



INAOE

Algoritmo robusto de marcas de agua usando similitudes basado en la codificación fractal

por

Pedro Aarón Hernández Ávalos

Tesis sometida como requisito parcial para
obtener el grado de

**MAESTRO EN CIENCIAS EN EL ÁREA DE
CIENCIAS COMPUTACIONALES**

en el

**Instituto Nacional de Astrofísica, Óptica y
Electrónica**

Enero 2008

Tonantzintla, Puebla

Supervisada por:

Dra. Claudia Feregrino Uribe, INAOE

Dr. René Armando Cumplido Parra, INAOE

©INAOE 2008

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



Resumen

El crecimiento desmedido de Internet en los últimos años, junto con el gran incremento del rendimiento de las computadoras ha facilitado el intercambio de información multimedia, específicamente imágenes. La necesidad de proteger dichas imágenes en contra de la piratería se ha vuelto un tema de vital importancia. Las marcas de agua digitales en imágenes se han utilizado como un medio de protección de derechos de autor, sin embargo éstas pueden servir para diferentes aplicaciones.

En esta investigación se desarrolla un algoritmo de marcas de agua digitales en imágenes de ocho bits en escala de grises utilizando selección de puntos característicos, regiones de búsqueda local e inserciones de similitudes. El algoritmo se crea utilizando el esquema basado en la codificación fractal de imágenes. Cinco diferentes enfoques fueron comparados con el nuevo algoritmo.

La contribución principal de esta investigación es que se logró disminuir la distorsión generada por la incrustación de la marca en la imagen portadora, al mismo tiempo se alcanzó una mayor robustez que el esquema basado en la codificación fractal, logrando así un nuevo algoritmo resistente a compresión JPEG y filtrado pasa bajas.

Abstract

The explosive growth of the internet in the last years along with the great increase in computers performance has facilitated the multimedia information exchange, images particularly. The necessity to protect such images against piracy has become an extremely important issue. Digital watermarking in images has been used as a way of copyright protection, although this can be used for different applications.

In this research an algorithm of digital watermarking for 8-bit gray scale images is developed. This algorithm utilizes a selection of interest points, local searching regions and similarities embedding, and it is created by a scheme based on fractal codification of images. Five different approaches have been compared with this new algorithm.

The principal contribution of this work is a decrease in the distortion generated by the mark embedding in the carrier image, besides a better robustness is achieved compared with the reference scheme based on fractal code. This new algorithm achieves resistance to JPEG compression and low pass filter.

Agradecimientos

Al pueblo de México, que mediante el CONACYT se me otorgó la beca número 201778, con la cual me fue posible estudiar este posgrado.

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

A mis asesores, la Dra. Claudia Feregrino Uribe y el Dr. René Armando Cumplido Parra por guiarme durante el camino de esta tesis, por alentarme, motivarme y por verter en mí parte de su sabiduría. De igual manera al Dr. Saúl Pomares por apoyarme durante momentos difíciles y enseñarme que la motivación es una fuerte compañera.

A mis padres por apoyarme en todo momento, por los sacrificios que hicieron y porque puedo contar con ustedes siempre. A mis hermanos por que son mis mejores amigos y fuente de inspiración. A mi familia por su apoyo sincero e incondicional.

A mis amigos por su estimada compañía y por haberme brindado su amistad.

*Por darme la vida,
Por amarme sin medida,
Por ser mis maestros de toda la vida,
Por enseñarme cosas más importantes que no se aprenden en la escuela,
Porque al escribir estas simples y sinceras palabras, que no expresan el inmenso
amor que tengo por ustedes, un par de lágrimas rodaron por mis mejillas...
Porque los amo, les dedico esta tesis.*

A mis padres, Pedro Hernández Hernández y Magaly Ávalos Reyna

A mis hermanos, Gibrán y Uziel, porque son mi modelo a seguir

*Y para ti mi vida, por haberme acompañado, apoyado y porque te amo con
alevosía, te dedico esta tesis.*

Para ti Jen.

Índice general

Índice general	VII
Lista de Figuras	x
Lista de Tablas	XIV
1 Introducción	1
1.1. Un poco de historia	1
1.2. La necesidad de ocultar información	4
1.3. Motivación	6
1.4. Objetivo General	7
1.5. Objetivos Secundarios	7
1.6. Estructura de la Tesis	8
2 Marco teórico	9
2.1. Esteganografía	9
2.1.1. Sistema esteganográfico	10
2.1.2. Aplicaciones	12
2.1.3. Requerimientos	13
2.1.4. Clasificación	15
2.1.5. Evaluación de los sistemas esteganográficos	20
2.1.5.1. Métricas para evaluar la imperceptibilidad	21
2.1.5.2. Métricas para evaluar la robustez	23

2.1.5.3.	Esquemas de ataques a las marcas de agua	23
2.2.	Codificación Fractal de imágenes	25
2.2.1.	Naturaleza fractal	25
2.2.2.	Compresión fractal de imágenes	27
2.2.3.	Sistemas de funciones iteradas	28
2.2.4.	Similitudes en las imágenes	30
2.2.5.	Codificación fractal de imágenes usando PIFS	32
2.2.5.1.	Compresión fractal utilizando PIFS	33
2.2.5.2.	Descompresión fractal utilizando PIFS	37
2.2.5.3.	Número de pasos en la compresión y descompresión fractal	40
3	Estado del arte	43
3.1.	Esquemas que utilizan la codificación fractal	43
3.2.	Esquemas inspirados por la codificación fractal	50
4	Esquema de marcas de agua propuesto	61
4.1.	Especificaciones	61
4.2.	Proceso de incrustación de la marca	63
4.3.	Proceso de extracción y detección de la marca	71
4.4.	Discusión del esquema propuesto	77
5	Implementación y resultados	79
5.1.	Implementación	79
5.2.	Pruebas realizadas	80
5.2.1.	Imágenes utilizadas	80
5.2.2.	Experimentos realizados	80
5.2.2.1.	Comportamiento del esquema	81
5.2.2.2.	Comportamiento del esquema ante ataques	85
5.2.3.	Comparación con otros esquemas	88

5.3. Discusión de los resultados obtenidos	91
6 Conclusiones y trabajo futuro	93
6.1. Revisión de objetivos	93
6.2. Conclusiones	94
6.3. Trabajo futuro	96
Bibliografía	99

Lista de Figuras

1.1. Crecimiento del número de sitios web en Internet	5
1.2. Crecimiento del número de hosts en Internet	5
2.1. a) Proceso de incrustación b) proceso de extracción c) proceso de de- tección	11
2.2. Relación entre los requerimientos básicos de un algoritmo de marcas de agua	16
2.3. Clasificaciones de las marcas de agua	17
2.4. Fractales con forma de plantas.	26
2.5. Fractal <i>hoja de maple</i> generado por un sistema de cuatro funciones iteradas.	27
2.6. Máquina copiadora que hace tres versiones reducidas de la imagen de entrada.	29
2.7. Primeras tres copias generadas por la máquina copiadora de la figura 2.6.	29
2.8. Fractal generado por un sistema de cuatro funciones iteradas:(2.13) - (2.16).	31
2.9. Imagen original de Lenna	32
2.10. Partes autosimilares de Lenna	32
2.11. Imagen de Lenna dividida en bloques rango de 8×8 píxeles	34
2.12. Imagen dominio contraída en 31×31 bloques dominio de 8×8 píxeles	35

2.13. Imagen resultante tras llevar a cabo la codificación fractal. Resolución de 256×256 píxeles y bloques rango de 8×8 píxeles	39
3.1. Bloque rango Rb y sus regiones de búsqueda local RBL_A , RBL_B . La RBL_C es formada por la unión de las anteriores.	44
3.2. Rango de valores de los coeficientes de brillo del mínimo al máximo.	47
3.3. Región de búsqueda local de bloques dominio para un bloque rango Rb propuesto por Zhen.	49
3.4. División de una imagen en regiones rango y regiones dominio.	51
3.5. a) SNR contra tamaño de los bloques rango, b) Precisión contra tamaño de los bloques rango.	53
3.6. a) Imagen particionada. b) Bloques dominio detectados. c) Bloques dominio después de la cuantización d) Detección de bloques rango por medio de similitudes.	55
3.7. Resultados obtenidos por el algoritmo de Patrick Bas para JPEG.	57
3.8. Inserción de similitudes en una ventana local.	58
4.1. Diagrama del esquema propuesto.	62
4.2. Imagen I_p de puntos característicos detectados en Lenna usando el detector de Harris.	63
4.3. Región de Incrustación RI_1 con $N_B = 25$ y $n = 4$	64
4.4. Regiones de Incrustación RI_i encontradas con $i = 1, 2, \dots, 8$, $N_B = 25$ y $n = 4$	65
4.5. Región de Incrustación RI_1 particionada.	66
4.6. a)Región de incrustación RI_p de RI_1 , b)Partición de RI_p , c)Bloques dominio D' seleccionados (bloques claros) y d)Bloques dominio D después de la decimación (bloques oscuros).	67
4.7. Región de búsqueda local utilizada en el esquema propuesto.	68
4.8. Región de búsqueda local asociada al bloque dominio D_9 y su respectivo bloque rango R_9	69

4.9. a)Bloque dominio D_9 , b)Bloque rango R_9 y c)Bloque rango modificado \hat{R}_9 con $\delta = 1$. Error $RMSE=2.0156$	69
4.10. Bloques rango (bloques blancos) y bloques dominio (bloques negros) seleccionados en RI_1	70
4.11. a)Imagen de RI_1 marcada con 16 bits, $PSNR=55.11$ dB. b)Imagen marcada con 16 bits en 8 RI , $PSNR=60.41$ dB.	71
5.1. Imágenes utilizadas en los experimentos.	80
5.2. a) Bloques rango contra PSNR, b)Bloques rango contra precisión y c)Bloques rango contra regiones de incrustación.	82
5.3. a) Tamaño de la región de incrustación contra PSNR, b)Tamaño de la región de incrustación contra precisión y c)Tamaño de la región de incrustación contra número de regiones de incrustación.	83
5.4. a) Tamaño de la región de incrustación contra bloques dominio detectados y b) Acercamiento de a).	84
5.5. a) Bits incrustados contra PSNR y b)Bits incrustados contra precisión.	84
5.6. Esquema sin detector. a) Bits incrustados contra PSNR y b)Bits incrustados contra precisión.	85
5.7. a) Precisión del esquema propuesto ante la compresión JPEG con bloques 4×4 y b) bloques 8×8	86
5.8. Resultados ante el ataque de ruido gaussiano	86
5.9. Imágenes marcadas con 34 bits y la adición de ruido gaussiano.	87
5.10. Comparación del esquema propuesto con el esquema de Patrick Bas ante JPEG incrustando una marca de 34 bits.	88
5.11. Comparación del esquema propuesto con el esquema de Puate ante la compresión JPEG. Marca de 32 bits. a)Bloques de 4×4 píxeles y b)Bloques de 8×8 píxeles.	89

5.12. Comparación del esquema propuesto con el esquema de Li ante la compresión JPEG. Marca de 32 bits. a)Bloques de 4×4 píxeles y b)Bloques de 8×8 píxeles.	89
5.13. Comparación del esquema propuesto con el esquema de Kamal ante la compresión JPEG. Marca de 49 bits. a)Bloques de 4×4 píxeles, b)Bloques de 8×8 píxeles y c)Bloques de 16×16 píxeles.	90
6.1. Marcado de la región de incrustación.	97
6.2. Esquema de inserción adaptativo.	98

Lista de Tablas

2.1. Transformaciones isométricas efectuadas en los bloques dominio . . .	37
2.2. Iteraciones de la etapa de descompresión fractal.	40
3.1. Resultados obtenidos por el esquema de Puate usando bloques rango tamaño 4×4 y redundancia 50.	45
3.2. Resultados obtenidos por el esquema de Puate usando bloques rango tamaño 8×8 y redundancia 25.	45
3.3. Resultados obtenidos por el esquema de Li <i>et-al</i> usando bloques rango tamaño 4×4	47
3.4. Resultados obtenidos por el esquema de Li <i>et-al</i> usando bloques rango tamaño 8×8	47
3.5. Resultados obtenidos por el segundo esquema de Kamal para compresión JPEG.	54
3.6. Resultados obtenidos por el tercer esquema de Kamal para compresión JPEG.	54
5.1. Comparación de la distorsión generada por cada esquema contra el esquema propuesto.	91

Capítulo 1

Introducción

1.1. Un poco de historia

A lo largo de la historia han existido una multitud de métodos para ocultar información, de los cuales el más antiguo encontrado hasta hoy es mencionado por Kanh en [Kah67] y cuyo origen es el antiguo Egipto, 4000 años atrás, donde la sustitución de símbolos jeroglíficos fueron usados para grabar información en la tumba de un noble llamado *Khnumhoteb II*.

Siglos después Heródoto (el padre de la historia) cuenta como se evitó la invasión de Persia a Grecia mediante el ocultamiento de un mensaje en unas tablillas de cera. Otro método ingenioso fue el afeitarse el cabello de un mensajero y tatuarse un mensaje en la cabeza, posteriormente cuando el cabello crecía el mensaje se ocultaba hasta que el cabello fuera afeitado nuevamente, el inconveniente de tal método es el tiempo que tarda el mensaje en ser recibido [JJ98].

En el año 1500 D.C. un monje alemán llamado Johannes Trithemius publica su máxima obra llamada *Steganographia*, un libro religioso de oraciones que oculta mensajes de hechicería en él, por lo que fue ferozmente perseguido por la iglesia de esos tiempos [Kah67]. Este libro fue el parteaguas para el inicio del ocultamiento de información en textos, además en él se acuñó el término *esteganografía*, usado

hasta nuestros días.

La palabra esteganografía proviene de las raíces griegas "steganos" que significa "oculto" y "graphos" que significa "escritura", lo cual en conjunto significa "escritura oculta o encubierta".

En 1593 Giovanni Battista muestra en su libro *De occultis literarum notis seu artis animemi occulte alijs significadi, aut ab alijs significata expiscandi enodandique, libri III* un método esteganográfico que utiliza una máscara para seleccionar las letras de un texto.

En 1665 Gaspari Schotti publica *Schola steganographica*, un libro en el cual se muestran más métodos esteganográficos en textos [Sch65].

Como en el siglo XVII existía una gran represión eclesiástica y política, muchos escritos ofensivos se publicaban anónimamente para evitar castigos severos, sin embargo Bishop Francis Godwin corrió el riesgo al publicar dos obras ofensivas con su nombre codificado en las letras mayúsculas iniciales de cada capítulo, tal método esteganográfico fue descubierto después de la muerte de Godwin [Lea96]. Este es un ejemplo claro de un método antiguo de protección de derechos de autor.

Las tintas invisibles fueron comúnmente usadas antes de la segunda guerra mundial, hacían que una simple carta pudiera contener un mensaje diferente dentro de ella. Otros métodos esteganográficos fueron utilizados también, como por ejemplo, una versión modificada del método de Godwin, el cual por ejemplo puede ocultar un mensaje en la segunda letra de cada palabra de un texto. Se usaron métodos más elaborados como lo son el cambiar la plantilla del texto, así por ejemplo la distancia entre espacios, letras, y palabras pueden ocultar un mensaje [LMBO95].

La esteganografía es diferente de la *criptografía*, aunque sus bases etimológicas sean muy parecidas, "kripto" significa "ocultar", si bien es cierto las dos tienen como tarea común la seguridad de la información pero se diferencian en la forma en como ambas llevan a cabo tal labor.

La criptografía tiene como función el proteger datos contra el acceso no auto-

rizado a los mismos, esta tarea se lleva a cabo *cifrando* o alterando tales datos, de tal forma que queden ilegibles. Para recuperar la información original de los datos cifrados es necesario llevar a cabo un proceso denominado *descifrar* mediante el uso de una *llave* (número o clave secreta).

Un inconveniente de la criptografía es que un "*enemigo*" puede detectar, interceptar y tratar de extraer la información de tales datos cifrados, aplicando distintas técnicas descifradoras. Por su parte la esteganografía tiene como función el ocultar un mensaje dentro de otro mensaje *inofensivo*, de tal forma que ningún *enemigo* pueda detectar que hay un segundo mensaje secreto presente.

La esteganografía es entonces una técnica diferente, ya que añade información extra al mensaje inocuo, pero al mismo tiempo lo deja en condiciones de ser utilizado. De manera tradicional la esteganografía consiste en ocultar un mensaje importante dentro de otro que no lo es.

Es bien conocido que la criptografía tiene como objetivos esenciales el garantizar la confidencialidad, autenticidad, no repudio y la integridad de los datos [MvOV96]. Por su parte la esteganografía debe considerar la amenaza de la eliminación no autorizada de los datos ocultos (esteganografía robusta), para lo cual no existe solución criptográfica [CDF06].

A mediados del siglo pasado empezó la investigación de la esteganografía electrónica como forma de proteger contenidos musicales de la piratería. En las décadas siguientes y con la creación y el crecimiento acelerado de la red de redes: *Internet*, la necesidad de proteger contenidos ha aumentado y ha cambiado, por lo que se han creado nuevas aplicaciones de la esteganografía como forma de ocultamiento de información [CM02].

En el año de 1996 se organiza la primer conferencia académica en el área de ocultamiento de información: *First International Workshop on Information Hiding* [And96], con lo que inicia de manera formal el inminente crecimiento de la investigación en la esteganografía.

1.2. La necesidad de ocultar información

El crecimiento de las redes de comunicación ha permitido el intercambio masivo de información, de ellas Internet es la única que ha mantenido un incremento logarítmico en el número de sitios web en los últimos años como se muestra en la figura 1.1. Dicho incremento se debe al aumento de *hosts* que requieren cada vez más servicios, figura 1.2 [Zak06]. Además la creación y aumento de redes, protocolos y aplicaciones *punto a punto* (usados para compartir información) han hecho que exista un gran flujo de información entre los usuarios de tales aplicaciones, la información que se transfiere es en su mayoría información *multimedia* (videos, audio, imágenes y textos), en donde la mayor parte de ésta incurre en la violación de derechos de autor y de copia.

Un estudio realizado en junio del 2007 por la empresa *Ellacoya Data* acerca del tráfico de Internet [Bur07] mostró que el 46 % es de tráfico web (HTTP), el 37 % es de tráfico generado por los programas *P2P* (punto a punto), el 9 % por grupos de noticias, el 3 % de video *streaming*, el 2 % de videojuegos y el 1 % de voz sobre IP. Aunado a esto, del 46 % del tráfico HTTP, el 45 % corresponde a páginas web de descarga directa, el 36 % corresponde a video streaming y un 5 % corresponde a audio streaming (*youtube*¹ por si sólo corresponde al 20 % del tráfico HTTP, cerca del 10 % del tráfico global). Con lo que se muestra que más de la mitad del tráfico de Internet corresponde a contenidos multimedia.

La protección de tales contenidos multimedia es un tema de vital importancia para la industrias de la música, cine, televisión, libros, software, etc. Por lo que han incrementado las investigaciones en lo que se refiere al ocultamiento de información para la protección del contenido, por ejemplo en la forma de *marcas de agua* (ocultar mensajes de derechos de autor) y *huellas digitales* (ocultar números seriales o un conjunto de características que distingan un objeto de otro similar). La idea es que después tal información pueda ser usada en contra de los

¹sitio web que permite a los usuarios subir, ver y compartir clips de videos.

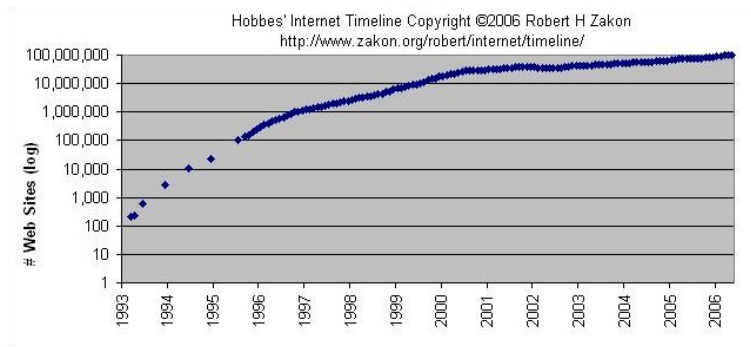


Figura 1.1: Crecimiento del número de sitios web en Internet

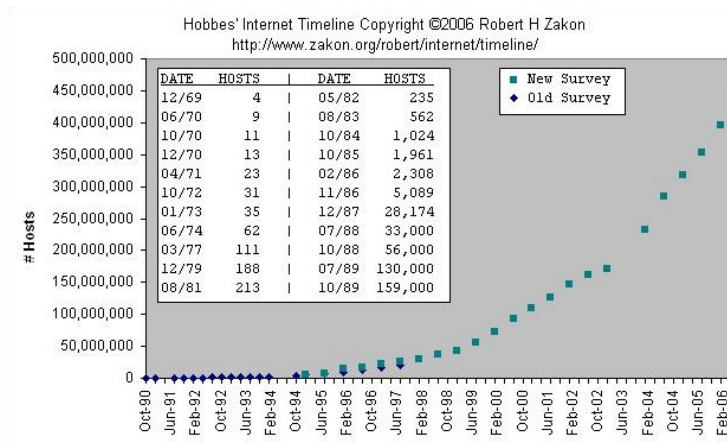


Figura 1.2: Crecimiento del número de hosts en Internet

infractores.

En el ámbito militar es donde más se ha utilizado la esteganografía. En la segunda guerra mundial mucha información fue filtrada, por ambos bandos, de forma oculta y cifrada para evitar que fuese hallada y decodificada.

En los últimos años la esteganografía ha ganado gran interés, debido en parte a la sospecha de que esta tecnología pudiera estar siendo utilizada por los terroristas para comunicarse los planes de próximos ataques. Aunque tales afirmaciones nunca han podido ser plenamente confirmadas, este supuesto ha sido un tema ampliamente discutido en la comunidad de las tecnologías de la información. Todo esto ha ayudado a que aún más gente conozca la esteganografía y su uso para ocultar información.

Pero hay muchas más aplicaciones que aparte de la esteganografía han incrementado el interés de las comunidades de negocios y académicas, como lo son las comunicaciones anónimas, canales cubiertos en sistemas de cómputo, protección del diseños de arquitecturas de hardware, detección de información oculta, etc.

1.3. Motivación

En la actualidad, la necesidad de proteger la autoría de contenidos digitales ha tomado gran fuerza, puesto que cada vez se comparten más contenidos multimedia a través de Internet. Para ello, se han creado muchos métodos que ocultan información en tales contenidos. Dicha información puede tomar distintas formas dependiendo de las características del método de inserción, del medio portador de los datos y del tipo de información a ocultar, que van desde el ocultar información del paciente en imágenes médicas, esconder números de serie en imágenes espaciales, hasta la utilización de tintas invisibles (que tienen información genética) para firmar documentos de gran importancia.

Existe una variada gama de técnicas esteganográficas de las cuales cada una tiene sus propias características. Específicamente en imágenes digitales se pueden encontrar distintas formas para ocultar información dentro de ellas, unas más populares que otras. Por ejemplo, la mayoría oculta información directamente en los píxeles variando la intensidad de color, otras más incrustan información usando los procesos internos (*transformadas*) de la *compresión* de imágenes. Llamándose así, técnicas esteganográficas en el dominio espacial y en el dominio de la transformada respectivamente.

Para cada uno de los principales métodos de compresión de imágenes que hay, es seguro que exista una técnica que oculte información usando sus tareas internas. Por ejemplo, se puede esconder información utilizando los coeficientes de la transformada de las compresiones JPEG y JPEG 2000, la selección de la paleta de colores de la compresión GIF o la forma de seleccionar los bloques de la

imagen en la compresión fractal.

La compresión fractal de imágenes tuvo su origen en la segunda mitad de la década de los 80's, iniciando con una gran expectativa debido al buen factor de compresión que tiene, sin embargo debido a la complejidad computacional que presenta no fue muy aceptada por la comunidad científica. Pero no fue la capacidad de compresión lo que hizo popular a esta técnica, sino que fue una de las primeras aplicaciones prácticas de la teoría fractal. La compresión fractal de imágenes a su vez fue el parteaguas para la creación de varias técnicas esteganográficas, las cuales tienen como principal característica el uso de bloques de imágenes y las afinidades entre ellos.

Los métodos de marcas de agua usando técnicas provenientes de la teoría fractal tienen gran robustez, sin embargo hoy en día son muy poco utilizados, debido a que la mayoría de ellas tienen alto costo computacional y además tienden a distorsionar la imagen marcada. La eliminación de tales desventajas los hacen atractivos para su utilización en la protección de derechos de autor, puesto que se convertirían en métodos robustos que distorsionen poco a la imagen marcada.

1.4. Objetivo General

Crear un algoritmo de marcas de agua robusto para imágenes en escala de grises utilizando código fractal, similitudes y puntos característicos.

1.5. Objetivos Secundarios

- Desarrollar un algoritmo con capacidad de ocultamiento y robustez mayor al algoritmo del cual fue basado [BCD98a, BCD98b] y algoritmos que usan técnicas fractales para la ocultamiento de la marca.
- Reducir la distorsión generada por la incrustación de la marca sin que se vea afectada la robustez.

