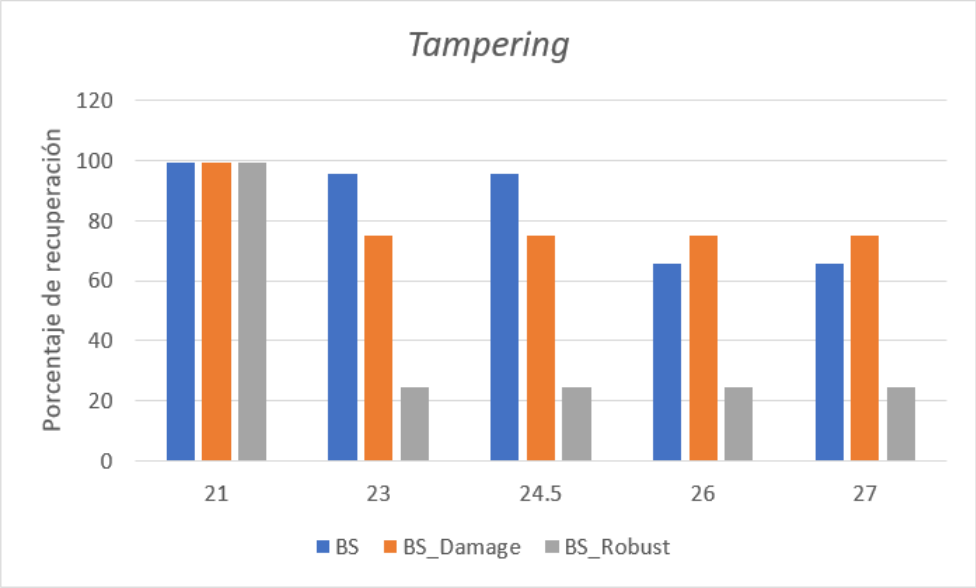

Figura 5.6: Gráficas de resultados del ataque *tampering* en la base de datos Inria

- En el método base BS, cuando es aplicado el ataque de *tampering* con severidad 23 % o menos, obtiene un porcentaje de recuperación del 100 % pero al llegar a 23 % la recuperación comienza a descender.

- En el método BS_{Damage}, cuando es aplicado el ataque de *tampering* con severidad menor a 21 %, mantiene el 100 % de recuperación. A partir de ahí los resultados obtenidos son cercanos al 100 % y al llegar a una severidad de 23 % cae su porcentaje de recuperación a $\approx 70 - 95$ %.

- El comportamiento del método BS_{Robust}, cuando es aplicado el ataque de *tamepering* con severidad 21 % o mayor el porcentaje de recuperación es cercano a 100 % y al llegar al 23 % éste baja de manera abrupta hasta llegar a $\approx 30 - 60$ %.

Resultados aplicando el ataque de Modificación LSB's

En las figuras 5.7 y 5.8 se muestran las gráficas de la simulación del ataque de Modificación LSB's en las bases de datos Pasadena House e Inria. Las gráficas no muestran resultados con ataques con una severidad del ataque mayor a 2 bits modificados, ya que en ningún método se recupera información a partir de ese valor. El ataque de modificación LSB's consiste en cambiar los n LBS's de cada pixel perteneciente a la imagen, el eje x corresponde al número de LSB's modificados.

Se observa que el método BS_{Robust}, barra color gris, es el único método capaz de soportar el ataque de modificación de LSB's, los otros métodos no logran autenticar el contenido con la modificación de los LSB's y por ende no existe la recuperación de contenido. No se observó diferencia en los resultados con respecto al tamaño de las imágenes.

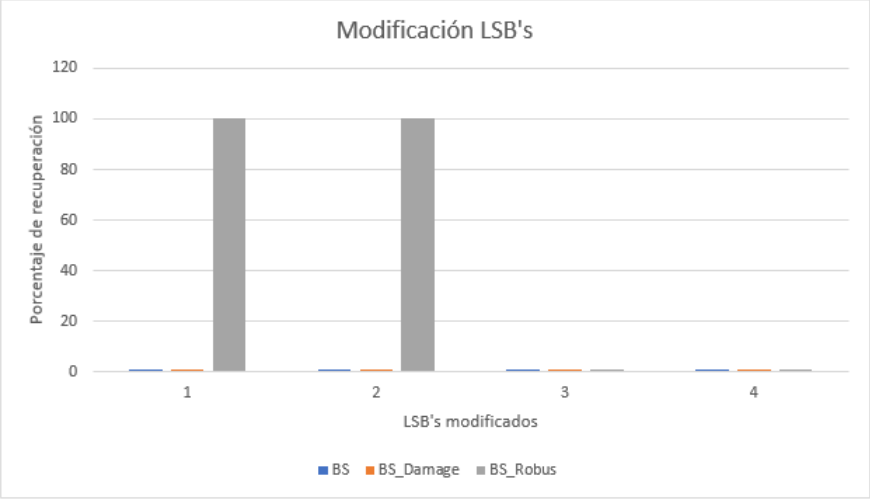

Figura 5.7: Gráficas de resultados del ataque modificación LSB's en la base de datos Pasadena House

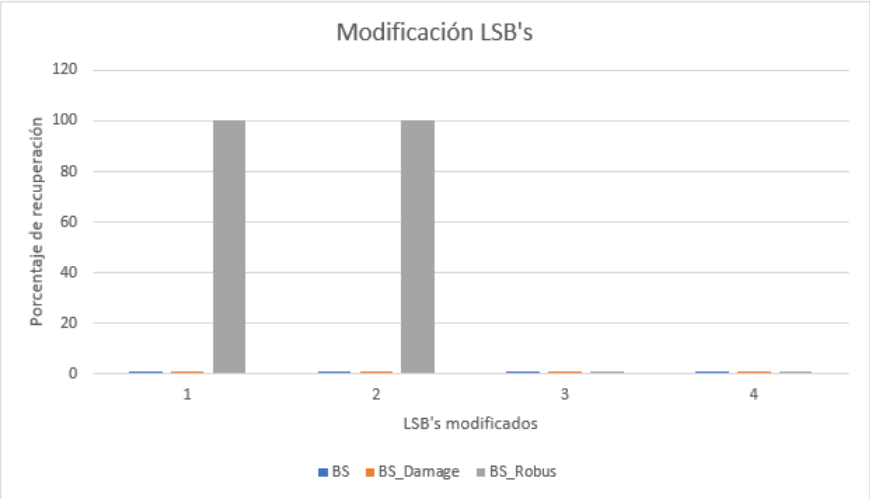

Figura 5.8: Gráficas de resultados del ataque modificación LSB's en la base de datos Inria

5.1.2. Del dominio de la frecuencia al dominio espacial

Al observar los resultados de los métodos propuestos BS_{Damage} y BS_{Robust}, se consideró realizar un experimento para aplicar las modificaciones BS_{Damage} y BS_{Robust} en el dominio espacial, esto es, proteger los pixeles en el dominio espacial en lugar de los coeficientes en el dominio de la frecuencia, como originalmente se ha propuesto. Los resultados presentados a continuación son resultados previos ya que se utilizó solo una base de datos y por lo tanto no es posible hacer conclusiones finales, pero son un buen acercamiento a cambio del dominio de la frecuencia al dominio espacial.