1.6. Estructura de la Tesis

Esta tesis está estructurada de la siguiente forma.

Capítulo 2. En este capítulo se describe el marco teórico de la tesis, se muestran los fundamentos utilizados a lo largo de esta investigación. Se presenta también una explicación teórica de las marcas de agua digitales y de la codificación fractal de imágenes.

Capítulo 3. Aquí se describe el estado del arte de los principales algoritmos de marcas de agua basados en codificaciones fractales. Así como también se describe el algoritmo base de la presente investigación.

Capítulo 4. Se desarrolla el algoritmo de marcas de agua propuesto. Se detallan los procesos internos que el algoritmo lleva a cabo: detección de puntos característicos, regiones de búsqueda local, incrustamiento, detección y extracción de la marca.

Capítulo 5. Se presentan la pruebas realizadas al esquema propuesto, así como también la comparación con otros esquemas de marcas de agua inspirados por la codificación fractal.

Capítulo 6. Se describen las conclusiones obtenidas y el trabajo a futuro.

Capítulo 2

Marco teórico

Los conceptos básicos de la esteganografía y de la compresión fractal de imágenes son tratados en el presente capítulo. Se abordan los principios fundamentales de los sistemas esteganográficos, así como la clasificación de los mismos y el posicionamiento del esquema propuesto en dicha clasificación. Además se explica el primer esquema de codificación fractal de imágenes. Todo esto es con el fin de dar los conceptos y teorías tratadas en los capítulos siguientes.

2.1. Esteganografía

La palabra esteganografía es poco usada en la actualidad para referirse a los métodos de ocultamiento de información, en su lugar se suelen utilizar palabras como *marcas de agua* (*watermarking*), *incrustación de datos* (*data embedding*) y *ocultamiento de información* (*information hiding*), de entre ellas, marcas de agua es más reconocida por la mayoría de las personas. Sin embargo, hay quienes tratan a la esteganografía y a las marcas de agua como diferentes [KP00, JJD01]. En esta investigación son tomadas de manera indiferente, al igual que los otros términos mencionados anteriormente, ya que son formas de llamar al hecho de ocultar información en un medio.

2.1.1. Sistema esteganográfico

La idea básica de ocultar un mensaje en un documento o imagen recae en el hecho de que el mensaje no sea detectado o reconocido por una tercera persona "enemiga" o "atacante", en otras palabras, es ocultar un mensaje importante en un medio inocuo para evitar levantar sospecha de tal ocultamiento.

De manera general, un *sistema esteganográfico* consta de tres etapas: el *proceso de incrustación o inserción*, el cual representa el esquema que oculta la información dentro de un contenido multimedia también llamado *medio portador o medio original*. La *información a ocultar o marca* puede ser un número de serie, información de derechos de autor, datos del contenido multimedia, una imagen monocromática u otro contenido multimedia. Después del proceso de inserción de la marca mediante el uso de algún algoritmo, el medio original quedará ligeramente modificado y recibe el nombre de *medio marcado o estegomedio*. La diferencia entre el medio original y el medio marcado debe ser nula o muy pequeña.

Seguido del proceso de incrustación, el medio marcado será enviado por Internet o por cualquier otro canal a un receptor. Siempre y cuando exista duda de los derechos de autor, de copia o cualquier otro evento relacionado con el medio marcado o medio a probar, la marca será decodificada para identificar y verificar tales datos insertados. El *proceso de decodificación* consta de dos pasos: uno conocido como *proceso de extracción* de la marca, el cual es por lo general inverso al proceso de incrustación de la marca. El otro es conocido como *proceso de detección* y se encarga de decidir la validez de la marca incrustada. Este último puede no ser parte del proceso de decodificación.

En la figura 2.1 se muestran los procesos llevados a cabo en un sistema esteganográfico. Por lo general, en el proceso de inserción mostrado en la figura 2.1a se tiene el medio original (X), por ejemplo una imagen, en la que el algoritmo de inserción le esconde una marca (W), el resultado de este proceso es un medio marcado (X'), por ejemplo una imagen marcada. En este proceso una *llave* (por

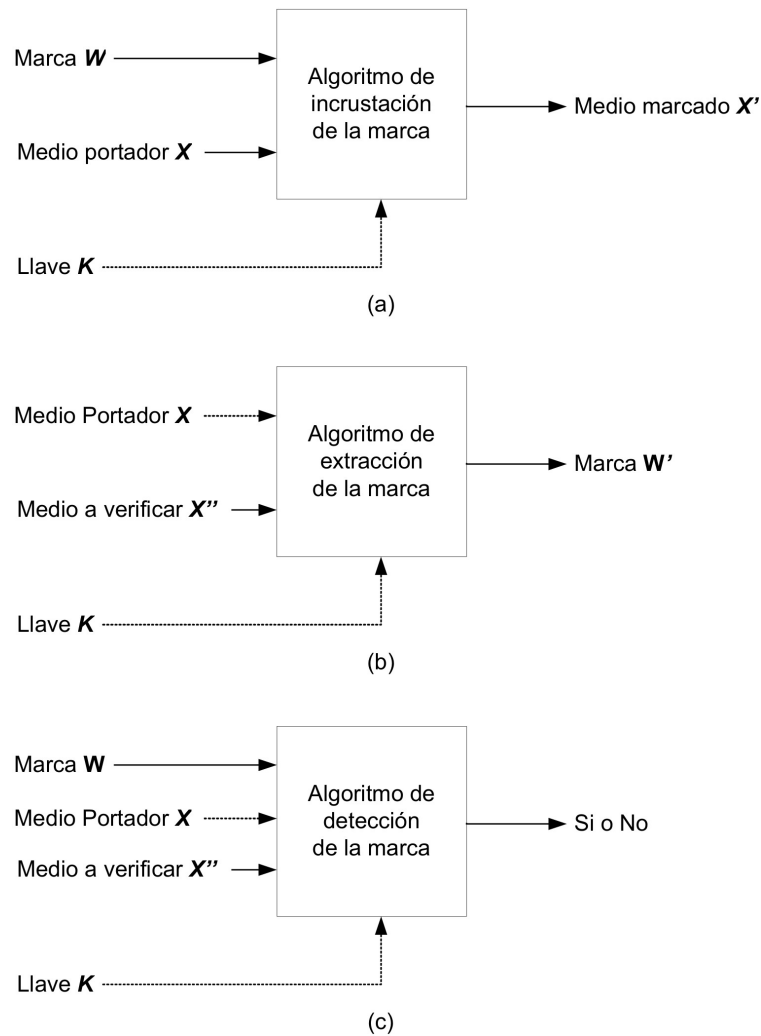


Figura 2.1: a) Proceso de incrustación b) proceso de extracción c) proceso de detección

ejemplo un número de secuencia) puede ser utilizada para producir una marca de agua más segura.

Las líneas punteadas en figura 2.1 indican que pueden ser utilizadas para un diseño particular. El proceso de extracción de la marca mostrado en la figura 2.1b es utilizado para obtener la marca ocultada en un medio, tal proceso puede utilizar la imagen portadora o la llave según sea necesario. Por último, el proceso de detección mostrado en la figura 2.1c es utilizado para verificar la validez de la marca extraída, es decir la salida de este proceso es una decisión booleana. Este

proceso utiliza por lo general a la marca W para verificar la identidad de la marca extraída.

Formalmente, el proceso de incrustación de la marca es una función que mapea las entradas X , W y/o K a la salida X' , esto es,

$$X' = I(X, W, [K]) \quad (2.1)$$

donde $I(x)$ denota al proceso de incrustación, y $[K]$ indica que la llave K puede o no ser incluida. De forma similar el proceso de extracción de la marca $E(x)$, puede ser denotado por

$$W' = E(X'', [X], [K]) \quad (2.2)$$

Y el proceso de detección $D(x)$ es

$$Si \ o \ No = D(X'', [X], W, [K]) \quad (2.3)$$

Recordando que $[x]$ indica que el elemento x dentro de los corchetes puede ser opcional.

2.1.2. Aplicaciones

En general, las marcas de agua digitales son utilizadas por aplicaciones que:

1. Contienen información de propiedad.
2. Verifican el contenido de objetos.

El propósito de los primeros tipos de aplicaciones es identificar la propiedad de un objeto, en ellos es común la utilización de algoritmos que insertan marcas de agua *perceptibles* dentro de un medio. Por ejemplo, muchas compañías incrustan el logo de la misma en una esquina de un video o de una imagen, con el fin de hacer constar que el medio es de su propiedad. Para prevenir que alguien pueda eliminar la marca insertada en el medio, los algoritmos de marcas de agua *robustos*

son utilizados. Por el contrario, el propósito del segundo tipo de aplicaciones es asegurar la integridad u originalidad del medio marcado, para ello los algoritmos de marcas de agua *frágiles* son utilizados, ya que su característica principal es que pueden detectar (idealmente) la alteración de un bit en el medio marcado.

Los algoritmos de marcas de agua *semi-frágiles* combinan las ventajas de los algoritmos de marcas de agua robustos y marcas de agua frágiles. Un algoritmo de marcas de agua semi-frágil puede soportar ataques¹ en cierta medida, lo que el algoritmo frágil no soportaría. Por lo que los algoritmos de marcas de agua semi-frágiles tienen las características de los algoritmos robustos y frágiles a la vez dentro de niveles de distorsión específicos.

De esta forma quedan definidos los diferentes usos de los algoritmos de marcas de agua. Cada aplicación posee sus requerimientos muy particulares que deben ser tomados en cuenta para la elección del algoritmo de marcas de agua a utilizar. Un algoritmo de marcas de agua universal que pueda soportar todos los ataques y al mismo tiempo satisfacer todos los requerimientos deseables (sección 2.1.3), no existe. Sin embargo el desarrollo de algoritmos de marcas de agua para aplicaciones específicas es un hecho.

2.1.3. Requerimientos

Como se ha mencionado, cada aplicación de marcas de agua tiene necesidades o requerimientos específicos, por lo que no hay un conjunto de requerimientos que sean utilizados por todas las aplicaciones, sin embargo existen tres requerimientos básicos tomados en cuenta por los algoritmos de marcas de agua digitales, éstos se listan a continuación.

1. TRANSPARENCIA PERCEPTUAL. También llamada *transparencia* o *imperceptibilidad*. En muchas ocasiones se requieren algoritmos de marcas de agua

¹Análisis, Manipulaciones o modificaciones realizadas al medio marcado con el fin de alterar, eliminar o encontrar la marca insertada.

que no afecten la calidad del medio marcado, esta capacidad es la transparencia perceptual. Un algoritmo de marcas de agua es verdaderamente imperceptible si no se puede distinguir a simple vista las diferencias entre el medio marcado y el medio original. Aunque esta capacidad es en origen subjetiva, existen métricas para evaluarla. Como ejemplo se tiene a la relación señal a ruido (PSNR), el error cuadrático medio (EMS), la correlación, el error absoluto máximo (MAE), etc. Nótese que para poder evaluar esta capacidad es necesario comparar al medio marcado con el medio original. Como normalmente se distribuye el medio marcado sin el medio portador es suficiente que las modificaciones en el medio marcado pasen desapercibidas para que el algoritmo de marcas de agua utilizado sea considerado como imperceptible.

2. **ROBUSTEZ.** Es la capacidad que tiene un algoritmo de marcas de agua para poder extraer la marca incrustada en el estegomedio después de que éste último haya sido atacado con el fin de extraerle o eliminarle la marca. Ejemplos de ataques en imágenes digitales son los siguientes: compresiones, transformaciones geométricas, filtrados, conversiones digital - analógico (D/A) y analógico - digital (A/D), modificaciones del histograma, etc. Un algoritmo de marcas de agua frágil en imágenes digitales no tiene robustez en contra de ataques, tales como una simple compresión o una mínima modificación del histograma, ya que la más mínima modificación hace que se pierda la marca incrustada. Sin embargo, esta posible debilidad hace que la aplicación de este tipo de algoritmos sea útil para asegurar la integridad del medio marcado. Por otra parte, los algoritmos robustos tienden a modificar el medio marcado. No existe algoritmo alguno que sea robusto a todos los ataques, o más específicamente, no puede soportar un solo ataque en su totalidad sin que el medio marcado, que fue atacado, se degrade perceptualmente o que sea convertido en un medio inutilizable.

3. **CAPACIDAD.** Se refiere a la cantidad de información que puede ser ocultada en un medio, en otras palabras es el tamaño máximo de la marca que se puede ocultar. La cantidad de información que se requiera ocultar en la imagen depende de la aplicación en donde se vaya a utilizar, por ejemplo para el control de copias con un bit es más que suficiente, pero si se trata de ocultar información de derechos de propiedad intelectual es necesario ocultar más información. Por ejemplo si se desea ocultar el ISBN o International Standard Book Numbering (10 dígitos) de un libro en una imagen del mismo, sería necesario ocultar sólo 34 bits. Por otro lado existen aplicaciones en donde la marca es una imagen completa, aquí son necesarios muchos más bits. Existe una regla general en cuanto a la cantidad de información que se oculta en un medio, entre más se oculte más se modifica el medio y menos robusto es el medio marcado.

Como es de esperarse, todos los requerimientos básicos están relacionados entre sí. Por ejemplo, un algoritmo de marcas de agua en imágenes con mucha capacidad tiende a distorsionar la imagen, por lo que se vuelve perceptible, a la vez que la robustez se ve afectada, ya que entre más información se guarde en la estegoimagen, más difícil es garantizar la extracción de la marca después de que haya sido atacada. Esta relación entre los requerimientos básicos se observa mejor en la figura 2.2 [Lan00].

2.1.4. Clasificación

Existen múltiples clasificaciones de las marcas de agua, las principales de ellas son mostradas en la figura 2.3, en ella se muestran clasificaciones de acuerdo al uso específico, percepción o visibilidad, robustez, tipo de medio portador, esquema de inserción y de extracción.

La clasificación de las marcas de agua de acuerdo a su uso específico [VDM02] es la siguiente.

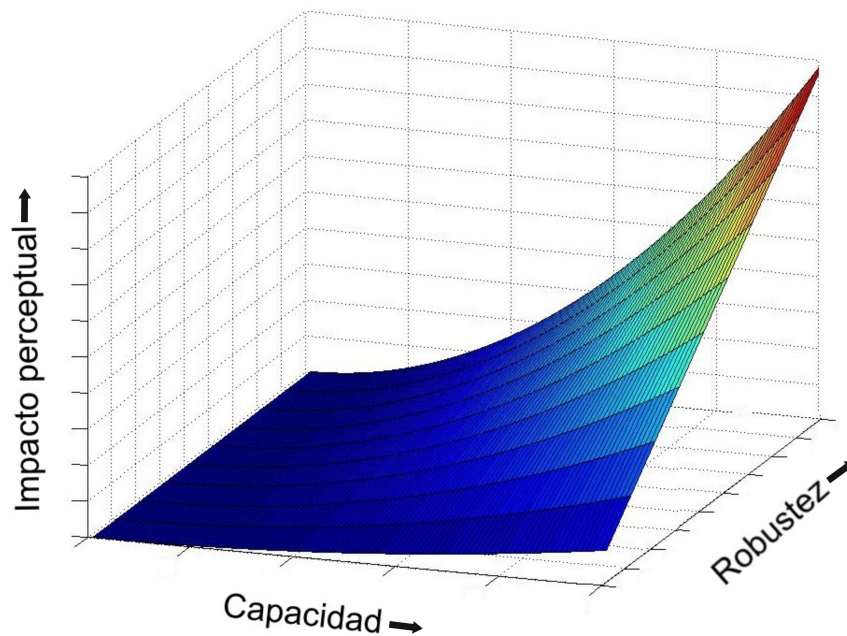


Figura 2.2: Relación entre los requerimientos básicos de un algoritmo de marcas de agua

1. OCULTAMIENTO DE INFORMACIÓN (*information hiding*): este tipo de marcas de agua se usa para evitar la detección del medio incrustado, esto es, se da un alto grado de importancia a la imperceptibilidad de la marca.
2. MARCAS DE AGUA ROBUSTAS: son utilizadas para evitar la extracción de la marca insertada, por lo que el requerimiento básico es la robustez, tales marcas de agua son utilizadas principalmente para esconder información de protección de derechos de autor y de derechos copia.
3. MARCAS DE AGUA FRÁGILES Y SEMI-FRÁGILES: este tipo de marcas de agua son utilizadas para evitar la falsificación del medio marcado. Como se ha mencionado anteriormente éstas son utilizadas para verificar la integridad y autenticación del medio marcado.

La imperceptibilidad de la marca insertada sirve para clasificar a los algoritmos de marcas de agua. Esta clasificación los divide en:

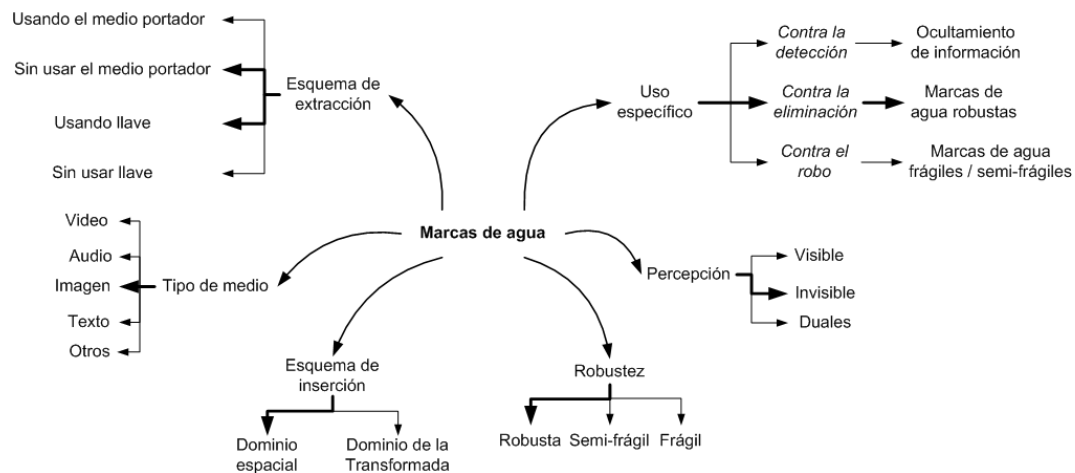


Figura 2.3: Clasificaciones de las marcas de agua

1. **MARCAS DE AGUA PERCEPTIBLES O VISIBLES:** Este tipo de marcas de agua es principalmente utilizado para incrustar patrones visibles en imágenes o videos. Su principal característica es que la marca insertada es visible pero no obstructiva, además de ser robustas. Cabe mencionar que no son muy populares en investigaciones académicas, aunque sus aplicaciones prácticas son consideradas fáciles de implementar.
2. **MARCAS DE AGUA IMPERCEPTIBLES O INVISIBLES:** En contraste a las marcas de agua visibles, este tipo de marcas de agua son perceptualmente invisibles. Muchos esquemas de este tipo ocultan información en imágenes o videos usando el conocimiento de cómo funciona el sistema visual humano o HVS (*human visual system*). Si se oculta una marca en audio, este tipo de marcas de agua son llamadas inaudibles.
3. **MARCAS DE AGUA DUALES:** Utilizan una marca de agua invisible como respaldo de una marca visible, siendo entonces una combinación de ambas.

La capacidad que tienen los algoritmos de marcas de agua para extraer la marca incrustada después de un ataque es conocida como robustez y sirve para clasificarlos. Tal clasificación es:

1. MARCAS DE AGUA ROBUSTAS: Tienen como principal característica el poder soportar modificaciones o distorsiones en el medio marcado, es decir, que se pueda rescatar la marca parcial o totalmente del estegomedio después de haber recibido distorsiones o modificaciones. Estas marcas son utilizadas principalmente para la protección de derechos de autor, protección de derechos de copia y como algoritmos de huellas digitales (*Fingerprinting*).
2. MARCAS DE AGUA FRÁGILES: Una marca de agua frágil es simplemente una marca que llega a ser indetectable después de la más mínima modificación del medio original, idealmente.
3. MARCAS DE AGUA SEMI-FRÁGILES: A diferencia de las marcas de agua frágiles, la marca incrustada en un medio llega a ser indetectable después de una cierta modificación, esto es, que es robusta hasta cierto nivel de distorsión. Este tipo de marcas de agua se usan para la autenticación del medio marcado.

El medio en donde es incrustada la marca define un tipo de clasificación de marcas de agua. La clasificación por tipo de medio portador es la siguiente:

1. MARCAS DE AGUA EN VIDEOS: Este tipo de marcas de agua esconden principalmente a la marca dentro de los *frames* (cuadros) del video.
2. MARCAS DE AGUA EN AUDIO: la marca es incrustada en amplitud, la fase o en el espectro de la señal del audio, por lo general la marca incrustada no es audible.
3. MARCAS DE AGUA EN IMÁGENES: Las imágenes son el medio más común para esconder información, la marca suele ocultarse dentro de los píxeles de la imagen o dentro del proceso de compresión de la misma.

4. MARCAS DE AGUA EN TEXTOS: La principal forma de esconder información dentro de un texto es mediante la modificación controlada del formato, espaciado y de las posiciones de letras en las palabras.
5. MARCAS DE AGUA EN OTROS MEDIOS: Las marcas de agua pueden ser ocultas en otros medios, por ejemplo en la organización interna de circuitos [LMSP99] y en billetes.

El esquema de inserción de la marca en el medio portador precisa la siguiente clasificación:

1. MARCAS DE AGUA EN EL DOMINIO ESPACIAL: Este tipo de marcas de agua añade la marca en la imagen portadora modificando los valores de los píxeles ligeramente. También conocido como *watermarking aditivo*.
2. MARCAS DE AGUA EN EL DOMINIO DE LA TRANSFORMADA: En los algoritmos de compresión de imágenes, audio y video, la realización de transformadas es una de las más importantes formas de construcción de bloques para el procesamiento del medio de entrada. Las marcas de agua en el dominio de la transformada ocultan información en los coeficientes generados al aplicar una transformada a un bloque del medio. Este esquema de marcas de agua es llamado *watermarking multiplicativo*.

La forma de extraer de la marca del estegomedio genera la siguiente clasificación:

1. EXTRACCIÓN SIN USAR EL MEDIO ORIGINAL: Este esquema de marcas de agua extrae la marca de agua sin utilizar el medio original. También son llamados algoritmos de *marcas de agua a ciegas, desatendidos o públicos*.
2. EXTRACCIÓN USANDO EL MEDIO ORIGINAL: A diferencia del tipo anterior, éste extrae la marca utilizando el medio portador. Llamados también algoritmos de *marcas de agua informados o privados*.

3. EXTRACCIÓN USANDO LLAVE: Este tipo de marcas de agua utiliza información adicional (llave) para la extracción de la marca.
4. EXTRACCIÓN SIN USAR LLAVE: En contraste con el esquema anterior, este tipo de marcas de agua no utiliza una llave para la extracción de la marca.

Cada esquema de marcas de agua cae en uno de los tipos de cada clasificación mostrada anteriormente. Particularmente, la clasificación del algoritmo realizado en esta investigación es por su uso específico un algoritmo de marcas de agua robusto, ya que es usado contra la eliminación de la marca. Por la percepción es un algoritmo invisible ya que la marca es imperceptible. Por su robustez es un algoritmo robusto. Por el tipo de medio es usado en imágenes. Por su esquema de inserción es usado en el dominio espacial y finalmente por su esquema de extracción es clasificado como un algoritmo que usa llave y que no usa el medio portador para la extracción. Cada una de las clasificaciones del algoritmo de esta tesis se muestran en la figura 2.3 con líneas remarcadas.

2.1.5. Evaluación de los sistemas esteganográficos

Una vez que un sistema de esteganográfico ha sido diseñado e implementado es importante evaluar su rendimiento objetivamente. Esta evaluación debe ser hecha comparando los resultados obtenidos contra los encontrados por sistemas de marcas de agua con el mismo propósito o con uno similar.

Para conocer la calidad de un algoritmo de marcas de agua es necesario evaluar los tres principales requerimientos de todo sistema esteganográfico: la capacidad, la robustez y la imperceptibilidad.

La imperceptibilidad de la marca puede ser evaluada principalmente de dos formas: de manera subjetiva usando la percepción de la vista humana o usando métricas de distorsión o de distancias. La robustez puede ser evaluada analizando el comportamiento del algoritmo de marcas de agua en contra de los principales ataques. La capacidad es evaluada midiendo la cantidad de información incrustada

en el estegomedio. Sin embargo, en sistemas robustos la capacidad no es una medida importante de comparación, ya que la característica de estos sistemas es que no necesitan ocultar demasiada información, por lo que no es de vital importancia evaluarla.

2.1.5.1. Métricas para evaluar la imperceptibilidad

La calidad y fidelidad de un medio marcado está en función de la perceptibilidad o visibilidad de la marca en el estegomedio. La fidelidad representa la diferencia del medio marcado con el medio original y la calidad representa la aceptabilidad del estegomedio. La mejor métrica para evaluar la calidad y fidelidad de un medio marcado son los sentidos humanos. Por ejemplo, para medios visuales es la vista; para auditivos el oído. Una métrica utilizada para evaluar la imperceptibilidad es mediante el uso de experimentos de selección, por ejemplo usando la prueba de *selección forzada de dos alternativas* (*T AFC* o *2 AFC* por sus siglas en inglés), esta prueba consiste en mostrarle a un conjunto de observadores dos opciones, en este caso son el medio original y el marcado, cada observador debe de seleccionar cual medio le parece con más calidad, después de la realización de varias pruebas con distintos medios se hace un análisis estadístico para poder determinar si el medio marcado tiene una calidad aceptable con respecto al medio original. Existen más métricas que utilizan la selección de alternativas [IC01], las cuales proveen medidas precisas de perceptibilidad, pero sin embargo, siguen siendo pruebas subjetivas, además de costosas, no fáciles de repetir y no pueden ser automatizadas. Por otra parte, existen métricas automatizadas para medir la distorsión causada por la incrustación de la marca en un medio, éstas son fáciles de aplicar y no son subjetivas. La distorsión causada por el proceso de ocultamiento de la marca puede ser definida como la diferencia o distancia entre el medio original y el medio marcado.

Las principales métricas objetivas para medir la distorsión entre dos medios, específicamente en imágenes [EF95, SRN01] (donde X es la imagen original de

tamaño M y X' la imagen marcada de tamaño N) son:

1. ERROR CUADRÁTICO MEDIO (MSE): Esta medida de distorsión es la más simple de calcular, se define como:

$$MSE = \frac{1}{M * N} \sum_{i=1}^M \sum_{j=1}^N (X(i, j) - X'(i, j))^2 \quad (2.4)$$

Para que la marca permanezca imperceptible el error MSE debe ser lo más pequeño posible.

2. ERROR RAÍZ CUADRÁTICO MEDIO (RMSE): Esta medida de distorsión es la raíz cuadrada del error cuadrático medio o MSE, se utiliza para darle mayor peso a los errores MSE pequeños. Al igual que el error MSE, la invisibilidad de la marca es mejor para los valores pequeños del error RMSE. Ésta se calcula de la siguiente manera:

$$RMSE = \sqrt{MSE} \quad (2.5)$$

3. RELACIÓN SEÑAL A RUIDO (SNR): Esta medida de distorsión es la más popular, se calcula de la siguiente forma:

$$SNR = 10 \log \left[\frac{\sum_{i=1}^M \sum_{j=1}^N (X(i, j))^2}{\sum_{i=1}^M \sum_{j=1}^N (X(i, j) - X'(i, j))^2} \right] dB \quad (2.6)$$

Entre más grandes sean los valores SNR menos imperceptible es la marca, la unidad de esta métrica está dada en decibeles.

4. RELACIÓN SEÑAL A RUIDO PICO (PSNR): Al igual que la SNR, los valores más grandes de PSNR indican una mejor imperceptibilidad de la marca. Se calcula de la siguiente manera:

$$PSNR = 10 \log_{10} \left(\frac{Max^2}{MSE} \right) = 20 \log_{10} \left(\frac{Max}{RMSE} \right) dB \quad (2.7)$$

donde Max es la magnitud máxima de un píxel, por ejemplo si se estuvieran comparando imágenes de 8 bits, Max toma el valor de 255. Los valores PSNR mayores de 30 dB son considerados aceptables.

5. ERROR ABSOLUTO MÁXIMO (MAE): Este error es el valor máximo de los valores absolutos de las diferencias de los dos medios.

$$MAE = \max(|X - X'|) \quad (2.8)$$

Al igual que el error MSE, entre más pequeño es el error MAE , mejor es la aproximación de los dos medios.

6. CORRELACIÓN CRUZADA(CORR): Esta métrica mide la similitud del medio original con el medio marcado, los valores que puede tomar van de 0 a 1. Estos valores indican el porcentaje de parecido entre los dos medios. Entre mayor sea el valor, la calidad del medio marcado será mejor. Tal medida se calcula de la siguiente manera:

$$Corr = \frac{\sum_{i=1}^M \sum_{j=1}^N X(i, j) X'(i, j)}{\sum_{i=1}^M \sum_{j=1}^N X(i, j)^2} \quad (2.9)$$

2.1.5.2. Métricas para evaluar la robustez

La robustez de un esquema de marcas de agua puede ser evaluada aplicando diferentes clases de distorsiones o modificaciones al medio marcado, tales ataques deben ser de interés o relevantes para el tipo de aplicación en donde se vaya a utilizar el esquema de marcas de agua. La robustez puede ser evaluada midiendo la probabilidad de extraer la marca después de que el estegomedio haya recibido un ataque, es decir se evalúa midiendo la precisión, la cual es la relación de la marca extraída con la marca originalmente incrustada en el estegomedio.

2.1.5.3. Esquemas de ataques a las marcas de agua

Los ataques tienen como tareas principales el eliminar la información oculta dentro del medio, evitar la correcta extracción de la marca aún y cuando ésta siga dentro del medio o simplemente analizar el medio para obtener la información oculta. Por lo general los ataques se realizan a propósito, pero cabe la posibilidad

de que éstos sean meramente accidentales. Por ejemplo, una imagen marcada con un algoritmo de marcas de agua semi-frágil puede ser comprimida mínimamente para reducir su tamaño(KB). Nótese que el objetivo no fue eliminar la información. La clasificación de los ataques se muestra a continuación.

ATAQUES PASIVOS: Esta clase de ataques no modifican ni distorsionan al estegomedio con el fin de eliminar la marca, simplemente realizan un análisis para identificar si contiene información oculta, de ser así algunos esquemas son capaces de determinar aproximadamente la cantidad de información ocultada en él. Ejemplos de este tipo de ataque son:

- **ATAQUES VISUALES:** Este tipo de ataques son realizados a través de la vista humana, es decir, el atacante observa el medio supuestamente marcado y determina si éste tiene información insertada tomando en cuenta los patrones visuales del medio. Un ejemplo de este ataque es el visualizar el bit menos significativo de una imagen, si dicha imagen oculta información que fue incrustada por el método *LSB*² (*least significant bit*) es muy probable de que exista una distorsión en el plano menos significativo, con lo que el atacante puede determinar si existe o no un mensaje oculto [WP00, Moe03].
- **ATAQUES ESTADÍSTICOS:** Este tipo de ataques analizan el medio marcado estadísticamente y determinan si existe información oculta en él. El método Chi-cuadrada es comúnmente utilizado para determinar si existe información y la cantidad de información ocultada. El inconveniente de este método es que se necesita el medio marcado y el medio original [WP00, Moe03].
- **ATAQUES DE COPIADO:** Este tipo de ataques analizan el medio marcado y estiman una posible marca y posteriormente la insertan en otro medio. Todo esto es llevado a cabo sin el *conocimiento* del algoritmo de incrustación de la marca.

²El método LSB oculta información en una imagen sustituyendo los bits de la marca por los bits menos significativos de los píxeles.

ATAQUES ACTIVOS: Este tipo de ataques tiene como principal objetivo el remover la marca o el no permitir la identificación de la misma. Para ello el medio marcado sufre alteraciones o distorsiones [VPP⁺01]. Ejemplos de este tipo de ataques son los siguientes:

- ATAQUES DE ELIMINACIÓN: Buscan eliminar la marca del estegomedio. Por ejemplo: la compresión, la adición de ruido, la cuantización, los recortes y el filtrado principalmente.
- ATAQUES DE SINCRONIZACIÓN: Buscan afectar la sincronización del detector de la marca con la información incrustada, con el fin de que no se pueda encontrar la marca oculta, aunque ésta siga dentro del medio marcado. Ejemplos de estos tipos de ataques son: la rotación, el desplazamiento, el escalado y el espejo de la imagen.
- ATAQUES DEL PROTOCOLO O DE FALSIFICACIÓN: Los atacantes insertan una marca en el estegomedio y claman al medio marcado como de su propiedad, esto crea ambigüedad con respecto al propietario del medio marcado.

Hasta ahora se ha revisado la clasificación de los algoritmos de marcas de aguas, así como las métricas de evaluación de los mismos. A continuación se describen los principios de la codificación fractal de imágenes. Todo esto es con el fin de conocer los conceptos que se abordarán en los capítulos siguientes.

2.2. Codificación Fractal de imágenes

2.2.1. Naturaleza fractal

El término *fractal* fue acuñado por Benoit Mandelbrot en 1983 (descubridor de uno de los más bellos y complejos conjuntos matemáticos, que lleva su nombre) con

un neologismo derivado de la palabra latina *fractus* que significa "interrumpido", "irregular" o "fraccionario". Este nombre describe muy bien una de las propiedades geométricas de tales objetos: tienen una *dimensión fraccional* [Man82].

Básicamente los fractales se caracterizan por dos propiedades: *autosemejanza* (o *autosimilitud*) y *autorreferencia*. La autorreferencia determina que el propio objeto aparece en la definición de sí mismo, con lo que la forma de generar el fractal necesita algún tipo de algoritmo recurrente. La autosemejanza implica invarianza de escala, es decir, el objeto fractal presenta la misma apariencia independientemente del grado de ampliación al que sea sometido. Por más que se amplíe cualquier zona de un fractal siempre hay estructura hasta el infinito, apareciendo muchas veces el objeto fractal inicial contenido en sí mismo.

Existen fractales con formas variadas, inclusive con formas encontradas en la naturaleza, como por ejemplo nubes, montañas, árboles, hojas, etc.. La figura 2.4 muestra fractales con formas de plantas (fuente: [MK07]), en ella se pueden observar las dos características mencionadas con anterioridad [Bar88].



Figura 2.4: Fractales con forma de plantas.

A finales de la década de los 80's Michael Barnsley y su grupo lograron modelar y representar objetos de la naturaleza utilizando una técnica llamada *teorema del collage* que utiliza un sistema de funciones iteradas o IFS (capítulo 2.2.3) las cuales fueron propuestas años atrás por John Hutchinson. Esta técnica consiste en aplicar una función que transforme a un objeto (línea, punto, plano, imagen, etc.) y lo realice iterativamente, a tal grado de llegar a un punto en que la imagen

resultante no cambie. Por ejemplo, la figura 2.5 muestra una imagen de una hoja de maple. Dicha imagen fue hecha por un sistema de cuatro funciones iteradas.

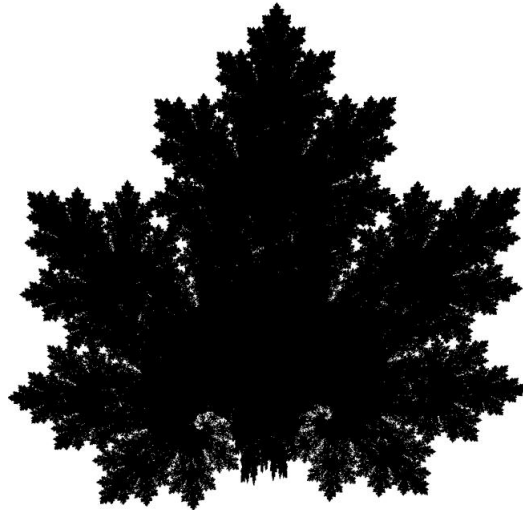


Figura 2.5: Fractal *hoja de maple* generado por un sistema de cuatro funciones iteradas.

2.2.2. Compresión fractal de imágenes

Las matemáticas fractales son buenas para generar imágenes de calidad de la naturaleza, pero ahora obsérvese de manera inversa, ¿serviría para comprimir imágenes?, esto es, que dada una imagen se obtenga un sistema de funciones iteradas que genere a la imagen original (total o muy cercanamente). Esto es conocido como el problema inverso y permanece sin solución.

En 1988 Barnsley muestra una serie de imágenes comprimidas utilizando su teorema del collage. Pero éstas fueron hechas manualmente, es decir, el proceso computacional de compresión debía ser asistido en su totalidad por una persona. Fue un estudiante suyo, Arnaud Jacquin, quien hizo obsoleto su algoritmo, al lograr hacer totalmente automático su proceso, utilizando lo que él llama *Sistema de Funciones Iteradas Particionadas* o *PIFS* (sección 2.2.5). Este algoritmo puede automáticamente convertir una imagen en un PIFS, comprimiéndola en el proceso.

Para su tesis de doctorado, Jacquin implementó su algoritmo en software y, aunque lento, logró que fuera totalmente automático. No obstante, pagó un gran precio y fue que el alto factor de compresión se esfumó. Una imagen a color de 24 bits puede ser comprimida entre 8:1 y 50:1 para que aún se pueda ver bien [Jac92].

2.2.3. Sistemas de funciones iteradas

Una forma de crear fractales es mediante los *Sistemas de Funciones Iteradas* o *IFSs*, por sus siglas en inglés. El funcionamiento de los IFSs es relacionado con el ejemplo de la máquina copiadora mostrada en [Fis95]. Imagine un tipo especial de máquina copiadora que reduce la imagen a ser copiada por la mitad y la reproduce tres veces en la imagen de salida, como se muestra la figura 2.6. A continuación la imagen de salida se vuelve a fotocopiar, lo que produce una imagen con nueve versiones reducidas de la imagen original, de igual manera se toma esta imagen de salida y se vuelve fotocopiar. La figura 2.7 muestra la iteración de este proceso en tres ocasiones, se puede notar que llega un momento en que todas las imágenes resultantes convergen en una misma figura sin importar cual sea la imagen inicial, también se puede observar que la figura resultante es una copia de si misma y tiene detalle en toda escala, tal imagen resultante es un *fractal*. Esta imagen final se conoce como *atractor del sistema*. En este ejemplo, la máquina fotocopidora representa un sistema de tres funciones y el acto de retroalimentación que transforma la imagen de salida en imagen de entrada representa la parte iterativa del sistema. Cada una de las tres funciones contrae y transforma a la imagen de entrada y en conjunto forman una imagen de salida con tres diferentes figuras, lo que representa a un Sistema de Funciones Iteradas. Cada función de un IFS, consta de una transformación afín. Una *transformación afín* (*affine transformation*) consiste de una rotación, una traslación y escalamiento que afecta a cada uno de los puntos que componen a la figura o curva fractal que está siendo estudiada. En pocas palabras lo que sucede es que un punto con

coordenadas (x, y) se traslada hasta quedar ubicado en las coordenadas (a', b') . La ecuación que controla dicha transformación es:

$$\begin{bmatrix} a' \\ b' \end{bmatrix} = w \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \quad (2.10)$$

o lo que es lo mismo

$$\begin{bmatrix} a' \\ b' \end{bmatrix} = \begin{bmatrix} ax + by + e \\ cx + dy + f \end{bmatrix} \quad (2.11)$$

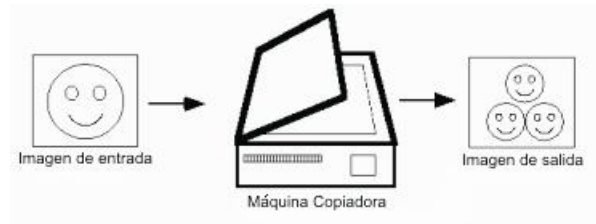


Figura 2.6: Máquina copiadora que hace tres versiones reducidas de la imagen de entrada.

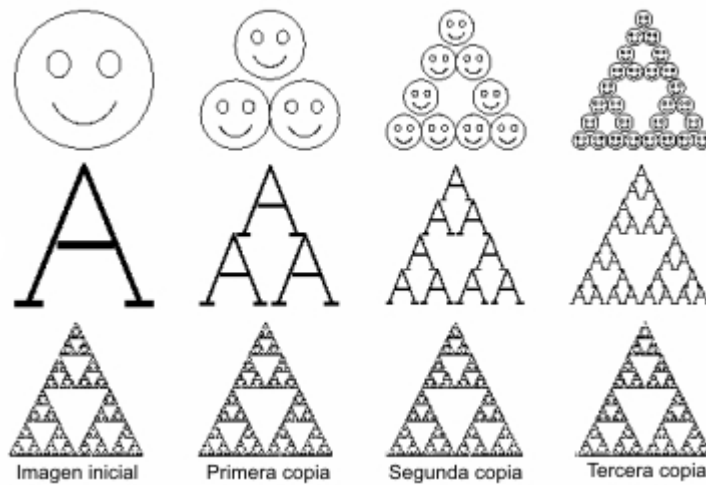


Figura 2.7: Primeras tres copias generadas por la máquina copiadora de la figura 2.6.

Los parámetros a , b , c y d se encargan de realizar la rotación de cada punto, mientras que sus magnitudes corresponden al factor de escalamiento, e y f son

responsables de realizar la traslación lineal en x y en y del mismo punto. Un Sistema de Funciones Iteradas tiene como forma general

$$T_k(x) = \begin{bmatrix} a_k & b_k \\ c_k & d_k \end{bmatrix} \cdot x + \begin{bmatrix} e_k \\ f_k \end{bmatrix} \quad (2.12)$$

Para $1 < k < n$, donde x es un punto en el plano \mathfrak{R}^2 y n es en número total de transformaciones afines. Si T_k es un mapeo contractivo, entonces el atractor puede obtenerse a través de un conjunto de funciones iteradas. En general un IFS puede ser clasificado como *IFS determinista* o *IFS aleatorio*. El IFS determinista tiene un alto costo computacional y en su lugar se utiliza un algoritmo caótico para la generación del atractor o fractal resultante. Dicho algoritmo caótico fue presentado por M. Barnsley como el juego del caos en [Bar88] y utiliza un conjunto de probabilidades $p = p_1, p_2, \dots, p_n$, donde la probabilidad p_k está asociada a T_k . La figura 2.8 muestra una imagen fractal generada por un sistema de cuatro transformaciones afines contractivas o funciones iteradas con probabilidades $p_1 = p_2 = p_3 = p_4 = 0.25$, mostradas en las ecuaciones (2.13) - (2.16).

$$T_1(x) = \begin{bmatrix} 0 & 0 \\ 0 & 0,16 \end{bmatrix} \cdot x + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (2.13)$$

$$T_2(x) = \begin{bmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{bmatrix} \cdot x + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix} \quad (2.14)$$

$$T_3(x) = \begin{bmatrix} 0.2 & -0.26 \\ 0.23 & 0.22 \end{bmatrix} \cdot x + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix} \quad (2.15)$$

$$T_4(x) = \begin{bmatrix} -1.15 & 0.28 \\ -0.26 & 0.24 \end{bmatrix} \cdot x + \begin{bmatrix} 0 \\ 0.44 \end{bmatrix} \quad (2.16)$$

2.2.4. Similitudes en las imágenes

Una imagen típica de un rostro, figura 2.9, no contiene el tipo de autorreferencia como las encontradas en las figuras 2.4, 2.5 y 2.8. La imagen no aparenta



Figura 2.8: Fractal generado por un sistema de cuatro funciones iteradas:(2.13) - (2.16).

contener transformaciones afines o alguna clase de autosimilitud de sí misma. La figura 2.10 muestra ejemplos de regiones de Lenna que son similares en diferentes escalas: una parte del hombro está sobrepuesta sobre una parte más pequeña que es casi idéntica y una región de su reflejo es similar a una pequeña región de su sombrero, después de una transformación.

En la figura 2.5, por ejemplo, se puede notar que dicha imagen fue formada utilizando copias completas de sí misma (después de aplicar las apropiadas transformaciones afines), tal y como se explica en el ejemplo de la máquina fotocopiadora, pero en el caso de la imagen de Lenna, ésta no puede ser formada utilizando copias completas de sí misma, sin embargo, si puede ser formada utilizando porciones de sí misma transformadas apropiadamente. Cabe mencionar que tales partes no son totalmente similares entre sí, son una aproximación, por lo que la imagen formada también será una aproximación de la imagen original. Esta búsqueda de similitudes dentro de una imagen es la base de la codificación



Figura 2.9: Imagen original de Lenna

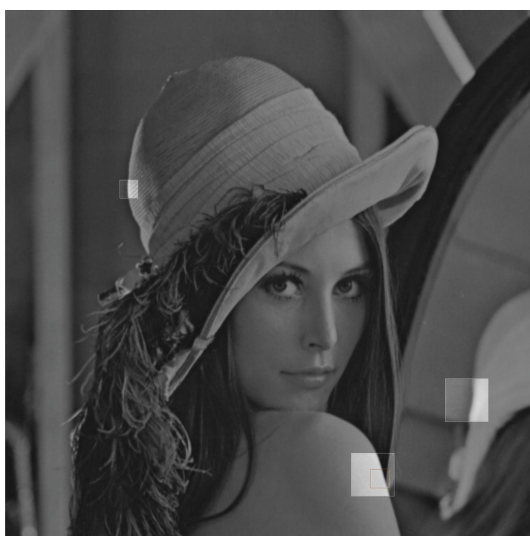


Figura 2.10: Partes autosimilares de Lenna

fractal [Fis95].

2.2.5. Codificación fractal de imágenes usando PIFS

El proceso de encontrar un IFS, tal que genere a la imagen a comprimir o un acercamiento de ella es tan complejo que parece imposible, ya que los escenarios son muy diversos, además éstos no obedecen a la geometría fractal, por lo tanto

para poder capturar la diversidad de las imágenes reales se utiliza el *sistema de funciones iteradas particionadas (PIFS)* [Jac92].

Esta técnica consiste en un conjunto de funciones iteradas, cada una de las cuales actúa codificando y reproduciendo solamente un segmento de la imagen, explotando así la similitud interna de bloques de la imagen. Por lo que la imagen se codifica en partes; de ahí el nombre.

La compresión de imágenes utilizando PIFS se considera fractal porque, al igual que las IFS generan un atractor que es autosemejante y autorreferenciado, tal atractor es la imagen comprimida.

2.2.5.1. Compresión fractal utilizando PIFS

La compresión de imágenes utilizando PIFS, se lleva a cabo a través de los pasos que se muestran a continuación. A la vez que se muestran los pasos se ejemplifica con una implementación llevada a cabo con una imagen de 256×256 píxeles en escala de grises.

1. PARTICIONAMIENTO DE LA IMAGEN. La imagen original se divide en $N_R \times N_R$ regiones sin traslapes denominadas *rangos*, cada una de ellas de $n \times n$ píxeles.

Tomando la imagen Lenna (Figura 2.9) como imagen a comprimir, se hace con ella una partición de 32×32 bloques rango (Figura 2.11) con 8×8 píxeles para cada uno. Con lo que $N_R = 32$ y $n = 8$.

2. FORMACIÓN DEL CONJUNTO DE BLOQUES DOMINIO. El conjunto de bloques dominio se forma haciendo una partición en $N_D \times N_D$ bloques traslapantes por la mitad, cada uno de ellos de $2n \times 2n$ píxeles, con lo que $N_D = N_R - 1$.

De esta forma se tienen 31×31 bloques dominio, cada uno de ellos con 16×16 píxeles. No es necesario tener físicamente al conjunto de bloques

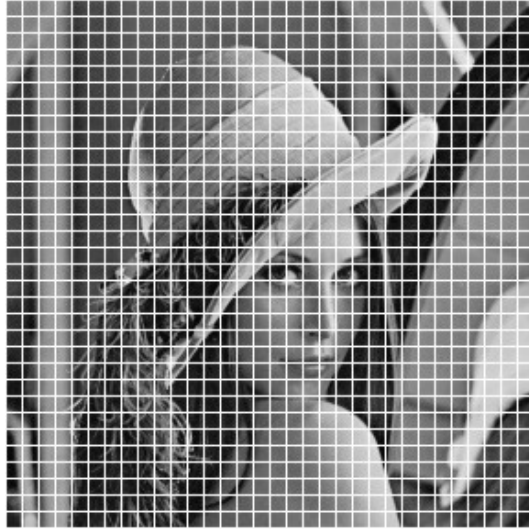


Figura 2.11: Imagen de Lenna dividida en bloques rango de 8×8 píxeles

dominio, cada bloque dominio se extrae de la misma imagen de entrada, mediante la unión de cuatro bloques rango contiguos.

3. CONTRACCIÓN ESPACIAL DE LOS BLOQUES DOMINIO. Los bloques dominio son reducidos de tamaño submuestreando o decimando sus columnas y sus renglones en un factor de dos.

La figura 2.12 muestra la contracción de la imagen dominio usando decimación (eliminación intercalada de renglones y columnas de píxeles).

4. EXTRACCIÓN DE UN BLOQUE RANGO DE LA IMAGEN. Se extrae de la imagen un bloque rango y se calcula su valor promedio.

El cálculo del valor promedio del bloque rango u -ésimo es:

$$\bar{r}_u = \frac{1}{n} \left(\sum_{i=1}^n \sum_{j=1}^n r_u(i, j) \right) \quad (2.17)$$

5. EXTRACCIÓN DE UN BLOQUE DOMINIO. Se extrae de la imagen 2.12 un bloque dominio y se calcula el valor promedio.

Para el cálculo del valor promedio del u -ésimo bloque dominio \bar{D}_u se utiliza también la fórmula 2.17.