Al aplicar el método BS_{Robust} en el dominio espacial, se observó que el costo computacional era igual al método base; se aumentó la cantidad de ataques soportados, siendo capaz de soportar la sustitución de LSBs; la calidad de la imagen marcada es de 37.9 dB, siendo la misma que el método base;

Al aplicar el método BS_{Damage} en el dominio espacial, se observó que el costo computacional tiene una reducción entre el 5 % y el 10 % comparado con el método

base, la calidad de la imagen marcada es la misma que el método base y se aumentó a 25.5 % el daño soportado en el ataque de *cropping*.

Se utilizó la base de datos USC-SIPI con un total de 145 imágenes para realizar dicho experimento. En la figura 5.9, 5.10 y 5.11 se observan las gráficas de los resultados. Debido a que la base de datos contenía imágenes que no cumplen con las características de imágenes naturales, éstas no se utilizaron para el experimento, dejando un total de 108 imágenes.

En la tabla 5.2 se muestran los ataques utilizados en la base de datos USC-SIPI, con sus respectiva severidad. Los ataques son los mismos que los utilizados en los experimentos anteriores.

Tabla 5.2: Ataques y distinta severidad utilizados en base de datos USC-SIPI.

Número	Tipo de ataque	Severidad
1	*Cropping*	22 %
2		23.5 %
3		24 %
4		25.5 %
5	*Cropping* Circular	5.5 %
6		6.5 %
7		7.5 %
8	Modificación LSB	1
9		2

En las figuras 5.9, 5.10 y 5.11, el eje x muestra el número de ataque aplicado al conjunto de imágenes, ver tabla 5.2, y el eje y muestra el porcentaje de recuperación. se observa que el comportamiento de los métodos en el dominio espacial es similar a los métodos en el dominio de la frecuencia. Teniendo la desventaja de tener un costo computacional alto, en las gráficas se observa que se tienen mejores resultados al

proteger imágenes de mayor tamaño, por ejemplo en las imágenes de tamaño 256×256, los métodos BS_{Damage} y BS_{Robust} muestran resultados con menor porcentaje de recuperación en comparación con los resultados al usar imágenes de 1024×1024.

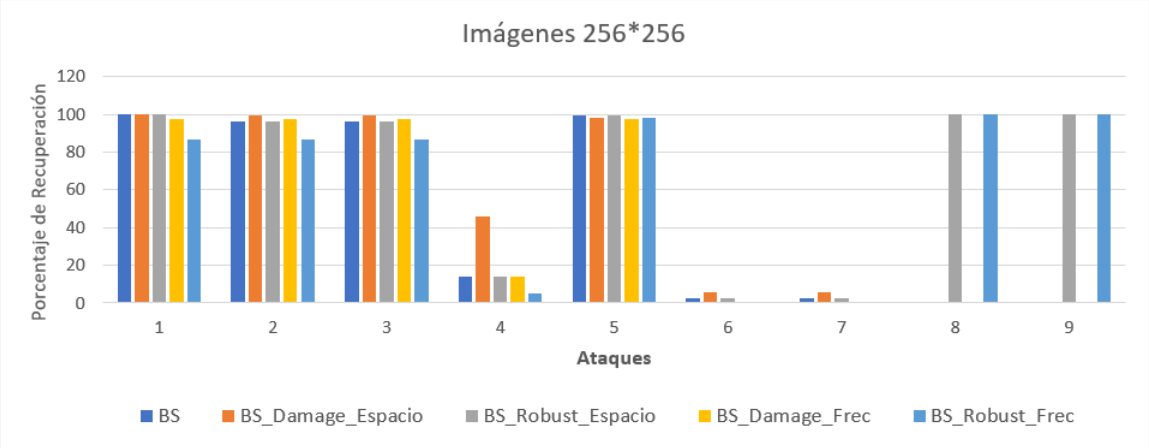

Figura 5.9: Gráficas de resultados utilizando base de datos SIPI con imágenes de 256×256

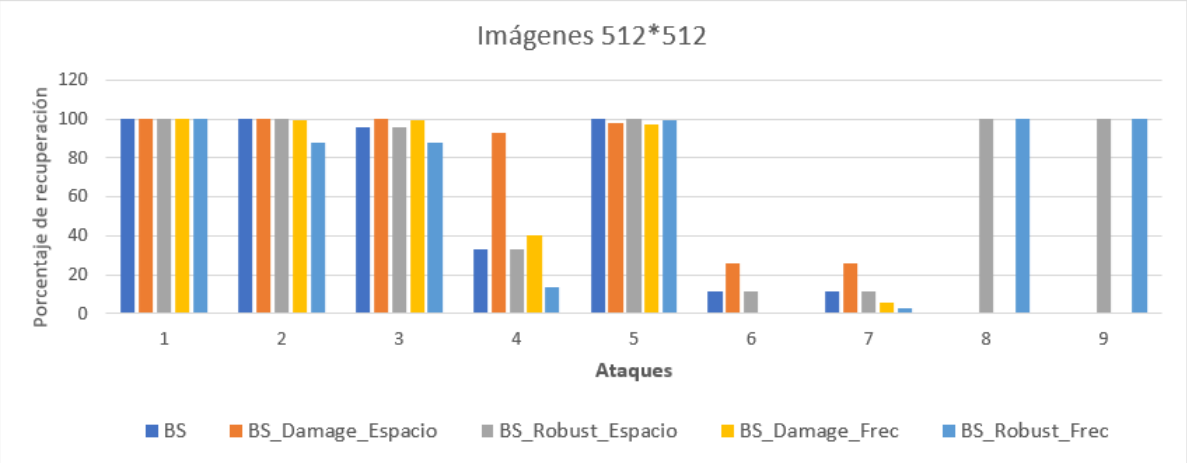

Figura 5.10: Gráficas de resultados utilizando base de datos SIPI con imágenes de 512×512