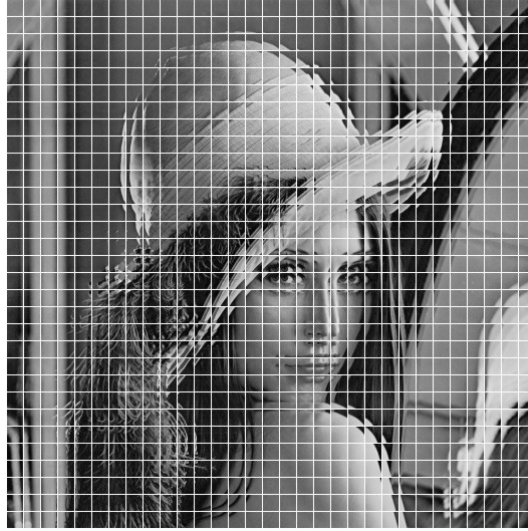


Figura 2.12: Imagen dominio contraída en 31×31 bloques dominio de 8×8 píxeles

6. CÁLCULO DEL PARÁMETRO DE AJUSTE DE CONTRASTE Y DE BRILLO.

Utilizando las ecuaciones (2.18) y (2.19) se calculan los parámetros de ajuste de contraste S_u y brillo O_u respectivamente.

Los parámetros de contraste y brillo ayudan a que el bloque dominio v -ésimo se parezca más al bloque rango u -ésimo, esto es para disminuir el error entre los bloques.

$$S_u = \sum_{i=1}^n \sum_{j=1}^n (r_u(i, j) - \bar{r}_u) (d_v(i, j) - \bar{d}_v) \quad (2.18)$$

$$O_u = \frac{1}{n^2} \left(\sum_{i=1}^n \sum_{j=1}^n r_u(i, j) - S_u \sum_{i=1}^n \sum_{j=1}^n d_v(i, j) \right)$$

$$O_u = \bar{r}_u - S_u \bar{d}_v \quad (2.19)$$

7. TRANSFORMACIÓN DE LUMINANCIA DEL BLOQUE DOMINIO. Se transforma la luminancia del bloque dominio de acuerdo a la ecuación (2.20), es decir:

$$d(i, j) = S_u d(i, j) + O_u \quad (2.20)$$

8. CÁLCULO DEL ERROR DE APROXIMACIÓN DEL BLOQUE DOMINIO AL BLOQUE RANGO. Se evalúa la semejanza entre el bloque dominio transformado y el

bloque rango, utilizando como criterio de distorsión el error $RMSE$ de la ecuación (2.5).

9. EVALUACIÓN DEL ERROR DE APROXIMACIÓN DEL BLOQUE DOMINIO QUE SE PROCESA. Se evalúa si el error de aproximación del bloque dominio que se procesa es el menor error de aproximación que se produce entre todos los bloques dominio. Si lo es, entonces se seleccionan los parámetros de la transformación del bloque como los *parámetros fractales* que codifican al bloque rango que está bajo consideración. Los parámetros fractales son: tipo de transformación, brillo, contraste, desplazamiento horizontal y desplazamiento vertical.

En caso de que el bloque dominio actual no produzca el menor error de aproximación, entonces se considera otro bloque dominio, y así sucesivamente hasta que se hayan considerado todos los bloques dominio.

Para cada bloque dominio se evalúan todas las transformaciones, en este caso ocho. Para todas ellas se calcula en error $RMSE$ y se comparan entre sí, seleccionando la menor de cada bloque dominio y a la vez seleccionando la menor de todos los bloques dominio. Las transformaciones llevadas a cabo en esta codificación son las llamadas transformaciones afines. La tabla 2.1 muestra las transformaciones que se evalúan en este método.

10. SELECCIÓN DE OTRO BLOQUE RANGO. Una vez que se ha completado la codificación de un bloque rango, comparándolo con todos los bloques dominio y sus transformaciones, se toma el siguiente bloque rango, y el proceso de codificación descrito se repite para este nuevo bloque. El proceso entero termina cuando se hayan codificado todos los bloques rango.

El *código fractal* de la imagen de este ejemplo consta de los cinco parámetros fractales para cada una de las 1024 funciones iteradas que actúan sobre la imagen particionada en 1024 bloques rangos.









Descripción de la transformación	Transformación
Identidad	
Reflexión con respecto al eje vertical $x=-x$	
Reflexión con respecto al eje horizontal $y=-y$	
Rotación 180° a la derecha	
Reflexión diagonal $y=x$	
Rotación 90° a la derecha	
Rotación 270° a la derecha	
Reflexión diagonal $y=-x$	

Tabla 2.1: Transformaciones isométricas efectuadas en los bloques dominio

2.2.5.2. Descompresión fractal utilizando PIFS

El proceso de descompresión, a diferencia del proceso de compresión, toma mucho menos tiempo. La decodificación de la imagen se realiza de manera iterativa y el proceso converge en la formación del *atractor de la imagen*. Los pasos para la descompresión se describen a continuación.

1. SELECCIÓN DE IMAGEN INICIAL. Se selecciona la imagen inicial del mismo tamaño que la imagen comprimida.

La imagen utilizada en el ejemplo es una imagen de 256×256 píxeles con color gris (puede ser de cualquier tipo), en la escala de grises de 256 valores, la imagen tiene el color 128. La tabla 2.2 muestra a la imagen inicial en la iteración 0.

2. SELECCIÓN DE UN PARÁMETRO FRACTAL. Se selecciona un parámetro fractal de los formados en el proceso de compresión, el cual genera un bloque rango.
3. TRANSFORMACIÓN DE UN BLOQUE DOMINIO REFERENCIADO POR EL PARÁMETRO FRACTAL. El bloque dominio con posición dada por los parámetros fractales posición horizontal y vertical es modificado usando los parámetros de contraste, brillo y transformación.

Al bloque dominio seleccionado usando los parámetros fractales de posición horizontal y vertical se le aplica la transformación de luminancia, dada por la ecuación (2.20), utilizando los parámetros de brillo y contraste. Posteriormente se aplica la transformación afín, tabla 2.1, dada por el quinto parámetro.

4. CONTRACCIÓN DEL BLOQUE DOMINIO TRANSFORMADO Y REPOSICIONAMIENTO. El bloque dominio es reducido de tamaño submuestreando o decimando sus columnas y sus renglones en un factor de dos. Teniendo el bloque contraído, se posiciona en el lugar del bloque rango al cual se refieren los parámetros fractales, modificando así la imagen inicial.

El bloque dominio contraído se copia en la posición dada por el número de parámetro fractal procesado, por ejemplo para el primer parámetro fractal corresponde la posición del primer bloque rango $(1, 1)$, para el segundo $(1, 2)$, etc.. Los códigos fractales representan al IFS que forma a un solo bloque rango, éstos se van almacenando secuencialmente fila a fila.

5. ITERACIÓN PARA LLEGAR AL ATRACTOR DEL SISTEMA. Se repiten los pasos 2 al 5 hasta llegar a la imagen atractor (figura 2.13), el número de iteraciones está dado por el error de aproximación del atractor actual con el atractor de la iteración anterior.



Figura 2.13: Imagen resultante tras llevar a cabo la codificación fractal. Resolución de 256×256 píxeles y bloques rango de 8×8 píxeles

La tabla 2.2 muestra el proceso de descompresión iteración por iteración. El proceso de descompresión empieza generalmente con un fondo gris plano (aunque puede ser cualquier otro). Luego el conjunto de transformaciones es repetidamente aplicado. Después de cuatro iteraciones el atractor se estabiliza, y el resultado aparece, que aunque no siendo una replica del original, es razonablemente parecido.

Dos características muy importantes de la compresión fractal son la *inescalabilidad* (sin escala) y el *mejoramiento de resolución*. Se dice que una imagen fractal carece de escala, ya que en la imagen puede hacerse zoom infinito, esto gracias a que las transformaciones de afinidad son especialmente contractivas, los detalles son creados con una resolución cada vez más fina en cada iteración. El mejoramiento de resolución se logra comprimiendo una imagen, expandirla a una resolución más alta, guardar, y luego descartar el IFS [Ram05].



Tabla 2.2: Iteraciones de la etapa de descompresión fractal.

2.2.5.3. Número de pasos en la compresión y descompresión fractal

Los pasos llevados a cabo por la compresión y descompresión fractal son *asimétricos*, ya que la compresión de la imagen consta de más pasos que la des-

compresión. En [Ram05] se hace un análisis de los pasos llevados a cabo por los procesos de compresión y descompresión, la cantidad de pasos es la siguiente:

$$\text{PasosCompresión} = N_{BR} \times N_{BD} \times N_{Tr} \times \text{ComparacionRD} \quad (2.21)$$

Donde N_{BR} representa el número de bloques rango, es decir, $N_{BR} = N_R \times N_R$, N_{BD} representa el número de bloques dominio, siendo $N_{BD} = N_D \times N_D$, N_{Tr} representa el número de transformaciones afines en este caso $N_{Tr} = 8$ y ComparacionRD representa el número de pasos llevados a cabo en la comparación de un bloque rango con un bloque dominio y tiene una magnitud de $14(n \times n) + 20$ pasos llevados a cabo, siendo $n \times n$ el tamaño en píxeles de los bloques rango y dominio, con lo que:

$$\text{PasosCompresión} = N_R \times N_R \times N_D \times N_D \times N_{Tr} \times (14(n \times n) + 20) \quad (2.22)$$

Para la imagen de ejemplo (256×256 píxeles) con bloques rango de 8×8 píxeles, la compresión toma 7,053,770,772 pasos, es decir

$$\text{PasosCompresión} = 32 \times 32 \times 31 \times 31 \times 8 \times (14(8 \times 8) + 20) = 7,053,770,772$$

El número de pasos llevados a cabo por la descompresión es menor que el de la compresión, tales pasos se describen a continuación

$$\text{PasosDescompresión} = N_{BR} \times \text{ReconstruirBR} \times \text{Iteraciones} \quad (2.23)$$

Donde ReconstruirBR representa el número de pasos llevados a cabo para reconstruir un bloque rango, esto es, tomar un bloque dominio, transformarlo, contraerlo y almacenarlo en la posición del bloque rango, tal acción toma $9(n \times n) + 8$ pasos, Iteraciones es el número de ciclos llevados a cabo por el proceso, por lo general son 10 para que el error de aproximación sea mínimo. De esta manera los pasos que toma la descompresión son

$$\text{PasosDescompresión} = N_R \times N_R \times (9(n \times n) + 8) \times 10 \quad (2.24)$$

Para la misma imagen de ejemplo utilizada anteriormente, la descompresión toma 5,980,160 pasos, esto es

$$\text{PasosDescompresión} = 32 \times 32 \times (9(8 \times 8) + 8) \times 10 = 5,980,160$$

De esta forma, la descompresión toma mucho menos pasos, por consiguiente menos tiempo que la compresión, conforme la imagen sea más grande y los bloques rango más pequeños la diferencia de pasos entre los dos procesos incrementa. En el ejemplo anterior la descompresión es 1,179 veces más rápida, si se disminuye el tamaño de los bloques rango en 4, la descompresión será 5,097 veces más rápida.

Para reducir esta diferencia se debe tomar en consideración el número de bloques rango y el número de bloques dominio, para ello se puede cambiar la forma y tamaño de tales bloques. La calidad de la imagen comprimida también será mejorada.

En este capítulo se analizaron los conceptos básicos de la esteganografía y de la compresión fractal de imágenes. Además se explicó el primer esquema de codificación fractal de imágenes, el cual es la fuente de inspiración de muchos esquemas de marcas de agua, incluyendo el propuesto en esta tesis.

Capítulo 3

Estado del arte

Como se ha mencionado, la idea principal de la codificación fractal de imágenes es determinar un conjunto de transformaciones contractivas para aproximar un bloque dominio de una imagen con un bloque rango. La composición de cada una de esas transformaciones genera una imagen atractora, la cual es una aproximación de la imagen original.

Existen muchos esquemas de marcas de agua, sin embargo son pocos los que utilizan o están inspirados por la codificación fractal. En el presente capítulo se abordarán los principales esquemas que utilizan o que han sido inspirados por esta forma de ocultar información.

3.1. Esquemas que utilizan la codificación fractal

Uno de los primeros trabajos inspirados por la codificación fractal de imágenes es el desarrollado por Joan Puate y Fred Jordan, descrito en [PJ96]. En este trabajo la marca se oculta dentro del proceso de compresión de la imagen, esto es, se modificó el algoritmo de compresión fractal propuesto por Jacquin [Jac92] limitando la búsqueda de bloques dominio a una región de búsqueda local (RBL), lo que dio lugar a los sistemas de funciones iteradas locales (LIFS por sus siglas

en inglés). La incrustación de la marca es llevada a cabo bit a bit. Cada uno es ocultado al sustituir un bloque dominio por un bloque rango transformado, con lo que el número de bloques rango determina la capacidad máxima. En el proceso de incrustación, el conjunto de bloques dominio D para un determinado bloque rango Rb está limitado por los subconjuntos RBL_A , RBL_B y RBL_C (RBL_C es formado por la unión de RBL_A y RBL_B), los cuales representan las regiones en donde se buscará el bloque que minimice, mediante una transformación, el error con el bloque Rb (figura 3.1). La regla de incrustación es simple, si se quiere incrustar un bit 1 se busca un bloque dominio en RBL_A , si se desea ocultar un bit 0 se busca en RBL_B , si la marca está totalmente incrustada el resto de los bloques rango son codificados buscando en RBL_C . Dado que la imagen es completamente dividida en bloques rango (es codificada fractalmente), este esquema utiliza una gran redundancia de información.

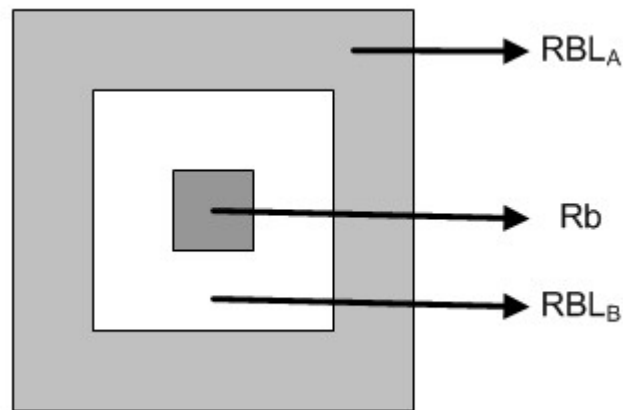


Figura 3.1: Bloque rango Rb y sus regiones de búsqueda local RBL_A , RBL_B . La RBL_C es formada por la unión de las anteriores.

Puesto que la codificación fractal de imágenes genera un vector de parámetros "fractales", el proceso de extracción de la marca es muy simple. Si el código fractal apunta a un bloque dominio en RBL_A , un bit 1 es extraído, si apunta a RBL_B un bit 0 es obtenido. Para determinar el fin del marca se ocultan 0's y 1's como delimitadores en RBL_C .

Los resultados reportados por este esquema no son muy buenos en lo que corresponde a la percepción visual, ya que el error $PSNR$ entre la imagen original y la imagen codificada es de 31.5 y 25.4 dB con bloques rango tamaño 4×4 y 8×8 respectivamente. Sin embargo, se muestran resultados aceptables en lo que corresponde a la compresión JPEG con calidades de 90 %, 75 % y 50 % al incrustar una marca de 32 bits. Las tablas 3.1 y 3.2 muestran los resultados reportados con bloques rango de tamaño 4×4 y 8×8 respectivamente. Se atacó a la imagen con un filtro pasa bajas con una ventana de 3×3 y se logró extraer la marca de 32 bits en su totalidad con una fiabilidad de 69.7 %. La fiabilidad corresponde al porcentaje de éxito al extraer un bit incrustado, sin embargo la fiabilidad es calculada tomando en cuenta únicamente los bits ocultados en RBL_A , por lo que no representa una métrica totalmente objetiva.

Bloques Rango tamaño 4×4				
	No JPEG	JPEG 90 %	JPEG 75 %	JPEG 50 %
Fiabilidad (%)	98.5	73.5	40.8	21.4
Bits detectados	32	32	32	29

Tabla 3.1: Resultados obtenidos por el esquema de Puate usando bloques rango tamaño 4×4 y redundancia 50.

Bloques Rango tamaño 8×8				
	No JPEG	JPEG 90 %	JPEG 75 %	JPEG 50 %
Fiabilidad (%)	99.25	92.3	88.8	77.2
Bits detectados	32	32	32	32

Tabla 3.2: Resultados obtenidos por el esquema de Puate usando bloques rango tamaño 8×8 y redundancia 25.

Como se muestra en las tablas anteriores, la marca es extraída casi en su totalidad después de los ataques de compresión y filtrado pasa bajas, sin embargo hay que tomar en cuenta que se utilizó una redundancia de hasta 50, esto es, se ocultó el mismo bit en 50 diferentes bloques rango.

En este esquema la marca incrustada se puede extraer en su totalidad, sin embargo las pruebas fueron hechas sólo en la imagen de Lenna con 256×256 píxeles (Figura 2.9), además se utilizó una redundancia elevada (25 ó 50). El ataque en la percepción visual es una desventaja muy marcada de este algoritmo.

Li Guanhua, Zhao Yao y Yuan Baozong propusieron en [GYB02] un esquema parecido al de Puate. Este esquema incrusta la marca en el proceso de compresión fractal de imágenes.

Como se ha mencionado, los parámetros fractales de una imagen incluyen: factor de escala de luminancia S_u (contraste), factor de desplazamiento de luminancia O_u (brillo), factor de transformación isométrica (transformadas afines) e información de la posición relativa del bloque rango con su respectivo bloque dominio. Los autores de este algoritmo utilizaron el ajuste de brillo para marcar la imagen, a diferencia de lo propuesto por Puate y Jordan quienes utilizaron la posición relativa de un bloque rango y su respectivo bloque dominio.

Para la incrustación de la marca, primero se propone la realización de la codificación fractal de la imagen, para confirmar todos los posibles valores del factor de brillo O_u , en donde cada bloque rango obtiene su apropiado parámetro. Luego se dividen todos estos valores en dos grupos, los cuales se definen como μ_A , μ_B y μ_C (μ_C es la unión de μ_A y μ_B), la figura 3.2 muestra como se forman estos grupos.

Cada bit de la marca es incrustado durante la codificación fractal de la siguiente forma: se selecciona un bloque rango, si se va a incrustar un bit 1, este bloque es codificado buscando su mejor valor de brillo en μ_A . Por el contrario, si se desea incrustar un bit 0, el bloque rango busca su mejor valor de brillo en μ_B . El resto de los bloques rango son codificados buscando en μ_C .

Como el proceso de incrustación es llevado a cabo a la par del proceso de codificación fractal, la imagen marcada es representada por los códigos fractales de dicha compresión. El proceso de extracción se desarrolla durante el proceso de descompresión fractal. La regla para decidir si un bloque rango ha sido marcado con un bit 0 ó con un bit 1, es la siguiente: si la mejor aproximación del brillo de

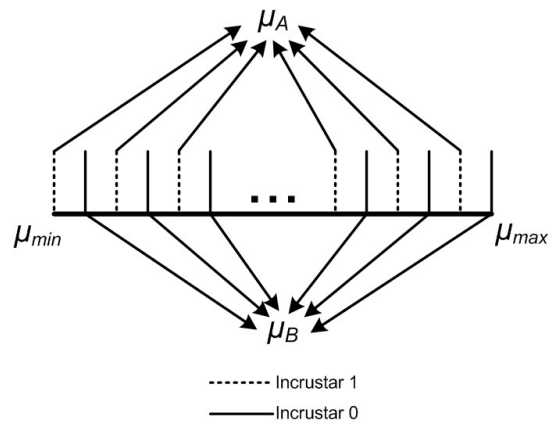


Figura 3.2: Rango de valores de los coeficientes de brillo del mínimo al máximo.

un bloque rango pertenece a μ_A entonces un bit 1 ha sido incrustado y si pertenece a μ_B un bit 0 se incrustó.

Al igual que el esquema de Puate, este esquema utiliza una región de búsqueda local o RBL para limitar la búsqueda de un bloque dominio, el cual minimice el error con un determinado bloque rango. A diferencia del esquema anterior, esta RBL no se divide en dos regiones, con ello se evita el problema de que al buscar similitudes en la región exterior (RBL_B) (ya que está más alejada del bloque rango central) se disminuya el grado de coincidencia generando una pobre robustez.

Bloques Rango tamaño 4×4					
Calidad JPEG	90 %	70 %	50 %	30 %	10 %
Bits detectados	30	26	20	12	9

Tabla 3.3: Resultados obtenidos por el esquema de Li *et-al* usando bloques rango tamaño 4×4 .

Bloques Rango tamaño 8×8					
Calidad JPEG	90 %	70 %	50 %	30 %	10 %
Bits detectados	32	32	32	29	25

Tabla 3.4: Resultados obtenidos por el esquema de Li *et-al* usando bloques rango tamaño 8×8 .

Los resultados obtenidos por este esquema son mostrados en las tablas 3.3 y

3.4. Cabe mencionar que las pruebas fueron hechas con la imagen de Lenna de 256×256 píxeles con la incrustación de una marca de 32 bits. Los ataques realizados fueron compresión JPEG con resultados mostrados en las tablas anteriores y filtro pasa bajas con un resultado de 32 bits detectados. La distorsión *PSNR* entre la imagen original y la imagen marcada es 30.05 dB con bloques rango 4×4 y 25.13 dB con bloques rango de 8×8 .

Al igual que el esquema anterior este esquema aumenta la percepción visual de la marca. Sin embargo tiene un alto costo computacional debido a la codificación fractal.

Una propiedad generada por la codificación fractal de imágenes es la llamada *punto fijo en la compresión fractal*, la cual se refiere al hecho de aplicar iterativamente la compresión fractal a una imagen hasta que la imagen inicial (previamente codificada) sea igual a la imagen después de ser codificada fractalmente, esto es que la distancia entre ellas sea cero y que los parámetros fractales sean los mismos. Esta propiedad fue utilizada por Zhen Yao en [Yao03] como una forma de marcar a una imagen, tal marca es utilizada para verificar la integridad de los datos, de tal forma que si la estegoimagen es modificada los parámetros fractales no coincidirán con los parámetros fractales de antes del ataque, pudiendo así comprobar un ataque a la integridad.

Zhen propuso además un esquema para ocultar información durante el proceso de codificación fractal de imágenes. En este esquema se utiliza una región de búsqueda local de bloques dominio para los bloques rango como se muestra en la figura 3.3.

Para el proceso de incrustación, la región de búsqueda local es dividida en dos, ya sea horizontal o verticalmente (figura 3.3), cada parte llamada RBL-0 y RBL-1 se utilizará para buscar el bloque dominio que minimice la distancia con el bloque rango *Rb* y dependiendo de la magnitud del bit que se incruste se utilizará una zona específica, de modo que la incrustación utiliza el siguiente procedimiento:

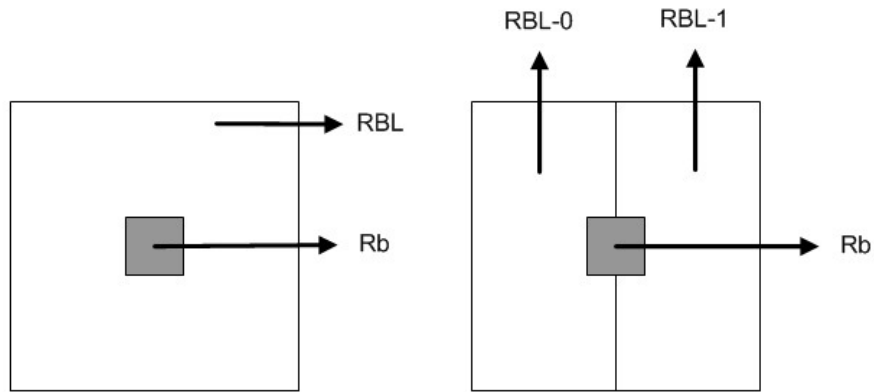


Figura 3.3: Región de búsqueda local de bloques dominio para un bloque rango Rb propuesto por Zhen.

1. Buscar en RBL-0 el bloque dominio que minimice el error E_{RBL-0} con el bloque rango Rb .
2. Buscar en RBL-1 el bloque dominio que minimice el error E_{RBL-1} con el bloque rango Rb .
3. Si se desea incrustar un bit 1 y si el error E_{RBL-0} aproxima mejor al bloque rango que el error E_{RBL-1} , se intercambian las posiciones de los bloques dominio en la imagen, de tal manera que el bloque dominio que mejor se aproxime al bloque rango quede en RBL-1.
4. Si se desea incrustar un bit 0 y si el error E_{RBL-1} aproxima mejor al bloque rango que el error E_{RBL-0} , se intercambian las posiciones de los bloques dominio en la imagen.
5. Reiniciar el proceso de codificación de la imagen desde el principio si una sustitución tuvo lugar.

El proceso de extracción de la marca es directo, sólo basta con analizar el código fractal de la imagen para saber si el bloque dominio que genera al bloque rango actual se encuentra en RBL-0 o en RBL-1.

Los resultados reportados en esta investigación son muy pocos, el error *PSNR* de la imagen original y la imagen marcada es 32.87 dB, usando la imagen de Lenna de 256×256 píxeles en una escala de grises, la marca incrustada tiene un tamaño de 8 bits.

Este esquema a diferencia de los anteriores hace un intercambio de bloques, lo que produce que la imagen original cambie, con lo que es necesario el reinicio de la codificación fractal. Esto genera que el costo computacional del algoritmo se incremente linealmente con el tamaño de la marca, además si la marca crece existe mayor posibilidad de intercambios de bloques, lo que produce un incremento en el riesgo de que exista una superposición de RBL, lo cual puede producir intercambios no deseados. Esto provocaría que el proceso sea más complejo y que la velocidad de convergencia llegue a ser imprevisible.

Este trabajo muestra la utilización de una región de búsqueda local dividida, inmersa en el proceso de compresión fractal. Como es de esperarse es un esquema computacionalmente costoso y con capacidad pequeña, sin embargo produce resultados buenos en contra de la compresión JPEG. La perceptibilidad por otra parte es muy notoria, esto es en gran medida por el esquema de compresión fractal. Uno de los requerimientos que se persiguen principalmente en los esquemas de marcas de agua es que la perceptibilidad sea mínima, pero dada la naturaleza de los esquemas analizados hasta el momento, la perceptibilidad pasa a segundo término.

3.2. Esquemas inspirados por la codificación fractal

A continuación se presentan esquemas de marcas de agua inspirados por la codificación fractal, pero a diferencia de los anteriores no son llevados a cabo durante el proceso de compresión fractal. Tres de los esquemas son propuestos

por Kamal Gulati en [Gul03], en ellos se propone la utilización de bloques rango, bloques dominio y el proceso de modificación de luminancia para generar un nuevo bloque marcado.

El primer esquema propuesto por Kamal divide la imagen portadora en regiones rango y regiones dominio, cada una de esas regiones son mostradas en la figura 3.4. Los bloques rango están dispuestos en las regiones posicionadas en los cuadrantes II y IV, las regiones posicionadas en los cuadrantes I y III son de bloques dominio.

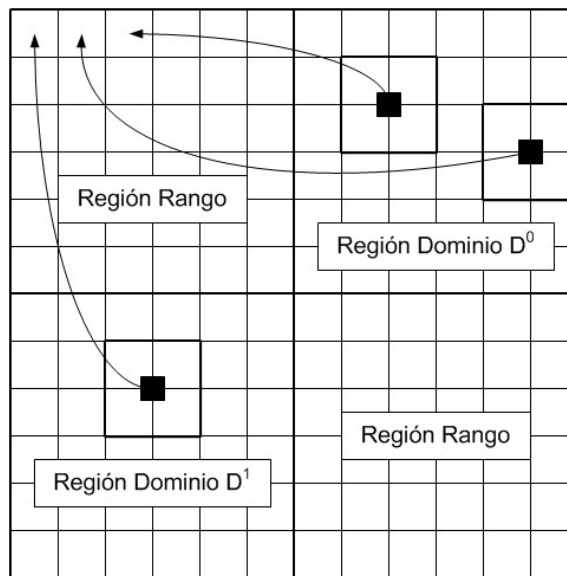


Figura 3.4: División de una imagen en regiones rango y regiones dominio.

El proceso de incrustación es llevado a cabo de manera muy parecida al esquema de Zhen, para ello primero se selecciona un bloque rango posicionado en cualquiera de las dos regiones correspondientes, luego se busca el bloque dominio que más se aproxime a él, mediante lo descrito en la sección 2.2.5, ajustando la luminancia y utilizando una métrica de distorsión, si se desea ocultar un bit 1 el bloque dominio modificado que mejor se aproxime al bloque rango debe ser parte de la región dominio D^1 posicionado en el cuadrante III, si se desea incrustar un bit 0, el bloque dominio debe ser parte de la región dominio D^0 (cuadrante I).

En la figura 3.4 se muestra un ejemplo de la incrustación de una marca de tres bits (100), se puede notar que el primer bloque rango tiene un bloque dominio que más se le aproxima en la región D^1 y los siguientes dos bloques tienen su correspondiente en la D^0 .

Algo que no es mencionado en este esquema es el caso en el que la región en donde se encuentre el bloque dominio modificado no corresponda al bit que se incrustará, es decir, que sea necesario realizar un cambio de posición, algo parecido a lo propuesto por Zhen, de ser necesario este cambio, el esquema tendrá que tomar en cuenta que la modificación de un bloque dominio puede alterar a la incrustación de un bit anterior, en caso de que los bloques rango estén superpuestos.

El segundo esquema es parecido en gran parte al esquema anterior, ya que utiliza la disposición de las regiones de bloques mostradas en la figura 3.4, sin embargo la forma de aproximar un bloque dominio a un bloque rango es la que varía. Tal proceso se lleva a cabo modificando el bloque dominio píxel a píxel tomando como pivote el primer píxel, de esta manera se modifican todos los píxeles del bloque dominio, con la única restricción de que todos sean diferentes del primer píxel. Terminando esta tarea de modificación, el bloque dominio modificado tendrá una apariencia más cercana al bloque rango actual, seguido de esto y al igual que el esquema anterior, el bloque rango es sustituido por el bloque dominio modificado. El bit incrustado será determinado por la posición en donde se ubique el bloque dominio que mejor aproxime al bloque rango, esto es, si pertenece a la región D^1 se incrustó un bit 1, si pertenece a la región D^0 se ocultó un bit 0.

La tarea de extracción se realiza buscando el bloque dominio que más se aproxime al bloque rango actual mediante la modificación píxel a píxel mencionada anteriormente. La localización del bloque dominio indicará el valor del bit extraído.

El tercer esquema considera únicamente el bit menos significativo (LSB por sus siglas en inglés) de los píxeles. Los LSB de los píxeles de un bloque rango son comparados con los LSB de los píxeles de todos los bloques dominio del cuadrante seleccionado (región D^1 o región D^0). El bloque dominio con la mínima diferencia

es escogido y su plano LSB es copiado sobre el plano LSB del bloque rango, de esta forma el plano LSB del bloque rango va a ser el mismo que el plano de su bloque dominio correspondiente, así el mapeo de los bloques es una función de identidad el cual es aplicado en el plano LSB de los bloques.

Para extraer los bits ocultos, los bloques rango son seleccionados en el mismo orden que el proceso de incrustación (aplica para los dos esquemas anteriores). Para un bloque rango seleccionado, los LSB de sus píxeles son comparados con los LSB de los píxeles de todos los bloques dominio (pertenecientes a la regiones D^1 y D^0), el bloque dominio con la menor diferencia es seleccionado y su localización indicará el valor del bit extraído.

Los resultados reportados por los esquemas propuestos por Kamal, se muestran en la figura 3.5 y en la tablas 3.5 y 3.6.

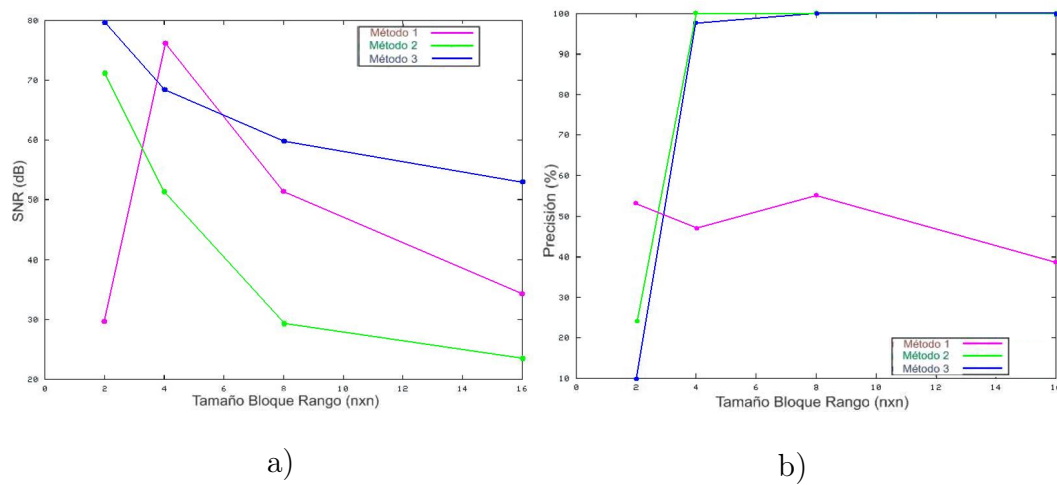


Figura 3.5: a) SNR contra tamaño de los bloques rango, b) Precisión contra tamaño de los bloques rango.

Las pruebas fueron realizadas en la imagen de Lenna de 256×256 píxeles en escala de grises con una marca de 49 bits. Como es de esperarse, en esta clase de métodos la perceptibilidad visual incrementa con respecto al tamaño de los bloques rango, sin embargo a mayor tamaño incrementa la robustez. Las tablas 3.5 y 3.6 muestran los resultados en contra de la compresión JPEG de los métodos

	B. Rango 4×4		B. Rango 8×8		B. Rango 16×16	
Calidad	<i>SNR</i>	Fiabilidad(%)	<i>SNR</i>	Fiabilidad	<i>SNR</i>	Fiabilidad
No. JPEG	52.81	89.79	52.82	85.71	52.87	91.83
JPEG 99%	48.77	73.46	48.83	79.59	48.94	89.79
JPEG 98%	44.12	67.34	44.18	71.42	44.44	95.91
JPEG 97%	41.05	59.18	41.13	71.42	41.48	75.51
JPEG 96%	37.45	51.02	37.55	75.51	37.95	81.63

Tabla 3.5: Resultados obtenidos por el segundo esquema de Kamal para compresión JPEG.

	B. Rango 4×4		B. Rango 8×8		B. Rango 16×16	
Calidad	<i>SNR</i>	Fiabilidad(%)	<i>SNR</i>	Fiabilidad	<i>SNR</i>	Fiabilidad
No. JPEG	52.81	61.22	52.81	100	52.80	100
JPEG 99%	48.78	53.06	48.78	65.30	48.76	100
JPEG 98%	44.11	38.77	44.11	57.14	44.10	51.02
JPEG 97%	41.01	28.57	41.04	38.77	41.02	53.06
JPEG 96%	37.44	42.85	37.44	55.10	37.42	51.02

Tabla 3.6: Resultados obtenidos por el tercer esquema de Kamal para compresión JPEG.

2 y 3, ambos muestran un buen comportamiento con bloques rango de tamaño 16×16 , pero al aumentar el factor de compresión, la robustez y la calidad se ven mermadas.

Otro esquema inspirado por la codificación fractal es propuesto por Patrick Bas, Jean Marq Chassery y Franck Davoine en [BCD98a, BCD98b]. Al igual que los esquemas anteriores utiliza una división de bloques rango y bloques dominio, así como la modificación de la luminancia de bloques para la incrustación de la marca.

El proceso de incrustación es llevado a cabo seleccionando primero a los bloques dominio, tantos como bits se deseen incrustar (al igual que los esquemas anteriores). A diferencia de los métodos anteriores no existe una región de búsqueda local ni regiones rango o regiones dominio.

Los bloques dominio se seleccionan directamente de la imagen completa detectando solamente aquellos con alta desviación estándar, lo cual indica que el

bloque no es parte de una región homogénea, este procedimiento se conoce como *detección de bloques dominio*. Además se emplea un mecanismo de *cuantización*, el cual limita sólo a seleccionar bloques diferentes entre sí. Después de esto se obtienen los bloques dominio aptos para la incrustación de la marca. La selección de bloques rango es llevada a cabo utilizando los bloques dominio del proceso anterior, para cada bloque dominio se busca el bloque rango que más se le aproxime mediante la modificación de la luminancia, al igual que en la compresión fractal descrita en la sección 2.2.5. La figura 3.6 muestra el proceso de selección de bloques dominio y rango.

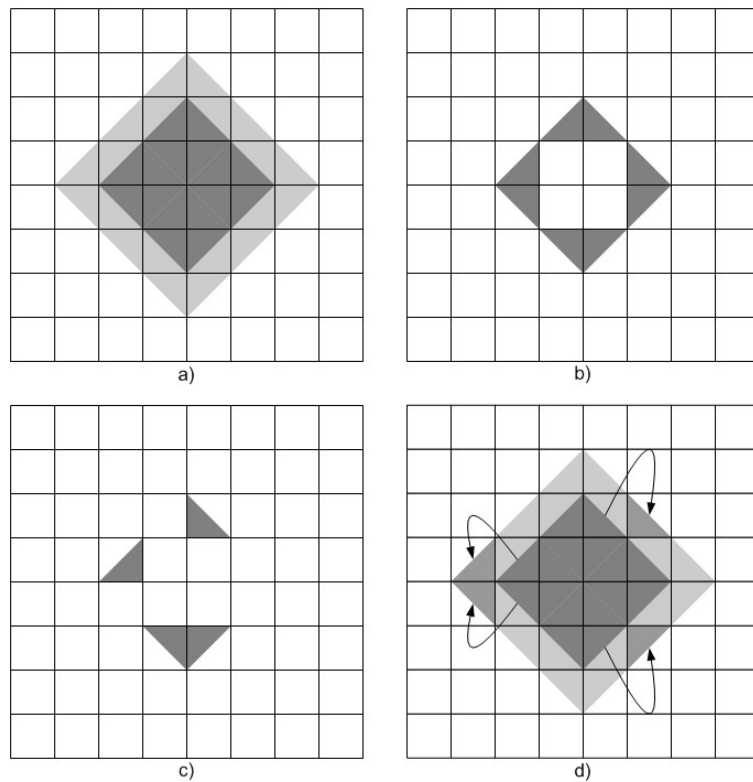


Figura 3.6: a) Imagen particionada. b) Bloques dominio detectados. c) Bloques dominio después de la cuantización d) Detección de bloques rango por medio de similitudes.

Hasta este momento se han seleccionado los bloques dominio y los bloques rango, no se ha modificado la imagen original, ni se ha insertado información alguna. La incrustación de la marca se lleva a cabo bit a bit, para ello el bloque

rango R , el cual es muy parecido a un bloque dominio D , es sustituido por una versión aproximada \hat{R} , la cual se calcula de la siguiente manera.

$$\hat{R} = \delta \cdot S \cdot \left(\frac{D}{\max(D)} \right) + \bar{R} \quad (3.1)$$

Donde \bar{R} es la media de R , S es el factor de magnitud de la marca, $\max(D)$ es el valor máximo de D y

$$\delta = \begin{cases} +1 & \text{si se incrusta un bit 1} \\ -1 & \text{si se incrusta un bit 0} \end{cases}$$

De esta forma es incrustado un bit en un bloque rango. Este proceso se repite para cada uno de los bloques dominio hasta haber terminado de incrustar la marca. La posición de los bloques rango donde se incrustó información se guarda en una tabla de referencias, ésta será usada en el proceso de detección.

Este esquema a diferencia de los otros tiene un proceso de detección, utilizado para probar la existencia de la marca extraída, para el cual se necesita tener una referencia de los bloques rango en donde se incrustaron bits. Para llevar a cabo el proceso de extracción es necesario tener la localización de los bloques dominio y sus bloques rango asociados, este proceso es similar al llevado a cabo durante la incrustación de la marca. Los pasos a seguir por ambos procesos son:

1. Obtener un bloque dominio D de la imagen.
2. Crear un bloque rango \hat{R} usando la formula 3.1 y buscar el bloque rango R que minimice el error RMS .
 - a) Si el índice de R se encuentra en la tabla de referencias de bloques rango el bit es detectado y la magnitud del bit depende del signo de δ .
 - b) Si el índice de R no se encuentra en la tabla de referencias de bloques rango el bit no es detectado.

3. Obtener otro bloque dominio D de la imagen e ir al paso 2 hasta terminar la extracción.

Como es de esperarse, este esquema tiene un alto costo computacional y por la sustitución de bloques produce una alta percepción visual teniendo un $PSNR$ de 39.75 dB pero esto depende del factor de magnitud de la marca S , en este caso se utilizó una magnitud $S = 20$. El comportamiento de este esquema ante la compresión JPEG se muestra en la gráfica de la figura 3.7, en ella se ocultaron 34 bits utilizando bloques de 8×8 píxeles.

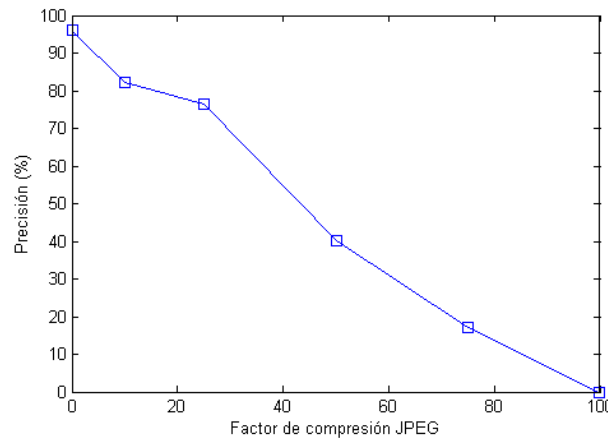


Figura 3.7: Resultados obtenidos por el algoritmo de Patrick Bas para JPEG.

Estos tres autores crearon otro esquema de marcas de agua [BCD99] que al igual que el anterior utiliza bloques rango y bloques dominio en los procesos de incrustación y extracción de la marca. El proceso de incrustación inicia con la selección de bloques dominio mediante un detector de puntos característicos (detector de Harris), de tal forma que un punto característico sea el centro del bloque dominio. Para cada bloque dominio se crea una ventana local de tamaño 4×4 bloques dominio como se muestra en la figura 3.8, la marca a incrustar es una distorsión que se hace a un bloque rango específico dentro de la ventana, tal distorsión es una versión reducida del bloque dominio central. En la figura 3.8 los bloques dominio son marcados con una cruz, la ventana es construida, luego se

selecciona un bloque rango el cual será modificado y sustituido por una versión reducida del bloque dominio, en este caso es el bloque rango de la esquina inferior izquierda. La etapa de extracción de la marca es realizada de la misma forma que la incrustación, se buscan los bloques dominio, luego se genera la ventana local, después se reconstruye un bloque marcado temporal el cual se compara con cada uno de los bloques rango de la ventana local.

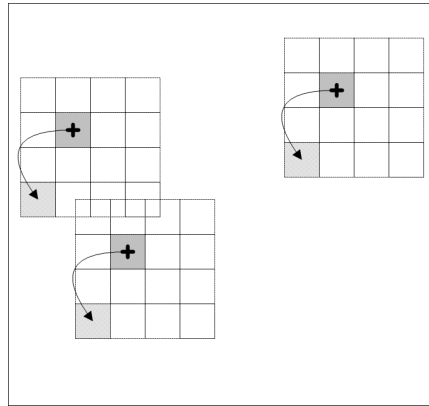


Figura 3.8: Inserción de similitudes en una ventana local.

Los resultados de este esquema muestran que es robusto en cierta medida a ataques geométricos, pero es débil en contra de ataques de filtrado y compresión JPEG. Este método es muy interesante ya que no inserta una marca de bits, sino que incrusta una modificación, lo que lo hace un buen método para proteger derechos de autor.

Existen otros esquemas de marcas de agua que utilizan la teoría nacida de la compresión fractal de imágenes, por ejemplo en [HC00] se propone el ocultar los parámetros fractales dentro de una imagen, esto es, que la marca sea una imagen codificada fractalmente. Existe otro esquema más ambicioso, propuesto por Khadivi en [Kha02], el cual oculta una marca de agua dentro de los parámetros de un sistema de funciones iteradas, de tal forma que la marca sea transformada a un fractal. La extracción es llevada a cabo usando el teorema de collage, para así poder obtener el IFS que generó a tal imagen fractal, teniendo el IFS se obtienen

los parámetros y consecuentemente la marca.

Los esquemas explicados en este capítulo son inspirados por la codificación fractal de imágenes. Los resultados mostrados hasta el momento son muy variados, pero todos tienen en común el deterioro de la imagen marcada y el elevado costo computacional de los procesos de incrustación y extracción, debido al procesamiento continuo de bloques. Estas dos principales deficiencias han hecho que estos esquemas sean considerados poco robustos y que tengan poca popularidad en la comunidad científica. Las investigaciones en la codificación fractal tienen no más de dos décadas y en los esquemas de marcas de agua inspirados por la codificación fractal tienen mucho menos tiempo. En esta tesis se hace una investigación de un novedoso esquema que contiene características que mejoran las deficiencias de este tipo de esquemas; Mejorando el costo computacional y principalmente mejorando la imperceptibilidad de la marca.

Capítulo 4

Esquema de marcas de agua propuesto

El esquema propuesto es una modificación al esquema de Patrick Bas *et-al* explicado anteriormente [BCD98a, BCD98b]. Este nuevo esquema tiene la característica principal de disminuir la perceptibilidad de la marca sin deteriorar la robustez. A continuación se describen los procesos de incrustación, extracción y detección de la marca, así como también las características y restricciones del mismo.

4.1. Especificaciones

El diagrama general del nuevo esquema se muestra en la figura 4.1, en ella se pueden observar los procesos de inserción, extracción y detección de la marca.

Como se ha descrito anteriormente, los esquemas basados en la codificación fractal tienen como característica principal la utilización de bloques de imagen para incrustar un solo bit, ésta a primera vista puede ser considerada como una desventaja, puesto que al marcar un bloque la estegoimagen se distorciona en gran medida y la capacidad de incrustación disminuye notablemente, sin embargo esta

característica intrínseca es muy importante para la robuztez de estos algoritmos, ya que un solo bit es guardado en toda una región de la imagen.

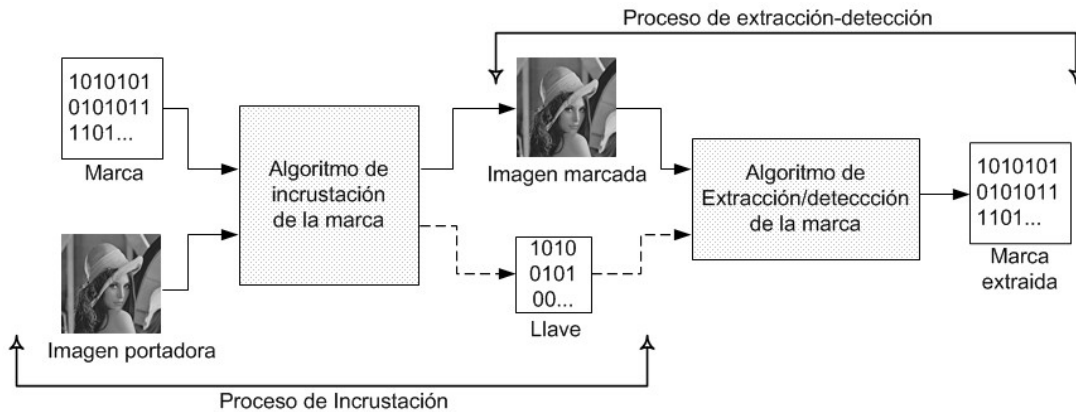


Figura 4.1: Diagrama del esquema propuesto.

El procesamiento de bloques de una imagen incrementa el costo computacional. Los esquemas tratados anteriormente sufren de esta restricción, sin embargo se han hecho mejoras utilizando distintos arreglos de bloques y mejorando las búsquedas de los mismos. En el esquema propuesto la selección de bloques dominio utiliza el detector de puntos característicos de Harris [HS88] (considerado como uno de los mejores detectores resistentes a ataques de compresión, de filtrado y geométricos [BCD02]), se limita la región de búsqueda mediante el uso de una RBL, se modificó el algoritmo de inserción para disminuir la distancia entre bloques y además se crearon regiones de incrustación para ocultar la marca redundantemente en la imagen.

A continuación se presentan los procesos de incrustación y extracción/detección del esquema propuesto, durante la explicación de tales procesos se ejemplificará con una marca $M = 1010101010101010$ de $M_n=16$ bits y como imagen portadora I la imagen de Lenna (figura 2.9) con resolución de 512×512 píxeles y con una profundidad de 8 bits en escala de grises.

4.2. Proceso de incrustación de la marca

El proceso de incrustación de la marca es llevado a cabo en dos etapas, la primera es encargada de elegir las k regiones de incrustación RI_k en las cuales la marca M se incrustará, la segunda inserta la marca M en una región de incrustación RI_k , esto es

$$\{RI_1, RI_2, \dots, RI_k\} \in I$$

En cada región de incrustación RI_i se ocultará una marca de longitud M_n

$$M = m_1m_2m_3\dots m_{M_n} \text{ con } m_i \in \{0, 1\}$$

La redundancia es determinada por k .

La selección de las k regiones de incrustación RI_k se describe a continuación.

1. SELECCION DE PUNTOS CARACTERÍSTICOS. *La imagen I_p es creada al detectar los puntos característicos de la imagen I .*



Figura 4.2: Imagen I_p de puntos característicos detectados en Lenna usando el detector de Harris.