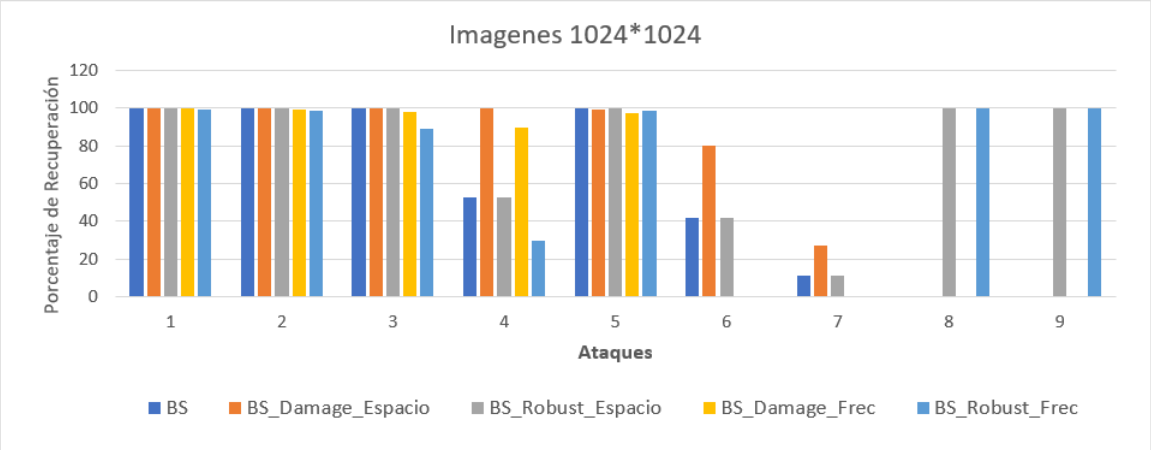

Figura 5.11: Gráficas de resultados utilizando base de datos SIPI con imágenes de 1024×1024

5.1.3. Discusión de resultados

En resumen, se tienen 2 diferentes métodos de auto-recuperación de imágenes.

1. Método 1 BS_{Damage}, este método tiene la ventaja de tener un costo computacional del $25\,\%$ en comparación con el método base; la cantidad de ataques soportados es el mismo pero la severidad de daño en los ataques de *cropping* y *tampering* es menor al método base $\approx 2\,\% - 4\,\%$; la calidad de la imagen marcada en promedio es de 28-30 dB, siendo menor al método base.

2. Método 2 BS_{Robus}, este método tiene la ventaja de tener un costo computacional de $25\,\%$ en comparación con el método base y poder soportar un ataque extra, modificación de LSB's; el soporte de los ataques de *cropping* y *tampering* es menor en comparación con el método base $\approx 3\,\% - 4\,\%$.

En la tabla 5.3 se presentan los resultados finales comparándolos con los métodos de la literatura.

Tabla 5.3: Tabla resumen resultados

Autor	Dominio	PSNR imagen marcada	*Tampering*	*Cropping*	Modificación LSB's
Zhan	Tiempo	37.9dB	<24 %	No	No
Dongmei	Tiempo	37.9dB	<33 %	No	No
Bravo	Tiempo	37.9dB	<25 %	<25 %	No
BS_Damage	IWT	28-30 dB	<21 %	<23 %	No
BS_Robust	IWT	28-30 dB	<21 %	<22 %	<3 LSB's

Los métodos que protegen los coeficientes en frecuencia tienen la desventaja de tener una calidad de imagen marcada relativamente baja, en promedio 28-30 dB. Los métodos propuestos en la frecuencia trabajan mejor cuando las imágenes tienen dimensiones mayores a 1024×1024 pixeles, esto por la diferencia observada entre las bases de datos de Inria y Pasadena House, siendo esta ultima la de mayor tamaño y la que obtuvo mejores resultados; la desventaja de un mayor tamaño es el costo computacional ya que este aumenta con la cantidad de pixeles presentes en la imagen a proteger.

La desventaja de los métodos que utilizan el dominio espacial como medio de inserción es su costo computacional, esto es debido que al proteger los pixeles es mucho mayor la cantidad de información a recuperar en comparación de los métodos que utilizan la frecuencia como medio de inserción.

El usuario final de los métodos propuestos tiene la opción de elegir el mejor método según sean sus necesidades, si es necesario dejar la distorsión de la imagen en niveles bajos, 37 dB, es recomendable usar el método base; pero si la distorsión no es importante y en cambio lo es el costo computacional, es recomendable usar los métodos propuestos BS_{Damage} o BS_{Robus}, siendo una ventaja para éste ultimo el soporte de una ataque extra.

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusión

En la actualidad el problema de auto-recuperación perfecta de imágenes se presenta como un problema sin una solución definitiva, debido al compromiso que existe entre la severidad de los ataques y la cantidad de ataques soportados. Los esquemas de auto-recuperación perfecta presentados en la literatura muestran distintas soluciones en donde cada uno utiliza diversas herramientas y técnicas de recuperación de información.

La presente investigación proporciona dos esquemas propuestos, los cuales aportan soluciones al problema principal. Dichas propuestas presentan ventajas y desventajas, destacando como ventajas principales el ahorro en tiempo de procesamiento y el aumento en el soporte de ataques en uno; la desventaja principal es la calidad de la imagen marcada, teniendo en promedio 28-30 dB. Esto se debe al uso del dominio de la frecuencia como medio a proteger y a la modificación de la información de control utilizada.

Al proteger solo las frecuencias bajas de la IWT, se reduce la cantidad de

información a proteger, por consecuencia se reduce el tiempo de procesamiento; la distorsión de la imagen marcada es mayor comparado con lo presentado en la literatura, debido a que la modificación en los coeficientes provoca mayores cambios en el dominio espacial. Con esta estrategia se obtienen resultados aproximados del porcentaje de recuperación de los ataques de *cropping* y *tamepring* en comparación a los métodos de la literatura.

Al analizar los resultados de los distintos ataques, se observó que los métodos propuestos tienden a soportar una severidad menor que el método base en los ataques de *tampering* y *cropping*. Ésto se debe a la disminución de las dimensiones de la matriz a proteger, al disminuir el número de coeficientes a proteger se aumenta la probabilidad de encontrar subconjuntos que contengan más de 3 coeficientes dañados, esto provoca que el método de recuperación de información en algunos casos sea incapaz de identificar el valor correcto de los coeficientes, por lo que no se puede recuperar completamente la imagen.

Por esta razón la el porcentaje máximo de severidad soportada de pixeles dañados en los ataques de *cropping* y *tampering* es de 21 %, esto para asegurar que en cualquier imagen natural sea posible realizar la recuperación sin errores. Tomando en cuenta el tamaño de la imagen, entre más grande sea ésta, la posibilidad de realizar una auto-recuperación perfecta exitosa es mayor. Se entiende por imagen grande aquella con dimensiones superiores a 512×512.

6.2. Trabajo a futuro

Como trabajo a futuro se observaron una serie retos. Estos se centran en la mejora del costo computacional y la mejora de la calidad de la imagen marcada.

- Disminuir el costo computacional a través de la implementación en hardware

o en cómputo paralelo. Esto es posible gracias a la independencia de las operaciones observadas en la solución propuesta. La sección del método en donde tendría mayor relevancia es en la de recuperación de información, ya que esta sección es donde se realiza el mayor número de operaciones.

- Disminuir la distorsión provocada al insertar información en los coeficientes, eso se podría lograr protegiendo las matrices de coeficientes HL, LH, HH. Al protegerlas no sería necesario eliminarlas y la distorsión disminuiría. Otro camino para proteger todos los coeficientes de la IWT es insertar la información de control en el dominio del espacio.

- Aumentar la severidad de los ataques, esto se podría lograr protegiendo la matriz LL en el dominio del espacio e insertar junto con esto información extra de la imagen, con esto se podría aumentar la severidad de los ataques soportados o incluso ataques distintos a los soportados actualmente.

Apéndice A

Algoritmo base

El método de Bravo Solorio [Bravo-Solorio et al., 2012], se tomó como base para el desarrollo del trabajo de investigación. Las ventajas de este método con respecto a los demás son: la cantidad de información de control que utiliza son 2 bpp y 1 bpp para realizar la autenticación; soportando un 25 % de daño con los tipos de ataques *tampering* y *cropping*.

Este método se puede dividir en 3 fases, inserción, extracción y recuperación de información. La fase de inserción se ilustra en la figura A.1, la fase de extracción de información se ilustra en la figura A.2 y la fase de recuperación de información se ilustra en la figura A.3.

A.1. Inserción

Se consideran imágenes de 8 bits de profundidad de color, la imagen de entrada Im tiene dimensiones n_1 y n_2, donde n_1 y n_2 son múltiplos de 8, siendo un total de N pixeles, $N = n_1 * n_2$. Cada pixel se representa con 8 bits $b1, b2, \cdots, b8$ donde b1 es el LSB y b8 es el MSB.

Usando una llave secreta $k1$ se permuta de manera pseudoaletoria los pixeles que forman a la imagen, para dicha permutación se utiliza el algoritmo Key Scheduling Algorithm (KSA), el cual es utilizado en el cifrador RC4 [Rivest, 1987].

Ya permutados los pixeles se forman subconjuntos de $m = 16$ pixeles, formando un total de $ns = N/m$ subconjuntos, donde $x_{i1}, x_{i2}, \cdots x_{im}$ denota los pixeles dentro del i-ésimo subconjunto. Se extraen los bits $b4, \cdots, b7$ de cada pixel por subconjunto, y son introducidos en una la función Hash, para obtener los bits de referencia r_i utilizando la siguiente ecuación:

$$r_i = H(\hat{x}_{i1}, \cdots, \hat{x}_{im}) \tag{A.1}$$

donde $H()$ es función Hash, usando el algoritmo SHA-2 y $\hat{x}_{ij} \in [0, 15]$ es dado por:

$$\hat{x}_{ij} = \lfloor x_{ij}/8 \rfloor mod 16 \tag{A.2}$$

el valor r_i se puede representar en una secuencia se bits $r_{i1}, \cdots, r_{im}$ llamados bits de referencia, los cuales son insertados en el $b3$ de cada pixel perteneciente al i-ésimo subconjunto, para evitar cambios en otros bits del pixel se eliminan los 3 LSB de cada pixel antes de hacer la inserción, se usa la siguiente ecuación:

$$x^w{}_{ij} = (\lfloor x_{ij}/8 * 8) + (r_{ij} * 4) \tag{A.3}$$

Una vez insertados los bits de referencia se permuta la imagen una vez mas, ahora usando la llave $k2$ con el método KSA. Se toman los 5 MSB de cada pixel para introducirlos a una función Hash y obtener $m = 16$ bits de referencia.

$$r_i = H(\breve{x}_{i1}, \cdots, \breve{x}_{im}) \tag{A.4}$$

donde $\breve{x}_{ij} \in [0, 31]$ es dado por:

$$\breve{x}_{ij} = \lfloor x_{ij}/8 \rfloor \tag{A.5}$$

los bits de referencia resultantes se insertan en el $b2$ de cada pixel perteneciente al i-ésimo subconjunto.

$$x^w{}_{ij} = x^w{}_{ij} + (rij * 2) \tag{A.6}$$

Para obtener el bit de autenticación de cada pixel, se parte de la imagen con los bits de referencia insertados en su posición original, se forman bloques de 8*8 pixeles, en total se tiene $n_b = N/64$ bloques. Cada bloque tendrá un código Cod distinto de 64 bits de longitud, el cual es formado de la siguiente manera:

$$Cod = I||n_1||n_2||p$$

donde $I =$ Indice de la imagen, un identificador por imagen; n_1 y n_2 son las dimensiones de la imagen original y $p =$ es el indice de bloque, denota el p-ésimo bloque. Obsérvese que Cod contiene un prefijo común, $(I||n1||n2)$, la longitud γ de éste puede variar dependiendo de los bits necesarios para la representación del numero de bloques p. El prefijo tiene la función de identificar los bloques que han sido dañados y los que no.

Para obtener los bits de autenticación se extraen los 7MSB de cada pixel por bloque, estos se introducen a una función Hash y se obtienen 64 $bits_{hash}$, los bits de Cod y los $bits_{hash}$ son operados por una función XOR:

$$a_{p,j} = w_{p,j} \bigoplus h_{p,j} \tag{A.7}$$

donde $p = 1, 2, \cdots, n_b$, $j = 1, 2, \cdots, 64$, $a_{p,j}$ son los bits de autenticación, $w_{p,j}$ son los bits de Cod y $h_{p.j}$ son los $bits_{hash}$.