La figura 4.2 muestra la imagen I_p . El detector de puntos característicos de Harris encuentra puntos salientes en imágenes, tales puntos se encuentran cerca de esquinas y bordes. En este nuevo esquema se utiliza una mejora del detector de puntos de Harris [BCD02], llevada a cabo mediante el uso de una operación de pre-filtrado de difuminación para evitar que el número potencial de puntos encontrados sea importante si el contenido de la imagen es ruidosa (ya que es sensible a esquinas o áreas texturizadas). Esto evita que la robustez del detector sea reducida.

Los puntos detectados en este paso son los más fuertes de la imagen I , cabe mencionar que cada uno tiene una magnitud que indica la fuerza del punto característico.

2. SELECCIÓN DE UNA REGIÓN DE INCRUSTACIÓN. *Se selecciona el punto P_{max} de mayor magnitud de I_p . La posición del punto P_{max} indica el centro de la región de incrustación RI_i de tamaño $N_B \times N_B$ bloques no traslapantes B_i de $n \times n$ píxeles cada uno.*



Figura 4.3: Región de Incrustación RI_1 con $N_B = 25$ y $n = 4$.

La figura 4.3 muestra la primera región de incrustación RI_1 detectada con

$N_B = 25$, $n = 4$ y el píxel con el punto característico más fuerte es $I_p(265, 275)$.

3. BÚSQUEDA DE OTRAS REGIONES DE INCRUSTACIÓN. *Se busca en I_p por otra posible región de incrustación RI_{i+1} .*



Figura 4.4: Regiones de Incrustación RI_i encontradas con $i = 1, 2, \dots, 8$, $N_B = 25$ y $n = 4$.

La figura 4.4 muestra las 8 regiones de incrustación RI_i encontradas en la imagen I_p con $i = 1, 2, \dots, 8$, $N_B = 25$ y $n = 4$.

Una vez seleccionadas las k regiones de incrustación RI_k , se describe el proceso de incrustación de la marca M en una región de incrustación RI_i .

1. PARTICIONAMIENTO DE LA REGIÓN DE INCRUSTACIÓN. *La región de incrustación RI_i se divide en $N_B \times N_B$ bloques no traslapantes B de tamaño $n \times n$.*

La figura 4.5 muestra a la imagen de RI_1 original y la particionada en 25×25 bloques de tamaño 4×4 píxeles para cada uno. Con lo que $N_B = 25$ y $n = 4$.



Figura 4.5: Región de Incrustación RI_1 particionada.

2. SELECCIÓN DEL CONJUNTO DE BLOQUES DOMINIO. *El conjunto de bloques dominio D se forma en dos pasos: detección y cuantización. La cantidad de bloques dominio es $|D| = M_n$, donde M_n es el tamaño de la marca.*

a) DETECCIÓN DE BLOQUES DOMINIO. *La imagen RI_p es creada al detectar los puntos característicos de RI_i , luego se realiza el particionamiento de la imagen RI_p en $N_B \times N_B$ bloques B_i con $i = \{1, 2, \dots, (N_B)^2\}$ de tamaño $n \times n$. El conjunto de bloques dominio $D' \in B$ se forma con la selección de los bloques B_i con puntos característicos.*

La figura 4.6 a) muestra la imagen RI_p , la 4.6 b) muestra la partición de RI_p y la 4.6 c) muestra los 121 bloques dominio D' detectados, los bloques claros representan a tales bloques. La métrica de detección de bloques dominio D' es seleccionar los bloques con puntos característicos.

b) DECIMACIÓN DE BLOQUES DOMINIO. *El conjunto de bloques dominio D' es decimado al eliminar los bloques que sean parecidos, formando al conjunto de bloques dominio $D \subseteq D'$.*

La figura 4.6 d) muestra los 16 bloques dominio D seleccionados (bloques oscuros), cada uno de los bloques dominio D_i son elegidos comparando la magnitud (promedio del valor de los puntos detectados en el bloque) y distancia entre los bloques D' usando la métrica RMSE de la ecuación 2.5.

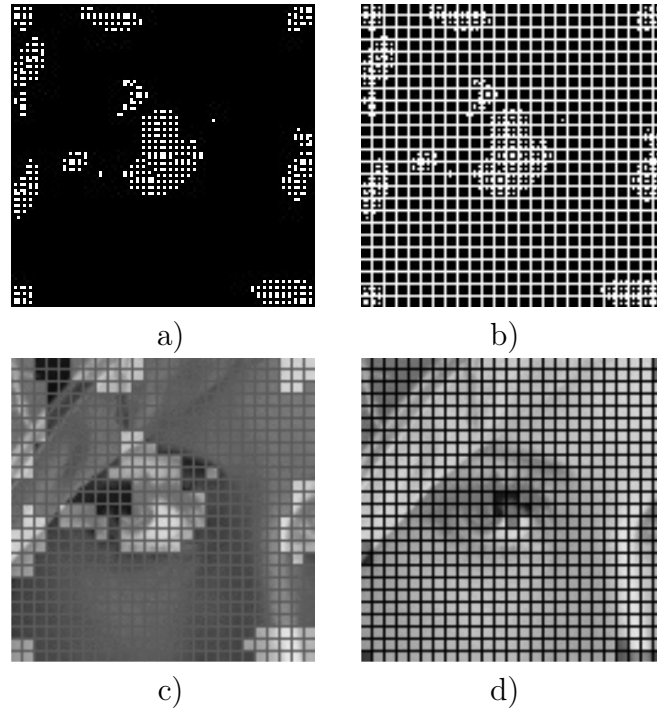


Figura 4.6: a) Región de incrustación RI_p de RI_1 , b) Partición de RI_p , c) Bloques dominio D' seleccionados (bloques claros) y d) Bloques dominio D después de la decimación (bloques oscuros).

3. SELECCIÓN DEL CONJUNTO DE BLOQUES RANGO. *Para cada bloque dominio D_i de D se construye una RBL de tamaño $N_{RBL} \times N_{RBL}$ bloques no traslapantes B_i , en donde se buscará el bloque rango R_i en el cual se incrustará el bit i de la marca. Para ello se modificará cada uno de los bloques rango $R \in RBL$ mediante la ecuación 4.1 (los bloques B_i pertenecientes a la RBL son llamados también bloques rango R). Luego se seleccionará el bloque modificado \hat{R} que mejor se aproxime al bloque rango R que lo genera.*

La posición del bloque seleccionado \hat{R} indica la posición bloque rango R_i en el cual se incrustará un bit de información. La ecuación de incrustación es

$$\hat{R} = \delta \cdot S \cdot \left(\frac{D_i - \bar{D}_i}{\max(D_i - \bar{D}_i)} \right) + \bar{R} \quad (4.1)$$

donde \bar{R} es la media de $R \in RBL$, $\max(x)$ es el valor máximo de x , la magnitud de la marca es $S = 2 * DevStd(R)$ donde $DevStd$ es la operación de desviación estándar y

$$\delta = \begin{cases} +1 & \text{si se incrusta un bit 1} \\ -1 & \text{si se incrusta un bit 0} \end{cases}$$

Como se puede apreciar, la ecuación 4.1 es similar a la ecuación 3.1, pero sufre la modificación de que se extrae una versión reducida del bloque dominio en lugar de tomar la normalización del mismo. La selección del conjunto de bloques rango R es llevada a cabo buscando el bloque que mejor aproxime al bloque \hat{R} , la métrica de distorsión utilizada es el cálculo del error RMSE usando la ecuación 2.5.

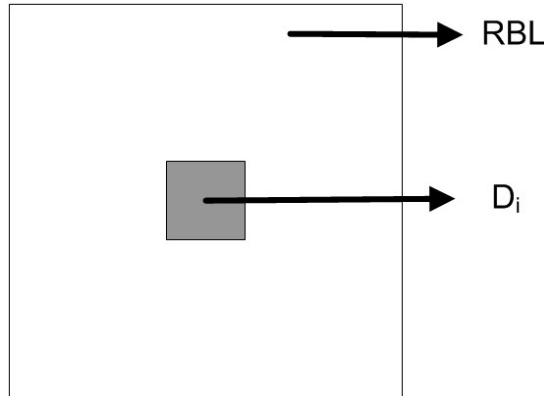


Figura 4.7: Región de búsqueda local utilizada en el esquema propuesto.

La figura 4.7 muestra la región de búsqueda local del bloque dominio central utilizada por este esquema. La figura 4.8 contiene la RBL con $N_{RBL} = 11$ para el bloque dominio D_9 ubicado en el centro de la misma y el bloque

rango R_9 encontrado. La figura 4.9 a) muestra el bloque dominio D_9 de la figura 4.8, la figura 4.9 b) muestra el bloque R_9 que se modificará para la incrustación de la marca y la figura 4.9 c) muestra el bloque rango modificado \hat{R}_9 para la incrustación de un bit 1. El error RMSE mínimo entre R y \hat{R}_i es 2.0156. Por último la figura 4.10 muestra los bloques rango seleccionados (bloques blancos) para cada bloque dominio (bloques negros).

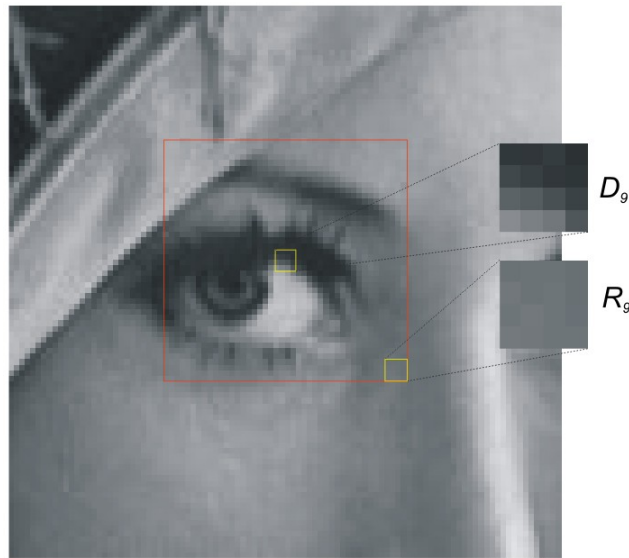


Figura 4.8: Región de búsqueda local asociada al bloque dominio D_9 y su respectivo bloque rango R_9 .

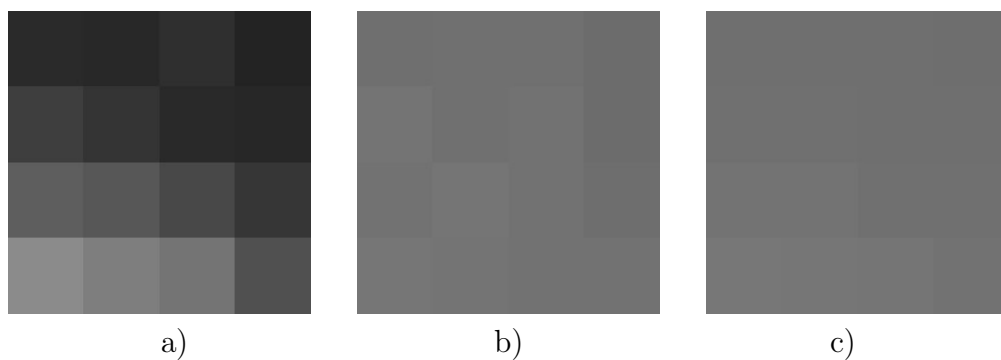


Figura 4.9: a)Bloque dominio D_9 , b)Bloque rango R_9 y c)Bloque rango modificado \hat{R}_9 con $\delta = 1$. Error $RMSE=2.0156$.

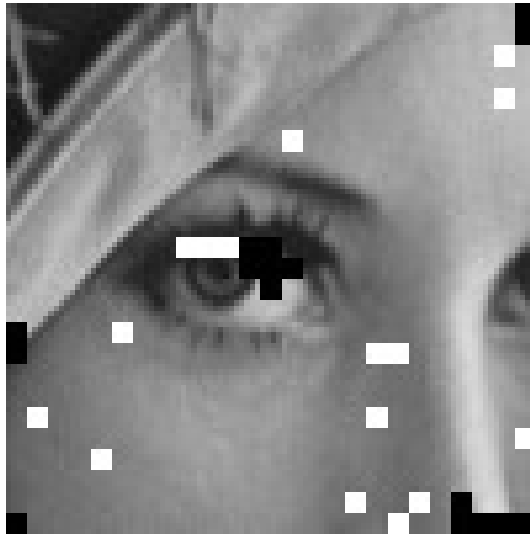


Figura 4.10: Bloques rango (bloques blancos) y bloques dominio (bloques negros) seleccionados en RI_1 .

4. INCRUSTACIÓN DE LA MARCA. *Teniendo el conjunto bloques rango R se procede a la incrustación de la marca, para ello cada bloque rango $R_i \in R$ se sustituye por \hat{R}_i usando la ecuación 4.1 tomando su respectivo bloque dominio $D_i \in D$ y su respectivo bit i de la marca M .*

La figura 4.11 a) muestra la región de incrustación RI_1 marcada, el PSNR entre la región de incrustación inicial y la marcada es 55.11 dB, el cual indica perceptibilidad nula o mínima de la marca. Cabe mencionar que cuando el medio marcado tiene valores PSNR menores de 40 dB se puede notar ligeramente la distorsión generada por la incrustación de la marca, pero hay que tomar a consideración que es necesario tener la imagen original.

El proceso de incrustación de la marca se lleva a cabo para cada región de incrustación RI_i , el número de regiones depende de las características de la imagen. En el ejemplo anterior se encontraron ocho regiones de incrustación. La figura 4.11 b) muestra la imagen final marcada, el PSNR entre la imagen portadora y marcada es 60.41 dB.



Figura 4.11: a)Imagen de RI_1 marcada con 16 bits, $PSNR=55.11$ dB. b)Imagen marcada con 16 bits en 8 RI , $PSNR=60.41$ dB.

4.3. Proceso de extracción y detección de la marca

El proceso de extracción es similar al proceso de incrustación de la marca, también se realiza en dos pasos: el primero selecciona las regiones de incrustación y el segundo extrae y detecta la marca de cada una de ellas. Teniendo las marcas extraídas y detectadas de cada región de incrustación posteriormente se utiliza una política para determinar la magnitud de la marca final extraída del sistema.

La selección de las regiones de incrustación RI es llevada a cabo de la misma manera que en el proceso de incrustación, tomando en cuenta las mismas condiciones, tales como los valores de N_b , n , así como las mismas restricciones en el detector de Harris.

El proceso para extraer la marca de una región de incrustación RI necesita de la magnitud de n y N_{RBL} , así como las restricciones del detector de Harris. El proceso de detección de la marca necesita los índices de los bloques rango seleccionados en el proceso de incrustación para poder determinar la existencia de cada bit.

Los primeros dos pasos llevados a cabo para realizar la extracción son los mismos que los de la incrustación de la marca, por lo que no serán descritos a

detalle, sin embargo los siguientes tres serán descritos con detenimiento.

Después de la selección de las RI , la extracción y detección de la marca de cada una toma los siguiente pasos:

1. PARTICIONAMIENTO DE LA REGIÓN DE INCRUSTACIÓN. *La región de incrustación RI se divide en $N_B \times N_B$ bloques no traslapantes de tamaño $n \times n$.*
2. SELECCIÓN DEL CONJUNTO DE BLOQUES DOMINIO. *El conjunto de bloques dominio D se forma en dos pasos: detección y decimación.*
 - a) DETECCIÓN DE BLOQUES DOMINIO.
 - b) DECIMACIÓN DE BLOQUES DOMINIO.
3. SELECCIÓN DEL CONJUNTO DE BLOQUES RANGO. *Para cada bloque dominio D_i de D se construye una RBL de tamaño $N_{RBL} \times N_{RBL}$ bloques no traslapantes B_i , en donde se buscará por el bloque rango R_i en el cual se incrustó el bit i de la marca. Para ello se modificará cada uno de los bloques rango $R \in RBL$ mediante la ecuación 4.1 con $\delta = +1$ y $\delta = -1$. Luego se seleccionará el bloque modificado \hat{R} que mejor se aproxime al bloque rango R que lo genera, usando la misma métrica que en el proceso de incrustación: el error RMSE. La posición del bloque seleccionado \hat{R} indica la posición bloque rango R_i en el cual se incrustó un bit de información, además se necesita almacenar el valor de δ que permite al \hat{R} seleccionado tener el menor error con respecto a R .*
4. EXTRACCIÓN DE LA MARCA. *Teniendo el conjunto de bloques rango R y el valor δ de cada uno de ellos, se procede con la extracción de la marca. Este proceso es llevado a cabo tomando en cuenta el signo de δ_i de cada bloque rango R_i . Si δ_i es positivo se extrae un bit $m_i = 1$, si δ_i es negativo se extrae un bit $m_i = 0$. Formando la marca $M = m_1m_2m_3\dots m_{M_n}$.*

5. DETECCIÓN DE LA MARCA. *Cada uno de los bits de la marca será comprobado, para ello se necesita la posición del bloque rango del cual se extrajo, dicha posición se busca en el vector de índices de bloques rango generado en el proceso de incrustación, si la posición del bloque rango se encuentra en el vector indica que el bit fue detectado, si no se encuentra el bit extraído no se detectó correctamente, esto indica que el bit extraído es con seguridad un falso positivo.*

Las marcas extraídas de cada RI_i están formadas de bits extraídos correctamente, bits no detectados y posiblemente bits falsos positivos. Tomando en cuenta estas posibilidades se necesita una política de selección de bits para formar la marca final extraída. Para ello se optó por utilizar, además de las marcas extraídas, el error RMSE E_{RMSE} de la aproximación de cada bloque rango R con el bloque modificado \hat{R} de donde se extrajo cada bit. Para la selección de un bit b de la marca M final, se toman cada uno de los bits con posición b de las marcas extraídas de cada RI y se utiliza la siguiente política:

1. La magnitud del bit b de la marca M es tomada de la magnitud del bit con mayor porcentaje de apariciones en las marcas extraídas de las RI .
2. Si los porcentajes de apariciones son los mismos, es decir que el número de bits unos y ceros son iguales, se hace un promedio de los errores E_{RMSE} de cada conjunto de bits (ceros y unos). La magnitud del bit b de la marca M se toma del conjunto con menor error promedio. Por ejemplo si el error del conjunto de ceros es menor, entonces el bit b tiene magnitud 0.

Con la imagen marcada obtenida por el ejemplo llevado a cabo en la sección anterior (proceso de incrustación de la marca) se procede a realizar el proceso de extracción de la marca, en él se encontraron ocho regiones de incrustación, de las cuales se obtuvieron los siguientes resultados:

Región	Marca extraída															
RI_1	1	0	1	0	1	1	-1	0	1	-1	1	0	1	0	1	0
RI_2	1	0	1	0	1	0	1	0	1	0	-1	0	1	0	1	0
RI_3	0	0	1	1	-1	0	1	-1	1	0	1	1	0	0	-1	0
RI_4	1	0	0	0	1	0	1	-1	0	0	0	-1	-1	0	0	1
RI_5	-1	0	1	0	1	0	1	0	1	0	1	0	1	0	-1	0
RI_6	1	1	1	0	-1	0	-1	0	1	0	1	0	1	0	1	0
RI_7	-1	0	1	0	1	0	1	-1	1	0	1	0	1	0	1	-1
RI_8	1	0	1	0	1	1	1	1	1	0	1	1	1	-1	1	-1
Marca Final	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

Después de la política de selección de bits la marca final extraída es: $M = 1010101010101010$. Cabe mencionar que los elementos etiquetados con -1 son los bits no detectados.

De manera general, el proceso de incrustación descrito en la sección 4.2 es llevado a cabo como se ejemplifica en el siguiente pseudocódigo:

1. `ProcesoIncrustación(ImgPortadora,Marca)`
2. `ImgMarcada=ImgPortadora`
3. `RI=SelecciónRI(ImgPortadora)`
4. Para cada k de RI hacer
5. `ImgPuntosRI=Harris(RI[k])`
6. `ImgPuntosRIParticionada=Particionar(ImgPuntosRI,TamBloques)`
7. `BDDetectados=DetectarBD(ImgPuntosRIParticionada)`
8. `BD=DecimarBD(BDDetectados,TamMarca)`
9. Para cada i de BD hacer
10. `RBL=SeleccionarRBL(BD[i])`
11. Para cada j de RBL hacer
12. `BRModificado[j]=IncrustarEnBloque(BD[i],RBL[j],Marca[i])`
13. `ErrorBR[j]=PSNR(BRModificado[j],BD[i])`

```
14.   fin Para
15.   BRSeleccionado=min(ErrorBR)
16.   SustituirBR(ImgMarcada,BRSeleccionado)
17.   Llave[k][i]=PosBRSeleccionado;
18.   fin Para
19. fin Para
20. retornar ImgMarcada,Llave
21. fin proceso
```

Se puede observar que en la línea 3 se lleva a cabo el proceso de selección de regiones de incrustación descrito al inicio de la sección 4.2. En las líneas 4-19 se lleva a cabo el proceso de incrustación de la marca en cada una de las regiones seleccionadas. En las líneas 5-8 se seleccionan los bloques dominio, en las líneas 10-14 se busca el bloque rango en la región de búsqueda local, en las líneas 15-16 se selecciona el bloque rango y se sustituye en la imagen marcada, cabe mencionar que dicho bloque rango ya fue marcado en la línea 12. En la línea 17 se guarda la posición del bloque rango y finalmente en la línea 20 se retorna la imagen marcada y la llave.

Por otra parte, el proceso de extracción descrito en la sección 4.3 tiene como pseudocódigo:

```
1. ProcesoExtracción(ImgMarcada,Llave)
2.  RI=SelecciónRI(ImgMarcada)
3.  Para cada k de RI hacer
4.    ImgPuntosRI=Harris(RI[k])
5.    ImgPuntosRIParticionada=Particionar(ImgPuntosRI,TamBloques)
6.    BDDetectados=DetectarBD(ImgPuntosRIParticionada)
7.    BD=DecimarBD(BDDetectados,TamMarca)
8.    Para cada i de BD hacer
9.      RBL=SeleccionarRBL(BD[i])
```

```
10. Para cada j de RBL hacer
11.   BRModificado0[j]=IncrustarEnBloque(BD[i],RBL[j],0)
12.   BRModificado1[j]=IncrustarEnBloque(BD[i],RBL[j],1)
13.   ErrorBR0=PSNR(BRModificado0[j],BD[i])
14.   ErrorBR1=PSNR(BRModificado1[j],BD[i])
15. fin Para
16. PosBRSeleccionado0=min(ErrorBR0)
17. PosBRSeleccionado1=min(ErrorBR1)
18. si (ErrorBR0[PosBRSeleccionado0]<ErrorBR1[PosBRSeleccionado1])
19.   Marca[i]=0
20.   PosBR=PosBRSeleccionado0
21. sino
22.   Marca[i]=1
23.   PosBR=PosBRSeleccionado1
24. fin si
25. IsDetectado=Detectar(Llave,PosBR)
26. si (IsDetectado==0)
27.   Marca[i]=-1
28. fin si
29. fin Para
30. fin Para
31. retornar Marca
32. fin proceso
```

El pseudocódigo del proceso de extracción es similar al del proceso de incrustación, las diferencias entre ellos son notables en las líneas 10-15, ya que en ellas se lleva a cabo la búsqueda del bloque rango en la RBL, para ello se necesita incrustar temporalmente un bit 1 y un bit 0. El bloque que minimice el error PSNR entre el bloque dominio y el bloque rango marcado determina la magnitud del bit extraído (líneas 18-24). El proceso de detección es llevado a cabo en las líneas 25-28. Finalmente la marca extraída es retornada en la línea 31.

4.4. Discusión del esquema propuesto

El esquema propuesto descrito anteriormente tiene varias diferencias con respecto al esquema en el que fue basado [BCD98a, BCD98b], entre las que se encuentran:

1. La utilización de regiones de búsqueda local para limitar las búsquedas de bloques rango.
2. La utilización de regiones de incrustación para ocultar la marca redundantemente.
3. La utilización del detector de puntos de Harris para seleccionar los bloques dominio y las regiones de incrustación.
4. La modificación de la ecuación de incrustación base mediante el uso de una versión reducida del bloque dominio D y el cálculo automático de la magnitud de la marca S mediante el uso de la operación de desviación estándar del bloque rango. Con ello se reduce el impacto perceptual de los bloques marcados.
5. La modificación del proceso de la selección de bloques rango.

Con estas modificaciones se busca que el nuevo esquema mejore la robustez ante ataques JPEG y disminuya el impacto perceptual al incrustar la marca. Este nuevo esquema debe ser utilizado en imágenes poco homogéneas, debido a la forma de seleccionar los bloques dominio. Sin embargo, con respecto al esquema base el costo computacional aumenta principalmente debido a la utilización de la detección de puntos característicos (ver sección 5.3). Al igual que el esquema base, este esquema utiliza una llave (información de posiciones de bloques rango) para la detección de la marca, el tamaño de tal llave es $M_n \times 2 \lceil \log_2(N_B) \rceil$ bits por cada

región de incrustación, donde M_n es el número de bits de la marca incrustada y N_B es el número bloques de un lado de la región de incrustación.