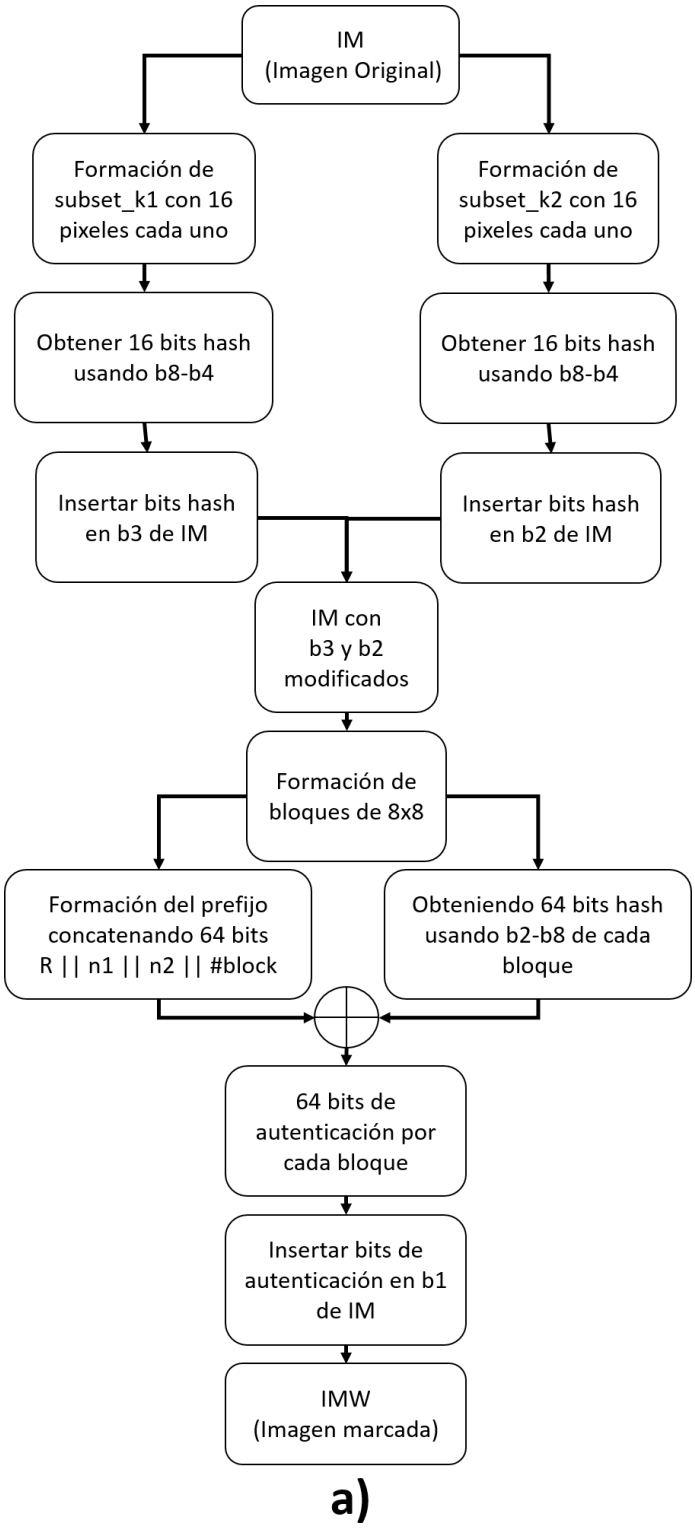

Figura A.1: Diagrama de la fase inserción de información del método propuesto por [Bravo-Solorio et al., 2012].

A.2. Auntenticación

Para la autenticación de información se presenta el esquema A.2. La imagen marcada es enviada a través de Internet por lo que ésta puede o ser atacada. La imagen marcada es divida en bloques de 8x8, por bloque se extraen los bits de autenticación, que están en el LSB de cada pixel. Se calculan los $bits_{hash}$ utilizando los 7MSB de cada pixel y se introducen en una función XOR:

$$w_{p,j} = a_{p,j} \bigoplus h_{p,j} \tag{A.8}$$

Con el XOR se obtienen los bits de Cod, a partir de Cod se obtienen el prefijo de longitud γ, se agrupan los $prefijos$ iguales en un conjunto A, si $|A| > TL$, donde $TL = 20$ se considera que la imagen contiene información insertada. Debido al ataque de *cropping* se tienen 64 posibles maneras de formar un bloque, desplazando el pixel de inicio en $i = [0, 1, \cdots, 8]$ y $j = [0, 1, \cdots, 8]$, donde $i = filas$ y $j = columnas$.

Al tener el prefijo correcto se procede a extraer las dimensiones originales de la imagen, si el p-ésimo bloque tiene el mismo prefijo que el seleccionado se considera un bloque reservado o bloque sin daño, en caso contrario el bloque se considera un bloque dañado y todos los pixeles pertenecientes al bloque también. Al identificar los bloques dañados y no dañados se considera que la imagen esta autenticada.

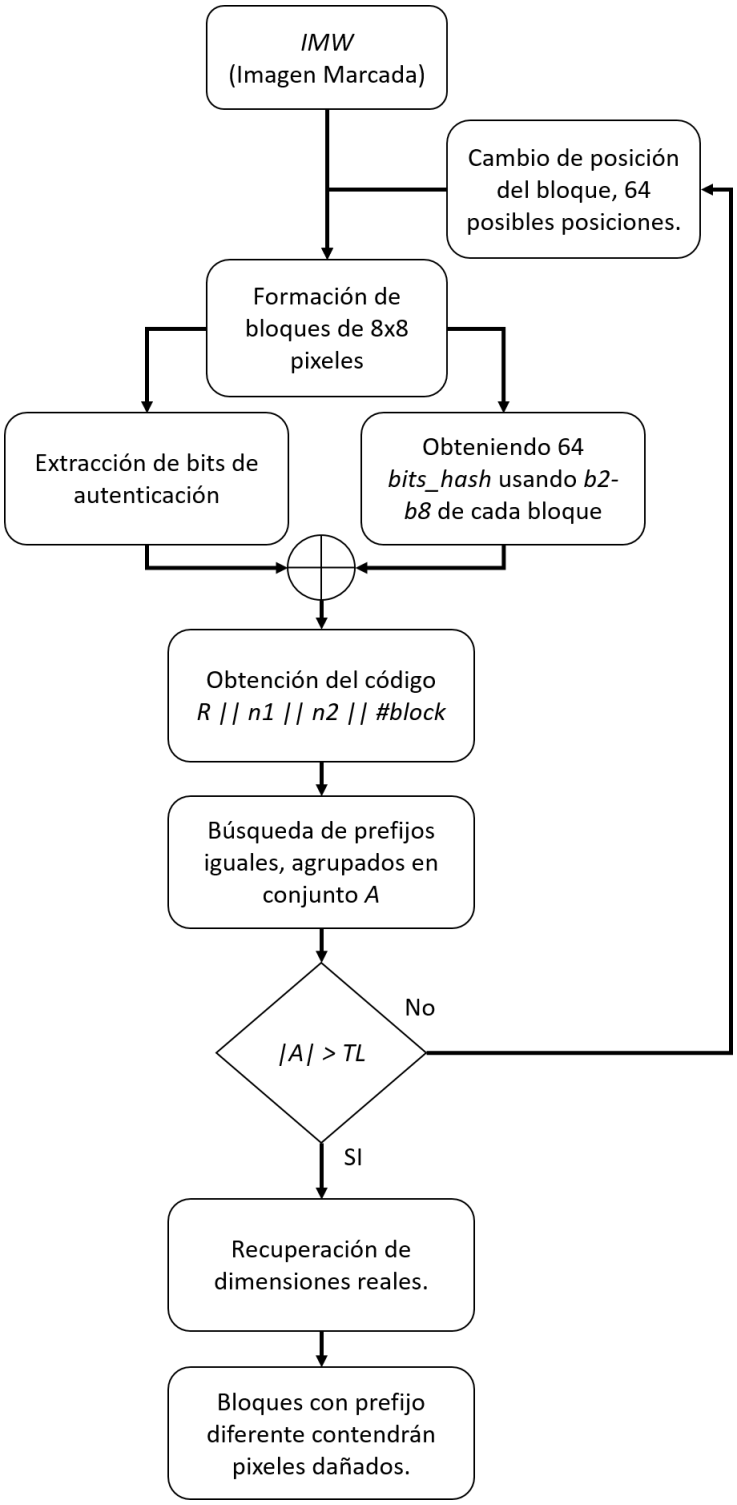

Figura A.2: Diagrama de autenticación de información del método propuesto por [Bravo-Solorio et al., 2012].

A.3. Recuperación de información

El proceso de recuperación se puede observar en la figura A.3 y es el siguiente: se permuta la matriz LL utilizando la llave $k1$, se forman los subconjuntos de tamaño $m = 16$. De todos los subconjuntos formados existen algunos que no se pueden procesar, los que están formados solo con coeficientes reservados y los que contienen demasiados coeficientes dañados. Para la recuperación se procesan los subconjuntos que tienen como máximo 3 coeficientes dañados y como mínimo 1, esto para evitar una búsqueda en un espacio demasiado grande. Si el subconjunto cumple con estas condiciones, el subconjunto se convierte en un subconjunto tratable y se realiza una búsqueda exhaustiva para encontrar los posibles verdaderos valores.

Se extraen los bits de referencia del i-ésimo subconjunto, que está en el $b3$ de cada coeficiente utilizando la siguiente ecuación.

$$r'_{ij} = \lfloor y_{ij}/4 mod 2 \rfloor \tag{A.9}$$

Se calculan los bits de referencia a partir de los coeficientes pertenecientes al i-ésimo subconjunto. $H(\hat{y}_{i1}, \cdots, \hat{y}_{im})$ donde $\hat{y}_{ij} = (\lfloor y_{ij}/8 \lfloor mod 16)$. Si y_{ij} pertenece a un coeficiente reservado, éste se introduce en la función hash $H(\cdot)$, de lo contrario se realiza una serie de iteraciones para introducir los 16 posibles valores de los 4 bits desconocidos para cada coeficiente dañado. Al comparar los bits reservados extraídos y calculados, se podrá conocer aquellos valores que sean candidatos a ser el valor correcto. La comparación se realiza bit a bit exceptuando los bits pertenecientes a coeficientes dañados. Se observa que como máximo son $16^3 = 4096$ posibles combinaciones a introducir a la función Hash, ya que son 16 posibles valores en cada uno de los coeficientes dañados y como máximo son 3 errores por subconjunto.

Al terminar el proceso de búsqueda se obtiene un conjunto R de posibles valores

por coeficiente dañado, que contienen todos los valores que coincidieron en el proceso de comparar los bits de referencia calculados y extraídos. Los valores en el conjunto R son expandidos a 5 bits, esto servirá para la búsqueda en los subconjuntos usando la llave $k2$. Si el i-ésimo coeficiente tiene un conjunto $R = \{7, 1\}$, la expansión sería $R = \{7, 23, 1, 17\}$. Para reducir el número de posibles valores se utiliza la vecindad espacial de un coeficiente de distancia, obteniendo un máximo de 8 vecinos, tomando en cuenta el valor de los vecinos solo si estos son no dañados. La diferencia entre el valor del coeficiente y el promedio de los vecinos debe ser menor a 5 para poderse tomar en cuenta como posible valor, en caso contrario se rechaza el valor. Por ejemplo, si el valor promedio es 20, el conjunto quedaría como, $R = \{23, 17\}$, los valores con una mayor diferencia de 5 son eliminados del conjunto R, esto para reducir el número de búsquedas en la siguiente etapa.

Posteriormente, se permutan los datos de la matriz LL utilizando la llave $k2$, formando subconjuntos de $m = 16$ coeficientes. Se toman solo los subconjuntos con un máximo de 4096 posibles combinaciones, teniendo en cuenta que habrá coeficientes con pocos posibles valores en el conjunto R que fueron encontrados en la etapa anterior. Se extraen los bits de referencia usando la siguiente ecuación.

$$r'_{ij} = \lfloor y_{ij}/2 \rfloor mod 2 \tag{A.10}$$

Se calculan los bits de referencia a partir de los coeficientes pertenecientes al i-ésimo subconjunto; se calculan los bits de referencia $H(\hat{y}_{i1}, \cdots, \hat{y}_{im})$ donde $\hat{y}_{ij} = \lfloor y_{ij}/8 \rfloor$. Si y_{ij} pertenece a un coeficiente reservado, éste se introduce en la función Hash $h(\cdot)$, si el conjunto R está vacío, se realiza una serie de iteraciones para introducir los 32 posibles valores de los 5 bits desconocidos para cada coeficiente dañado. En caso de que el conjunto R contenga valores, la iteración se realiza únicamente con estos.

Al finalizar la búsqueda, por coeficientes solo quedan un número limitado de

valores, por lo que si un coeficiente se asocia a un único valor, este se sustituye y es marcado como recuperado.

El proceso se repite en n-iteraciones de la permutación con $k1$ y después con $k2$, hasta que el número de coeficientes dañados sea 0. El método funciona debido a que la información está distribuída en los dos subconjuntos, por ejemplo, si en el primer subconjunto $k1$ un coeficiente X pertenece a un subconjunto con 6 coeficientes dañados, subconjunto no tratable, en la formación del segundo subconjunto $k2$, puede pertenecer a un subconjunto con solo 2 coeficientes dañados, éste se convierte en u subconjunto tratable, por lo cual el valor verdadero del coeficiente puede ser recuperado sin problemas.

El peor caso sucede cuando el i-ésimo coeficiente pertenece a un subconjunto en $k1$ con muchos coeficientes dañados y también en $k2$ pertenece a otro subconjunto con muchos coeficientes dañados. En estos casos los valores verdaderos no se pueden recuperar ya que el espacio de búsqueda es demasiado grande y no se podrá identificar cuál de ellos es el correcto.