Capítulo 5

Implementación y resultados

En este capítulo se describen los resultados de las pruebas realizadas al esquema de marcas de agua propuesto. Las pruebas son principalmente para evaluar la perceptibilidad, la robustez y la capacidad de incrustación de información. Al final del capítulo se discuten los resultados obtenidos en comparación con los resultados de los otros esquemas mencionados anteriormente.

5.1. Implementación

El esquema de marcas de agua propuesto fue implementado en matlab 7.4.0 (R2007a), tal herramienta de cómputo es una de las más utilizadas para realizar las primeras aproximaciones de un sistema computacional, ya que el manejo de muchos procesos son incluidos como funciones predefinidas. Para el completo entendimiento del esquema programado se redujo el uso de funciones predefinidas, permitiendo una fácil migración a otra herramienta o lenguaje de programación.

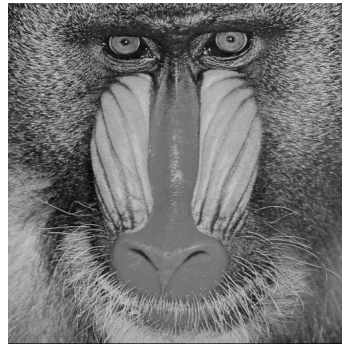
5.2. Pruebas realizadas

5.2.1. Imágenes utilizadas

Los experimentos realizados fueron hechos en las imágenes de la figura 5.1. Tales imágenes son comúnmente utilizadas como banco de pruebas para el procesamiento de imágenes, destacando a las imágenes de Lenna y Baboon. Las imágenes utilizadas tienen un tamaño de 512×512 píxeles en escala de grises con una profundidad de 8 bits.



Lenna



Baboon



Goldhill



Barbara



Peppers

Figura 5.1: Imágenes utilizadas en los experimentos.

5.2.2. Experimentos realizados

Los experimentos llevados a cabo se dividen en dos partes, la primera analiza el comportamiento del esquema en general, esto es con el fin de encontrar los

parámetros aptos para su buen funcionamiento. La segunda parte es para analizar la robustez del esquema en contra de ataques tales como: compresión JPEG, filtrado pasa bajas y ruido gaussiano.

5.2.2.1. Comportamiento del esquema

El comportamiento del esquema depende de la modificación de sus tres principales parámetros:

1. Tamaño de los bloques. *El tamaño de los bloques es $n \times n$ píxeles.*
2. Tamaño de la Región de búsqueda local. *La RBL tiene un tamaño de $N_{RBL} \times N_{RBL}$ bloques.*
3. Tamaño de la región de incrustación. *La RI es de tamaño $N_b \times N_b$ bloques.*

En el primer experimento se modificaron el tamaño de los bloques y el tamaño de la región de búsqueda local, el comportamiento del esquema se muestra en las gráficas de la figura 5.2, para ello se utilizó una región de incrustación de tamaño 25×25 bloques ($N_b = 25$) y una marca de 32 bits. Los resultados obtenidos mostrados en la gráfica de la figura 5.2 a) muestran que a medida que el tamaño de los bloques aumenta, el PSNR disminuye, esto es debido a que la distorsión generada por la sustitución de bloques es mayor con bloques grandes, con lo cual se recomienda que los bloques sean de tamaño pequeño para que no se afecte a la perceptibilidad en la imagen marcada. En la misma gráfica se observa el comportamiento con distintos tamaños de la RBL, a medida que el tamaño de la RBL crece la perceptibilidad disminuye. De la gráfica de la figura 5.2 b) se puede concluir que el esquema con bloques de tamaño 4×4 píxeles y RBL de tamaño 19×19 bloques muestra una mejor precisión. La precisión es la relación de la marca extraída con la marca incrustada, es decir, indica el porcentaje de éxito al extraer la marca. En c) se muestra que el número de las regiones de incrustación incrementa a medida que los bloques disminuyen su tamaño.

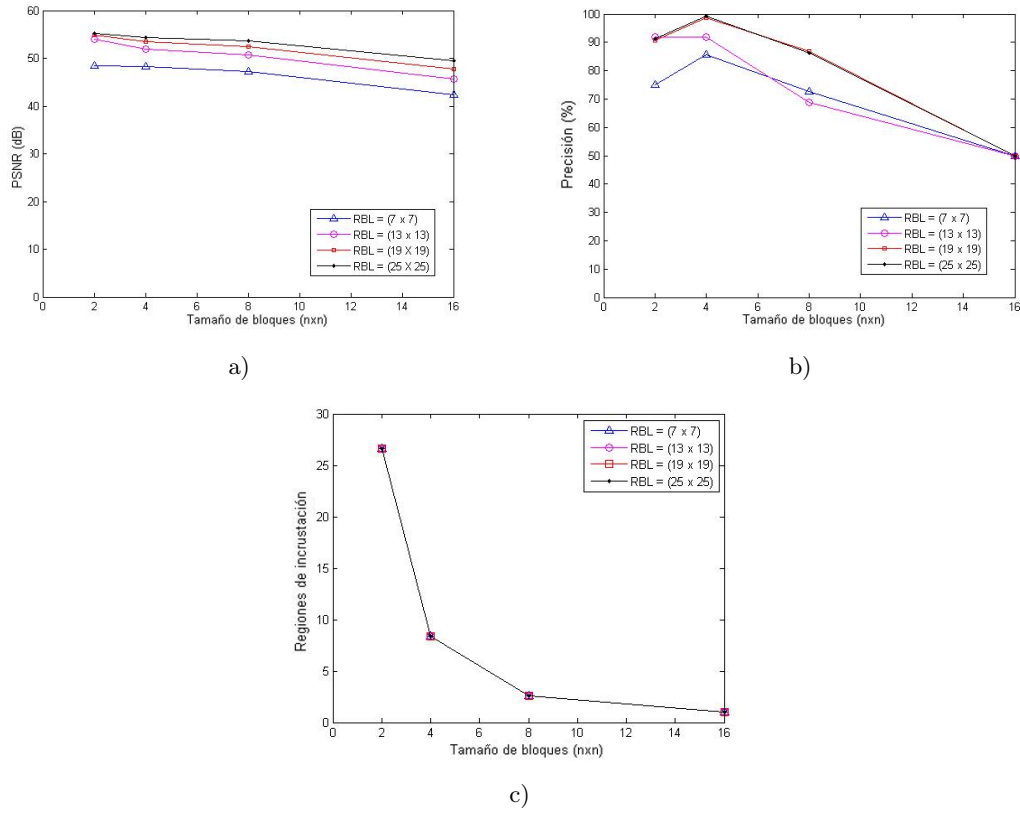


Figura 5.2: a) Bloques rango contra PSNR, b) Bloques rango contra precisión y c) Bloques rango contra regiones de incrustación.

Como se mostró anteriormente, el tamaño de los bloques y el tamaño de la región búsqueda local influyen mucho en la perceptibilidad y en la precisión. El objetivo del experimento anterior fue el encontrar los valores de n y N_{RBL} con los cuales el esquema se comporta mejor, obteniendo 4 y 19 respectivamente.

Por último se muestra la influencia del tamaño de la región de incrustación en el comportamiento del esquema, para ello se utilizaron bloques de tamaño 4×4 píxeles, RBL de tamaño 19×19 bloques y una marca de 32 bits. La figura 5.3a indica que el tamaño de las regiones de incrustación influyen en la perceptibilidad de la marca, de manera que a mayor tamaño menor distorsión. Tomando en cuenta la influencia con la precisión (figura 5.3b), se encuentra que la región de incrustación tamaño 29×29 bloques tiene un mejor desempeño, puesto que pre-

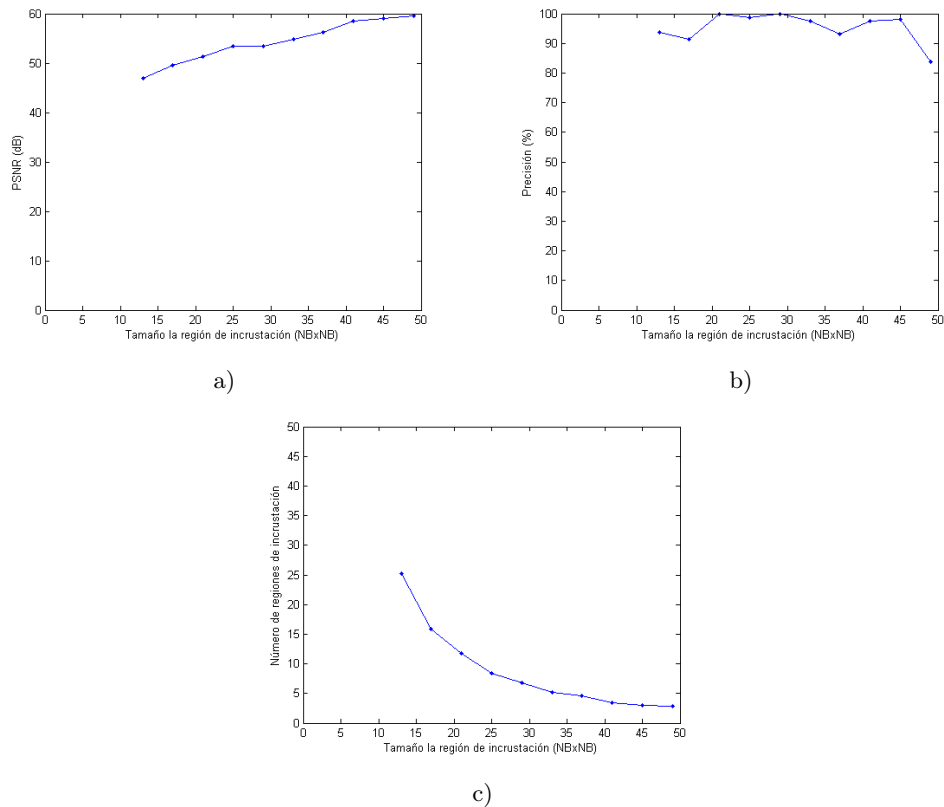


Figura 5.3: a) Tamaño de la región de incrustación contra PSNR, b) Tamaño de la región de incrustación contra precisión y c) Tamaño de la región de incrustación contra número de regiones de incrustación.

senta una menor perceptibilidad y tiene menor número de regiones de incrustación (figura 5.3c).

Un experimento adicional muestra la relación del tamaño de las regiones de incrustación con el número promedio y máximo de bloques dominio detectados (figura 5.4), éstos últimos influyen en el tamaño de la marca a incrustar. Aquí se muestra que el tamaño de la región de incrustación influye en la capacidad del esquema. Cabe mencionar que los resultados obtenidos por esta prueba dependen de las imágenes utilizadas, puesto que son imágenes poco homogéneas, lo cual influye en la cantidad de puntos detectados.

Al final de estos experimentos, los parámetros con los que el esquema encuentra mejores resultados son: bloques de tamaño 4×4 píxeles, RBL de tamaño 19×19

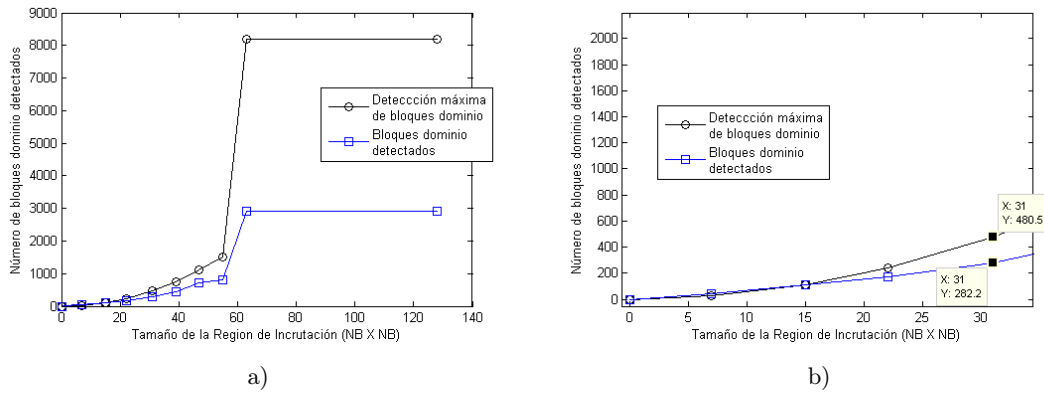


Figura 5.4: a) Tamaño de la región de incrustación contra bloques dominio detectados y b) Acercamiento de a).

bloques y RI de tamaño 29×29 bloques. Ahora se mostrará el comportamiento con respecto al tamaño de la marca, para ello se utilizó una marca de la forma $(01)^*$. La figura 5.5 muestra los resultados obtenidos. Se puede observar que el esquema funciona con marcas pequeñas de hasta 34 bits, sin olvidar que ésta se incrusta en promedio 6.8 veces en la imagen portadora.

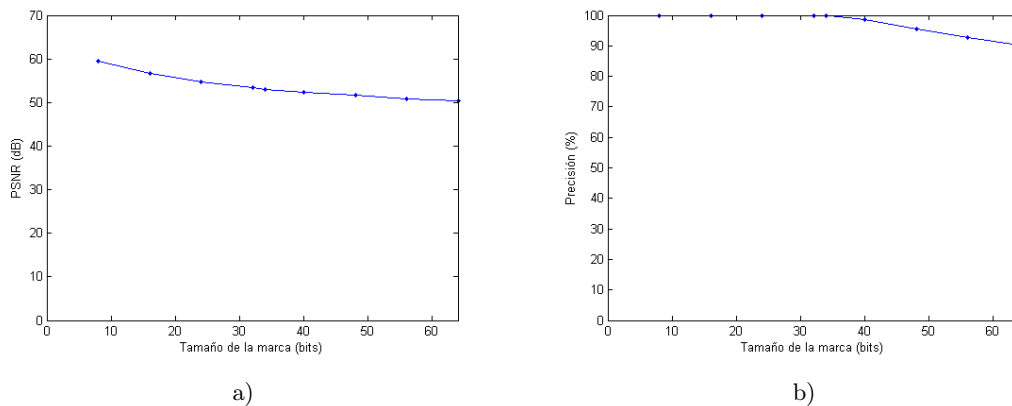


Figura 5.5: a) Bits incrustados contra PSNR y b) Bits incrustados contra precisión.

Como se ha mencionado, el esquema propuesto utiliza una llave para poder *detectar* la marca, ésta guarda las referencias de los bloques rango en donde se incrustó la marca, su tamaño está en función del número de bits de la marca, del tamaño y de la cantidad de regiones de incrustación. La siguiente prueba

muestra el comportamiento del esquema ante la ausencia de la llave, lo cual deja al esquema sin la capacidad de decidir si cada uno de los bits extraídos son válidos. En la figura 5.6 se muestra que la detección de bloques es un proceso importante para la precisión del esquema, puesto que sin él ésta disminuye. Sin embargo, puede ser utilizado con marcas muy pequeñas.

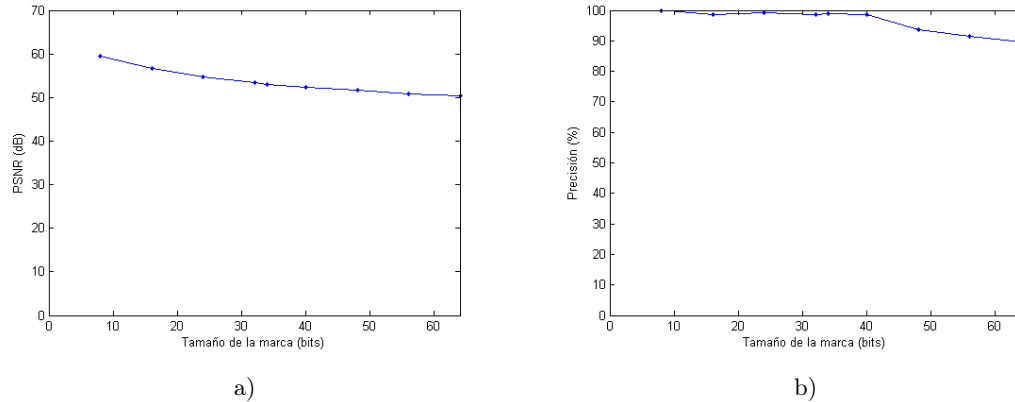


Figura 5.6: Esquema sin detector. a) Bits incrustados contra PSNR y b) Bits incrustados contra precisión.

5.2.2.2. Comportamiento del esquema ante ataques

La compresión JPEG es el principal ataque al que son sometidas las imágenes, por lo que es importante que los esquemas de marcas de agua sean robustos a él. La figura 5.7 muestra los resultados obtenidos del esquema ante tal ataque, en la gráfica 5.7a se muestra el comportamiento del esquema con bloques de 4×4 píxeles, RBL de 19×19 bloques y RI de tamaño 29×29 bloques, se puede observar que el comportamiento se ve mejorado en la gráfica 5.7b, puesto que se utilizaron bloques de tamaño 8×8 , en ambos casos se incrustó una marca de 34 bits. La consideración de usar bloques más grandes fue tomada en cuenta puesto que la compresión JPEG utiliza bloques de 8×8 píxeles dentro del proceso de compresión.

Un ataque adicional al que fue sometido el esquema propuesto es el añadir

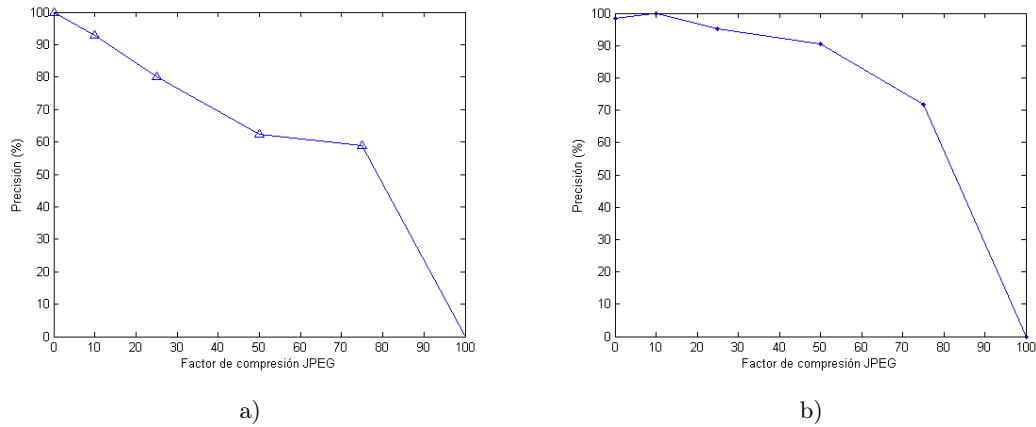


Figura 5.7: a) Precisión del esquema propuesto ante la compresión JPEG con bloques 4×4 y b) bloques 8×8 .

ruido gaussiano al medio marcado, para ello se utilizó una media de 0 y una varianza acotada por $[0.0001-0.01]$, como se muestra en la figura 5.8. La precisión del esquema ante este ataque no fue muy buena, puesto que este ataque distorsiona mucho a la imagen marcada. En la figura 5.9 se muestran las imágenes marcadas (de la sección 5.2.1) y las mismas con ruido gaussiano con media de 0 y varianza de 0.01, el PSNR promedio es 20.12 dB. En este caso y al igual que en el experimento anterior se utilizaron bloques de tamaño 4×4 píxeles, RBL de 19×19 bloques y RI de 29×29 bloques, cuando se usaron bloques de 8×8 píxeles la RBL es de 13×13 bloques y RI de 21×21 bloques.

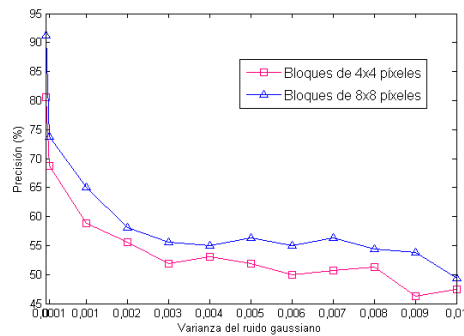


Figura 5.8: Resultados ante el ataque de ruido gaussiano

Imágenes Marcadas



Adición de ruido gaussiano

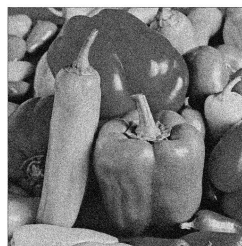
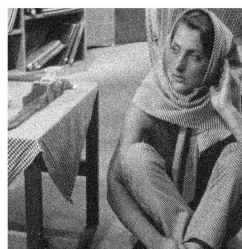
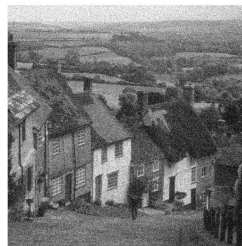
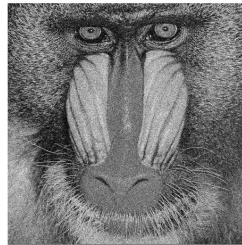
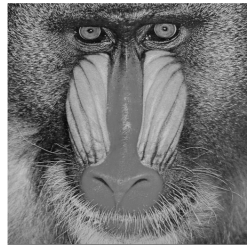


Figura 5.9: Imágenes marcadas con 34 bits y la adición de ruido gaussiano.

La precisión del esquema ante el filtrado pasa bajas gaussiano usando una ventana de 3×3 y una desviación estándar de 0.5 es del 71 % usando bloques de 4×4 píxeles y de 73 % usando bloques de 8×8 píxeles, el PSNR promedio es de 37.5 dB.

5.2.3. Comparación con otros esquemas

La investigación desarrollada en esta tesis está inspirada en el esquema de Patrick Bas *et-al* [BCD98a, BCD98b]. En este nuevo esquema se lograron obtener mejoras que lo hacen un mejor método, puesto que redujo el impacto perceptual de la imagen marcada y al mismo tiempo aumentó la robustez en contra de ataques JPEG. La figura 5.10 muestra tal mejora al incrustar una marca de 34 bits.

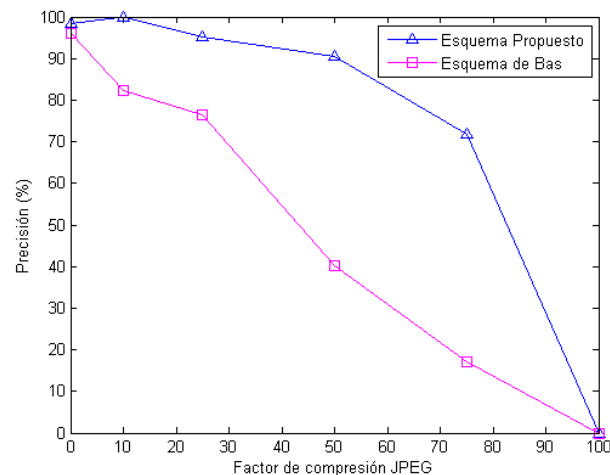


Figura 5.10: Comparación del esquema propuesto con el esquema de Patrick Bas ante JPEG incrustando una marca de 34 bits.

Como se ha mencionado, la causa del alto impacto perceptual de la imagen marcada es la sustitución de bloques de imagen. Con respecto a otros algoritmos inspirados por la codificación fractal, el esquema propuesto tiene una degradación de la imagen marcada casi imperceptible.

En las figuras 5.11, 5.12 y 5.13 se muestran la comparaciones del esquema base con los esquemas de Puate [PJ96], Li [GYB02] y Kamal [Gul03] ante el ataque de

compresión JPEG, para ello se igualaron las condiciones con las que reportaron tales resultados. En el trabajo de Zhen Yao [Yao03] se reportó únicamente la incrustación de 8 bits, puesto que tal esquema tiene muy alto costo computacional. El nuevo esquema oculta sin problemas esos 8 bits.

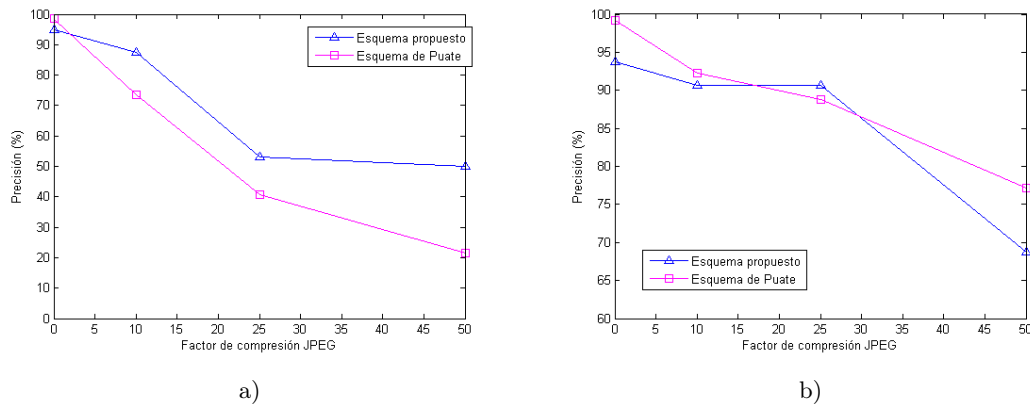


Figura 5.11: Comparación del esquema propuesto con el esquema de Puate ante la compresión JPEG. Marca de 32 bits. a)Bloques de 4×4 píxeles y b)Bloques de 8×8 píxeles.