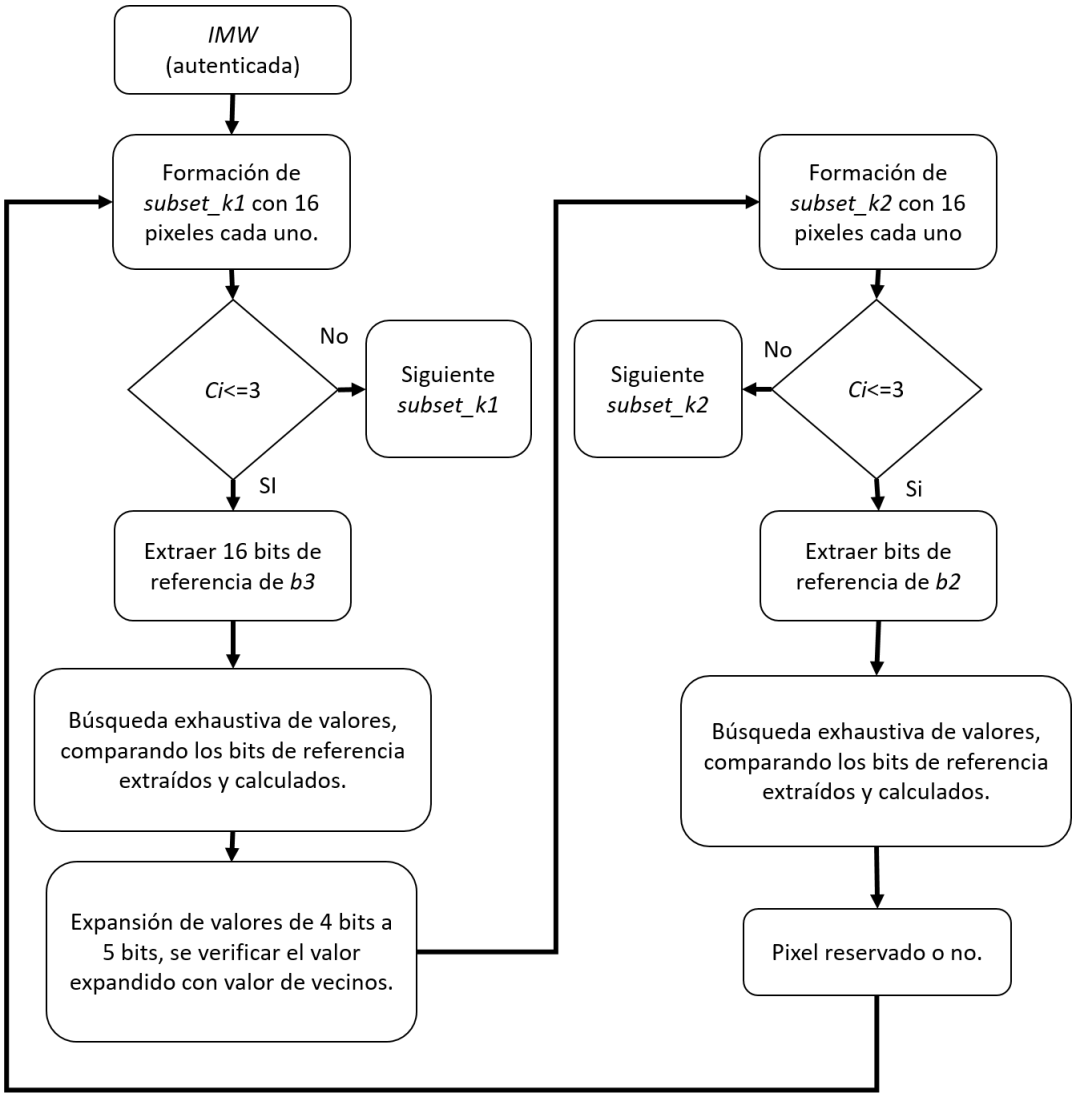

Figura A.3: Diagrama de recuperación de información del método propuesto por [Bravo-Solorio et al., 2012].

Bibliografía

[Amandí and Campo, 2006] Amandí, A. and Campo, M. (2006). *Introducción a la Transformada Wavelet*. Departamento de Señales y Sistemas. Universidad de Navarra.

[Barni and Bartolini, 2004] Barni, M. and Bartolini, F. (2004). *Watermarking systems engineering: enabling digital assets security and other applications*. CRC Press.

[Bravo-Solorio et al., 2012] Bravo-Solorio, S., Li, C.-T., and Nandi, A. K. (2012). Watermarking method with exact self-propagating restoration capabilities. In *Information Forensics and Security (WIFS), 2012 IEEE International Workshop on*, pages 217–222. IEEE.

[Calderbank et al., 1998] Calderbank, A., Daubechies, I., Sweldens, W., and Yeo, B.-L. (1998). Wavelet transforms that map integers to integers. *Applied and computational harmonic analysis*, pages 332–369.

[Chang and Tai, 2013] Chang, Y. and Tai, W. (2013). A block-based watermarking scheme for image tamper detection and self-recovery. *Opto-Electronics Review*, pages 182–190.

[Cox et al., 2007] Cox, I., Miller, M., Bloom, J., Fridrich, J., and Kalker, T. (2007). *Digital watermarking and steganography*. Morgan Kaufmann Publishers In.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Inria person dataset. `http://pascal.inrialpes.fr/data/human`. Accessed: 2017-07-17.

[Diego, 2008] Diego, R. I. (2008). *Análisis wavelet aplicado a la medida de armónicos, interarmónicos y subarmónicos en redes de distribución de energía eléctrica*. PhD thesis.

[Fridrich and Goljan, 1999] Fridrich, J. and Goljan, M. (1999). Images with self-correcting capabilities. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, pages 792–796. IEEE.

[Haar, 1910] Haar, A. (1910). On the theory of orthogonal function systems. *Mathematische Annalen, Springer*, pages 331–371.

[He et al., 2012] He, H., Chen, F., Tai, H.-M., Kalker, T., and Zhang, J. (2012). Performance analysis of a block-neighborhood-based self-recovery fragile watermarking scheme. In *IEEE Transactions on Information Forensics and Security*, pages 185–196. IEEE.

[He et al., 2009] He, H.-J., Zhang, J.-S., and Tai, H.-M. (2009). Self-recovery fragile watermarking using block-neighborhood tampering characterization. In *International Workshop on Information Hiding*, pages 132–145. Springer.

[Helle and Perona, 2000] Helle, C. and Perona, P. (2000). Dataset pasadena houses 2000. `http://www.vision.caltech.edu/html-files/archive.html`. Accessed: 2017-07-17.

[Hernández, 2013] Hernández, M. A. (2013). Esquema robusto de marca de agua digital reversible en imágenes. Master's thesis, Instituto Nacional Astrofisica Optica y Electronica.

[Herrigel et al., 2001] Herrigel, A., Voloshynovskiy, S. V., and Rytsar, Y. B. (2001). Watermark template attack. In *Photonics West 2001-Electronic Imaging*, pages 394–405. International Society for Optics and Photonics.

[Hore and Ziou, 2010] Hore, A. and Ziou, D. (2010). Image quality metrics: Psnr vs. ssim. In *Pattern recognition (icpr), 2010 20th international conference on*, pages 2366–2369. IEEE.

[Hung and Chang, 2007] Hung, K. L. and Chang, C.-C. (2007). Recoverable tamper proofing technique for image authentication using irregular sampling coding. In *Autonomic and Trusted Computing*, pages 333–343. Springer.

[Hyvearinen et al., 2009] Hyvearinen, A., Hurri, J., and Hoyer, P. O. (2009). *Natural Image Statistics, vol. 39*. Springer.

[Korus and Dziech, 2013] Korus, P. and Dziech, A. (2013). Efficient method for content reconstruction with self-embedding. In *IEEE Transactions on Image Processing*, pages 1134–1147. IEEE.

[Kutter and Petitcolas, 1999] Kutter, M. and Petitcolas, F. A. (1999). Fair benchmark for image watermarking systems. In *Electronic Imaging'99*, pages 226–239. International Society for Optics and Photonics.

[Leboeuf, 2016] Leboeuf, K. (2016). 2016 update: What happens in one internet minute? `http://www.excelacom.com/resources/blog/2016-update-what-happens-in-one-internet-minute`. Accessed: 2017-07-17.

[Lee and Won, 1999] Lee, J. and Won, C. S. (1999). Authentication and correction of digital watermarking images. In *Electronics Letters*, pages 886–887. IET.

[Lee et al., 2007] Lee, S., Yoo, C. D., and Kalker, T. (2007). Reversible image watermarking based on integer-to-integer wavelet transform. In *IEEE Transactions on Information Forensics and Security*, pages 321–330. IEEE.

[Lee and Lin, 2008] Lee, T.-Y. and Lin, S. D. (2008). Dual watermark for image tamper detection and recovery. pages 3497–3506. Elsevier.

[Mallat, 1989] Mallat, S. G. (1989). A theory for multiresolution signal decomposition: the wavelet representation. In *IEEE transactions on pattern analysis and machine intelligence*, pages 674–693. IEEE.

[Navas et al., 2008] Navas, K., Aravind, M., and Sasikumar, M. (2008). A novel quality measure for information hiding in images. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–5. IEEE.

[Nematollahi et al., 2016] Nematollahi, M. A., Vorakulpipat, C., and Rosales, H. G. (2016). *Digital Watermarking*. Springer.

[Niu et al., 2015] Niu, D., Wang, H., Cheng, M., and Zhou, L. (2015). Self-embedding watermarking scheme based on mds codes. In *International Workshop on Digital Watermarking*, pages 250–258. Springer.

[Noriega et al., 2011] Noriega, J. A. M., Kurkoski, B. M., Miyatake, M. N., and Meana, H. P. (2011). Image authentication and recovery using bch error-correcting codes. *International Journal of Computers*, pages 26–33.

[Pereira et al., 2001] Pereira, S., Voloshynovskiy, S., Madueno, M., Marchand-Maillet, S., and Pun, T. (2001). Second generation benchmarking and application oriented evaluation. In *International Workshop on Information Hiding*, pages 340–353. Springer.

[Petitcolas, 2000] Petitcolas, F. A. (2000). Watermarking schemes evaluation. In *IEEE signal processing magazine*, pages 58–64. IEEE.

[Petitcolas et al., 1998] Petitcolas, F. A., Anderson, R. J., and Kuhn, M. G. (1998). Attacks on copyright marking systems. In *International workshop on information hiding*, pages 218–238. Springer.

[Phadikar et al., 2012] Phadikar, A., Maity, S. P., and Mandal, M. (2012). Novel wavelet-based qim data hiding technique for tamper detection and correction of

digital images. *Journal of Visual Communication and Image Representation*, pages 454–466.

[Poularikas, 2010] Poularikas, A. D. (2010). *Transforms and applications handbook*. CRC press.

[Rey and Dugelay, 2002] Rey, C. and Dugelay, J.-L. (2002). A survey of watermarking algorithms for image authentication. In *EURASIP Journal on Advances in Signal Processing*, page 218932. Springer.

[Rivest, 1987] Rivest, R. (1987). Rivest cipher 4 (rc4).

[Som et al., 2015] Som, S., Palit, S., Dey, K., Sarkar, D., Sarkar, J., and Sarkar, K. (2015). A dwt-based digital watermarking scheme for image tamper detection, localization, and restoration. pages 17–37.

[Sweldens et al., 1995] Sweldens, W. et al. (1995). The lifting scheme: A new philosophy in biorthogonal wavelet constructions. *Wavelet Applications in Signal and Image Processing*, 3:68–79.

[Thompson and Shure, 1995] Thompson, C. and Shure, L. (1995). *Image Processing Toolbox: For Use with MATLAB;[user's Guide]*. MathWorks.

[Wang et al., 2011] Wang, H., Ho, A. T., and Zhao, X. (2011). A novel fast self-restoration semi-fragile watermarking algorithm for image content authentication resistant to jpeg compression. In *Digital Forensics and Watermarking*, pages 72–85. Springer.

[Wang et al., 2013] Wang, X., Zhang, D., and Guo, X. (2013). A novel image recovery method based on discrete cosine transform and matched blocks. *Nonlinear Dynamics*, pages 1945–1954.

[Wang et al., 2004] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. In *IEEE transactions on image processing*, pages 600–612. IEEE.

[Weber, 1997] Weber, A. G. (1997). The usc-sipi image database version 5. *USC-SIPI Report*, pages 1–24.

[Zhang and Wang, 2008] Zhang, X. and Wang, S. (2008). Fragile watermarking with error-free restoration capability. In *Multimedia, IEEE Transactions on*, pages 1490–1499. IEEE.

[Zhang and Wang, 2009] Zhang, X. and Wang, S. (2009). Fragile watermarking scheme using a hierarchical mechanism. In *Signal processing*, pages 675–679. Elsevier.

[Zhang et al., 2010] Zhang, X., Wang, S., Qian, Z., and Feng, G. (2010). Reversible fragile watermarking for locating tampered blocks in jpeg images. In *Signal Processing*, pages 3026–3036. Elsevier.

[Zhang et al., 2011] Zhang, X., Wang, S., Qian, Z., and Feng, G. (2011). Reference sharing mechanism for watermark self-embedding. In *Image Processing, IEEE Transactions on*, pages 485–495. IEEE.

Integer Wavelet Transform (IWT) Peak Signal-to-Noise Ratio (PSNR) Mean Squared Error (MSE) Structural Similarity Index (SSIM) IWT Discrete Wavelet Transform (DWT) Discrete Stationary Wavelet Transform (DSWT) Discrete Cosine Transform (DCT) Discrete Fourier Transform (DFT) Least significant bit (LSB) Most significant bit (MSB) Secure Hash Algorithm (SHA) Maximum Distance Separable (MDS) Bits per pixel (BPP)