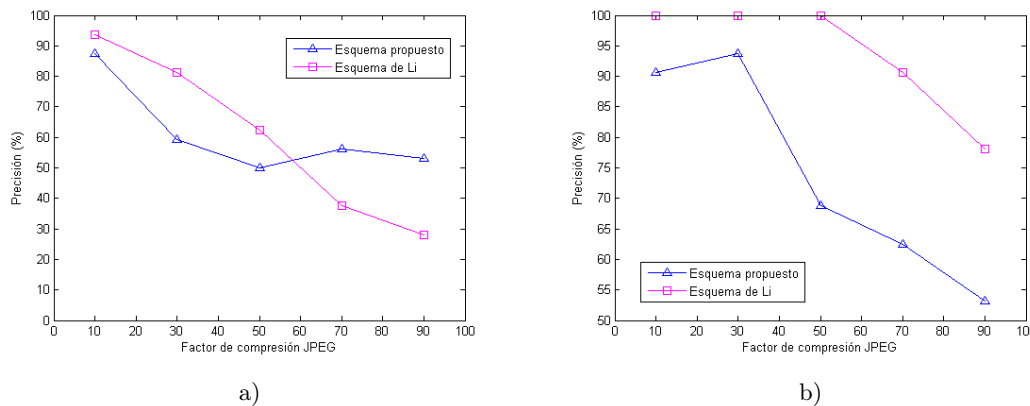


Figura 5.12: Comparación del esquema propuesto con el esquema de Li ante la compresión JPEG. Marca de 32 bits. a)Bloques de 4×4 píxeles y b)Bloques de 8×8 píxeles.

La comparación en cuanto al impacto perceptual se muestra en la tabla 5.1, en ella se puede apreciar que el esquema propuesto supera a los otros esquemas, de ellos a algunos considerablemente. La distorsión generada en la estegoimagen

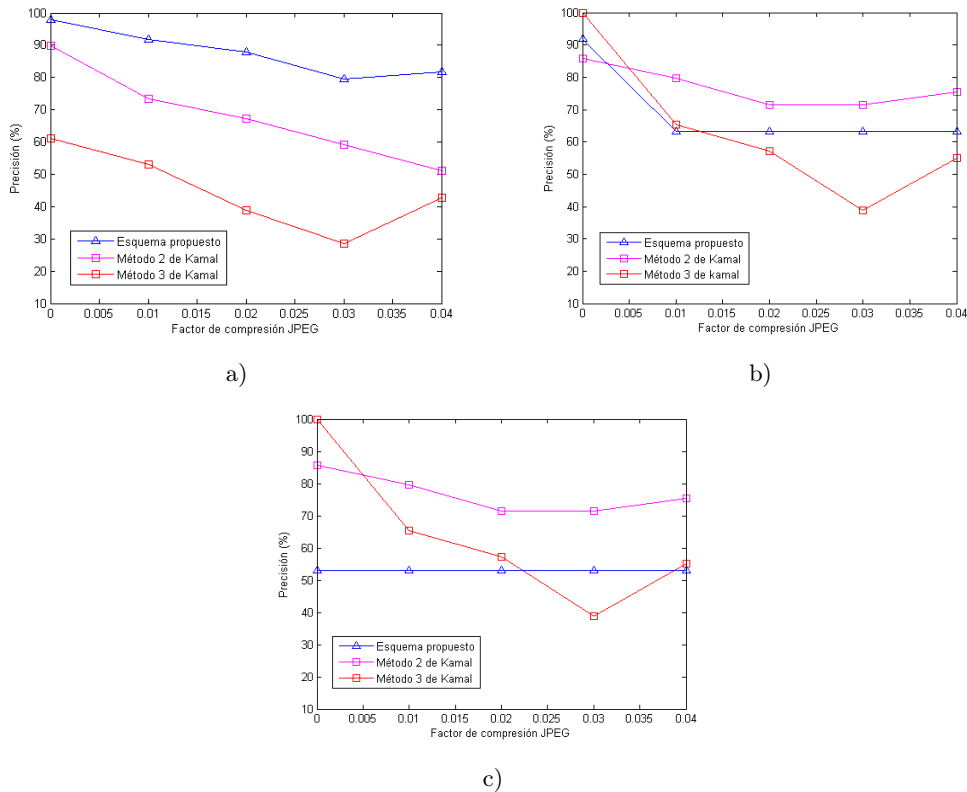


Figura 5.13: Comparación del esquema propuesto con el esquema de Kamal ante la compresión JPEG. Marca de 49 bits. a)Bloques de 4×4 píxeles, b)Bloques de 8×8 píxeles y c)Bloques de 16×16 píxeles.

por cada uno de los esquemas (columna 1) con los cuales se compara el esquema propuesto corresponde a diferentes condiciones (columna 3), por lo cual la segunda columna muestra la distorsión del esquema propuesto en esas mismas condiciones. La medida de distorsión utilizada es PSNR (ecuación 2.7) para cuatro esquemas y SNR (ecuación 2.6) para dos, cabe mencionar que la entre mayor sea la magnitud de estas métricas menor es el impacto perceptual en el medio marcado. El tamaño de los bloques rango es de 4×4 píxeles para todos los casos. Al igual que las pruebas anteriores se igualaron condiciones para que la comparación sea justa.

Los resultados obtenidos de la comparación del esquema propuesto con los otros esquemas muestran que éste se comporta mejor con bloques 4×4 . El nuevo esquema se ve superado por el esquema de Li . Tales resultados no reflejan del

Distorsión por esquema	Distorsión esquema propuesto	Marca (bits)
Puate <i>et-al</i> : 31.5 dB (PSNR)	55.76 dB (PSNR)	32
Li <i>et-al</i> : 30.05 dB (PSNR)	55.76 dB (PSNR)	32
Zhen : 32.87 dB (PSNR)	62.01 dB (PSNR)	8
Kamal-II : 52.81 dB (SNR)	53.66 dB (SNR)	49
Kamal-III : 52.81 dB (SNR)	53.66 dB (SNR)	49
Bas <i>et-al</i> : 39.75 dB (PSNR)	52.98 dB (PSNR)	34

Tabla 5.1: Comparación de la distorsión generada por cada esquema contra el esquema propuesto.

todo la capacidad del esquema propuesto, ya que éste funciona mejor cuando en la imagen portadora se encuentran varias regiones de incrustación y con las imágenes utilizadas se encontraron a lo más dos regiones de incrustación usando bloques de tamaño 8×8 y 16×16 . En todas estas pruebas comparativas se utilizaron imágenes de 256×256 píxeles (exceptuando la de Bas que fue realizada con imágenes de 512×512 píxeles). como se ha mencionado, la comparativa con estos esquemas se hizo igualando las condiciones con las que sus autores reportaron los resultados. En el caso de la comparativa con Li las imágenes usadas son de 256×256 píxeles y los bloques son de tamaño 8×8 y 16×16 , lo que ocasiona que el algoritmo propuesto encuentre a lo mas 2 regiones de incrustación, limitando la precisión del mismo.

5.3. Discusión de los resultados obtenidos

En este capítulo se mostró que el comportamiento del esquema propuesto depende del ajuste de sus tres principales parámetros y que cada uno de ellos influye en los otros dos. Su comportamiento ante el ataque de compresión JPEG es bueno y mejora en mucho al esquema base. tomando en cuenta la distorsión generada en la imagen marcada, el algoritmo aquí propuesto supera a los otros esquema con los que fue comparado.

Se puede notar que el esquema propuesto es más robusto que el esquema del

cual fue basado [BCD98a, BCD98b], sin embargo el costo computacional aumenta debido al uso de puntos característicos para la selección de las regiones de incrustación y del conjunto de bloques dominio, lo que no ocurre en el esquema del cuál fue basado el algoritmo propuesto. La tarea de buscar los bloques rango a partir de los bloques dominio seleccionados es más simple puesto que se limita a una región de búsqueda local, la cual es 35 % menor que el tamaño de una región de incrustación (usando región de búsqueda local de 19×19 bloques y regiones de incrustación de 29×29 bloques). Además, en las pruebas realizadas (figura 5.2c) el número promedio de regiones de incrustación es menor que la razón del tamaño de las regiones de incrustación con respecto a la imagen original. En comparación con los esquemas de marcas de agua que usan compresión fractal (mencionados en la sección), el esquema aquí propuesto tiene un mejor desempeño, ya que la compresión fractal es un proceso muy costoso (ver sección 2.2.5.3) y aunado a esto se le añade el costo de la incrustación de la marca. El esquema propuesto por Kamal [Gul03], que no usa compresión fractal, es menos costoso que el aquí propuesto ya que hace una búsqueda de bloques rango en una región que ocupa la cuarta parte de la imagen, además este esquema no detecta al conjunto de bloques dominio porque es parte de una región predefinida. Sin embargo, no en todos los casos supera a nuestro esquema, como en impacto perceptual, ya que el nuevo esquema distorsiona menos a la imagen marcada. Por lo anterior y los resultados obtenidos, se puede notar que el esquema propuesto en este trabajo de tesis presenta un buen compromiso entre costo computacional, robustez e impacto perceptual

Capítulo 6

Conclusiones y trabajo futuro

En este capítulo se describen las conclusiones a las que se llegó en esta investigación, así como las posibles modificaciones o extensiones futuras que se pueden hacer al esquema propuesto.

6.1. Revisión de objetivos

La investigación de esta tesis trata de la generación de un nuevo algoritmo de marcas de agua usando similitudes, el cual es inspirado por la codificación fractal de imágenes.

Recordando que el objetivo general es: *Crear un algoritmo de marcas de agua robusto para imágenes en escala de grises utilizando código fractal, similitudes y puntos característicos.*

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

1. Utiliza imágenes en escala de grises con una profundidad de 8 bits.
2. Utiliza similitudes de bloques (un bloque rango mapea a un bloque dominio).
3. Utiliza un detector de puntos característicos (detector de Harris) para la selección de bloques dominio y regiones de incrustación.

4. Utiliza el código fractal para ocultar información. El código fractal se refiere al hecho de mapear un bloque con otro mediante una aproximación, la forma de incrustar un bit de información es llevada a cabo por la sustitución de un bloque rango por un bloque rango modificado.
5. Es robusto a compresión JPEG y filtrado pasa bajas.

De los puntos descritos anteriormente se logró crear un nuevo esquema más robusto que el esquema base y que los inspirados por la codificación fractal (en cierta medida) sin perder la esencia de la búsqueda de similitudes y la sustitución de bloques.

Por otra parte, los objetivos secundarios se alcanzaron, ya que el nuevo esquema:

1. Aumentó la robustez en contra de la compresión JPEG, a cambio de esto la capacidad se disminuyó en 16 bits (nuevo esquema: 34 bits, esquema de Bas: 50 bits) puesto que se aplica una redundancia en un factor de 6.8 aproximadamente, es decir se ocultan un total de 217 bits brutos.
2. Reduce considerablemente el ataque perceptual a la imagen marcada con respecto a la métrica de distorsión PSNR, puesto que el algoritmo base promedió un PSNR de 39.75 dB y el esquema aquí propuesto promedió un PSNR de 52.98 dB.

6.2. Conclusiones

El esquema de marcas de agua creado está basado en la investigación de Bas *et-al* [BCD98a, BCD98b], sin embargo la forma de incrustar la marca en la imagen portadora sufrió algunos cambios, de los cuales se concluye lo siguiente:

1. La imperceptibilidad de la marca se mejoró sustancialmente debido al ajuste automático del factor de magnitud de la marca S , dicho factor toma su va-

lor de la desviación estándar del bloque rango R . Este parámetro indica la fuerza de marca incrustada, es decir que a mayor magnitud es más probable que se encuentre la marca incrustada pero esto ocasiona que el bloque marcado se distorsione en gran medida. En el esquema original se utilizaba una magnitud fija, lo que ocasionaba que en ciertos bloques se notará la incrustación del bit. Con esta modificación se calcula una magnitud para cada bloque rango R en donde se incrustará un bit.

2. El uso de una versión reducida del bloque dominio D hace que el bloque rango modificado \hat{R} sea más parecido al bloque rango R . Esto repercute en la imperceptibilidad de la marca, a la vez de que en la fase de detección asegura una mayor aproximación con el bloque dominio.
3. La búsqueda de bloques rango mediante la utilización de una incrustación temporal aumenta la probabilidad de correspondencia de bloques en la etapa de extracción de la marca, puesto que se simula una extracción *a priori*.

Como se ha mostrado en este trabajo, el esquema oculta la marca en diferentes zonas de la imagen, esto aumenta la robustez y no afecta a la imperceptibilidad de la marca incrustada. Pero el hecho de utilizar un número mayor de regiones de incrustación no garantiza la completa extracción de la marca, puesto que crece la posibilidad de falsos positivos (los cuales se reducen en el proceso de detección). Por otra parte, el tamaño de las regiones de incrustación repercute principalmente en la perceptibilidad de la marca, incrementándose cuando las regiones son pequeñas, sin embargo la precisión disminuye en regiones grandes. La principal desventaja del uso de regiones de incrustación es que el esquema puede perder la sincronización de las regiones ante algún ataque geométrico, cabe mencionar que los bits incrustados no se pierden, siguen en la imagen pero el esquema es incapaz de seleccionar la retícula de bloques B en donde se ocultaron. En la siguiente sección se hace una recomendación para hacer robusto a ataques geométricos al nuevo esquema.

Con base en el trabajo realizado se concluye que las regiones de búsqueda local mayores aseguran una mejor correspondencia de un bloque con otro, lo que hace que la perceptibilidad de la marca incrustada disminuya y sobre todo que aumente la precisión del esquema. Además, al limitar la búsqueda de bloques en un área determinada aumenta la velocidad de los procesos de incrustación y extracción.

Podría pensarse que los esquemas de marcas de agua inspirados o que usan la codificación fractal de imágenes no son la mejor opción para incrustar información o que son una medida de comparación no justa (por su baja capacidad y su alta perceptibilidad), pero hay que tomar en cuenta que la base de su funcionamiento es la búsqueda de similitudes y la sustitución de bloques de imagen y tales esquemas son poco abordados en las investigaciones, por lo que el nuevo esquema abre la posibilidad de mejorar tal situación.

Existen una infinidad de esquemas de marcas de agua, cada uno con sus características particulares que los hacen únicos. En esta tesis se hizo una comparación del esquema propuesto con otros que ocultan la marca de forma similar (inspirados por la codificación fractal), sin embargo existen esquemas que mejoran en cierta medida al esquema propuesto, pero ellos mismos tienen sus propias limitantes que hacen que otros los mejoren. Los requerimientos (robustez, capacidad, etc.) de cada uno de ellos los hacen singulares y aptos para una aplicación específica. Por ejemplo si se desea ocultar información de control de copias basta con ocultar unos pocos bits, en este caso un esquema que oculta mucha información no es funcional.

6.3. Trabajo futuro

Las perspectivas a futuro del esquema creado son las siguientes:

1. Utilizar las similitudes en los coeficientes DCT.
2. Crear una política para descartar regiones de incrustación mal formadas.
3. Utilizar una disposición de bloques adaptativa.

El primer punto puede ser desarrollado dentro del proceso de la compresión JPEG, dado que dentro de la compresión JPEG se calculan los coeficientes de la transformada discreta de coseno en ventanas de 8×8 (por lo general), las cuales pueden ser tomadas como la partición de bloques B , con ello se puede llegar a mejorar la robustez en contra de este ataque.

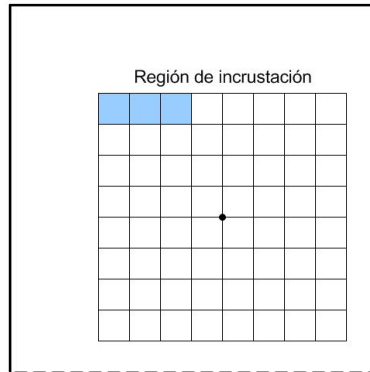


Figura 6.1: Marcado de la región de incrustación.

El segundo punto incumbe a la capacidad de detectar una posible región de incrustación mal formada, es decir que el punto característico P_{max} no sea el adecuado, una posible solución es marcar los primeros bloques con una marca de inicio de región. La figura 6.1 muestra la política de marcado de la región de incrustación. Con esta adaptación el esquema puede detectar la pérdida de sincronización.

El tercer punto cambia en su totalidad el criterio de selección y la forma de los bloques B , para ello se propone que la región de incrustación sea formada por un mosaico de bloques triangulares con centro en el punto característico máximo P_{max} . Tales bloques triangulares después serán transformados a bloques triangulares isósceles rectos (mediante alguna transformada afín), en estos nuevos bloques triangulares se ocultará la información. Consecuentemente para ocultar un bit es necesario realizar una sustitución de bloques con lo que es necesario llevar a cabo la inversa de la transformada afín. Este cambio haría al esquema robusto en contra de ataques geométricos puesto que este tipo de transformaciones mantienen las

proporciones de las líneas y razones de distancia [Wei07]. La figura 6.2 muestra una posible región de incrustación usando una disposición de bloques adaptativa. En [BCD00] se propone la idea de usar patrones triangulares en marcas de agua.

Una aplicación en donde el esquema descrito en esta tesis puede utilizarse es por ejemplo: el marcar las imágenes de un libro digital para el control de copias o para los derechos de autor (ISBN).

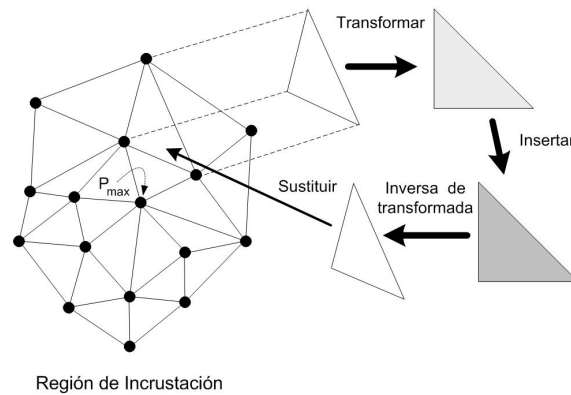


Figura 6.2: Esquema de inserción adaptativo.

Bibliografía

- [And96] Ross J. Anderson, editor, *Proceedings of the First International Workshop on Information Hiding*, tomo 1174, Springer-Verlag, London, UK, 1996, ISBN 3540619968. [Citada en p. 3]
- [Bar88] Michael Barnsley, *Fractals everywhere*, Academic Press Professional, Inc., San Diego, CA, USA, 1988, ISBN 0120790629. [Citada en p. 26, 30]
- [BCD98a] Patrick Bas, Jean Marq Chassery y Franck Davoine, “Self-similarity based image watermarking,” *EUSIPCO’98*, tomo 4:páginas 2277–2280, 1998. [Citada en p. 7, 54, 61, 77, 88, 92, 94]
- [BCD98b] Patrick Bas, Jean Marq Chassery y Franck Davoine, “Using the Fractal Code to Watermark Images,” en “IEEE Int. Conference on Image Processing 98 Proceedings,” tomo 1, páginas 469–473, Chicago, Illinois, USA, oct 1998. [Citada en p. 7, 54, 61, 77, 88, 92, 94]
- [BCD99] Patrick Bas, Jean Marq Chassery y Franck Davoine, “A geometrical and Frequential Watermarking Scheme using similarities,” en “IST/SPIE conference on security ans watermarking of multimedia contents,” 3657, páginas 264–272, IST/SPIE, SPIE, San Jose, California, USA, ene 1999. [Citada en p. 57]
- [BCD00] Patrick Bas, Jean Marq Chassery y Franck Davoine, “Robust Watermarking Based on the Warping of Pre-Defined Triangular Patterns,”

- EI'2000: Security and Watermarking of Multimedia Content II*, páginas 99–109, 2000. [Citada en p. 98]
- [BCD02] Patrick Bas, Jean Marq Chassery y Franck Davoine, “Geometrically Invariant Watermarking Using Feature Points,” *IEEE Transactions on Image Processing*, tomo 11(9):páginas 1014–1028, 2002. [Citada en p. 62, 64]
- [Bur07] Matt Burke, “Ellacoya Data Shows Web Traffic Overtakes Peer-to-Peer (P2P) as Largest Percentage of Bandwidth on the Network,” jun 2007. [Citada en p. 4]
- [CDF06] I. J Cox, G. Doerr y T. Furon, “Watermarking is not Cryptography,” en “Proceedings of the 5th International Workshop on Digital Watermarking,” páginas 1–15, 2006. [Citada en p. 3]
- [CM02] I.J. Cox y M.L. Miller, “The First 50 Years of Electronic Watermarking,” *JASP*, tomo 2002(2):páginas 126–132, 2002. [Citada en p. 3]
- [EF95] Ahmet M. Eskicioglu y Paul S. Fisher, “Image Quality Measures and Their Performance,” *IEEE Transactions on communications*, tomo 43(12):páginas 2959–2965, dic 1995. [Citada en p. 21]
- [Fis95] Yuval Fisher, *Fractal Image Compression: Theory and Application*, Springer-Verlag, 1995, ISBN 0387942114. [Citada en p. 28, 32]
- [Gul03] Kamal Gulati, *Information Hiding Using Fractal Encoding*, Proyecto Fin de Carrera, Indian Institute of Technology Bombay, Mumbai, India, ene 2003. [Citada en p. 51, 88, 92]
- [GYB02] Li Ghuanhua, Zhao Yao y Yuan Baozong, “Using the fractal code to watermark images,” *ICSP 02*, páginas 829–832, 2002. [Citada en p. 46, 88]

- [HC00] Cheng-Hsiung Hsieh y Shen-Shyong Chen, “Application of fractal model to visible watermarking,” 2000. [Citada en p. 58]
- [HS88] C. Harris y M. Stephens, “A Combined Corner and Edge Detection,” en “Proceedings of The Fourth Alvey Vision Conference,” páginas 147–151, 1988. [Citada en p. 62]
- [IC01] Matthew Miller y Jeffrey Bloom Ingemar Cox, *Digital Watermarking*, Morgan Kaufmann, 2001, ISBN 1558607145. [Citada en p. 21]
- [Jac92] A. E. Jacquin, “Image coding based on a fractal theory of iterated contractive image transformations,” *IEEE Trans. Image Processing*, tomo 1(1):páginas 18–30, 1992. [Citada en p. 28, 33, 43]
- [JJ98] Neil F. Johnson y Sushil Jajodia, “Exploring Steganography: Seeing the Unseen,” *IEEE Computer*, tomo 31(2):páginas 26–34, 1998. [Citada en p. 1]
- [JJD01] N.F. Johnson, S. Jajodia y Z. Duric, “Information hiding: steganography and watermarking - attacks and countermeasures,” 2001. [Citada en p. 9]
- [Kah67] David Kahn, *The Codebreakers*, MacMillan, New York, 1967. [Citada en p. 1]
- [Kha02] M. R. Khadivi, “IFS and its use in cryptography and steganography,” Informe técnico, Jackson State University, Mississippi, USA., 2002. [Citada en p. 58]
- [KP00] Stefan Katzenbeisser y Fabien A. Petitcolas, *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House Inc., Norwood, MA, USA, 2000, ISBN 1580530354. [Citada en p. 9]

- [Lan00] Gerrit Cornelis Langelaar, *Real-time Watermarking Techniques for Compressed Video Data*, Tesis Doctoral, Delft University of Technology, Netherlands, feb 2000. [Citada en p. 15]
- [Lea96] Thomas Leary, “Cryptology in the 15th and 16th Century,” *Cryptologia XX*, (3):páginas 223–242, Jul 1996. [Citada en p. 2]
- [LMBO95] Steven H. Low, Nicholas F. Maxemchuk, Jack Brassil y Lawrence O’Gorman, “Document Marking and Identification Using Both Line and Word Shifting,” en “INFOCOM 2,” 3, páginas 853–860, Jul 1995. [Citada en p. 2]
- [LMSP99] John Lach, William H. Mangione-Smith y Miodrag Potkonjak, “Robust FPGA intellectual property protection through multiple small watermarks,” en “DAC ’99: Proceedings of the 36th ACM/IEEE conference on Design automation,” páginas 831–836, ACM Press, New York, NY, USA, 1999, ISBN 1581331097. [Citada en p. 19]
- [Man82] Benoit B. Mandelbrot, *The Fractal Geometry of Nature*, W. H. Freeman, ago 1982, ISBN 0716711869. [Citada en p. 26]
- [MK07] Dmitriy Mekhontsev y Alexei Kravchenko, “3D Fractals,” Web resource, 2007. [Citada en p. 26]
- [Moe03] T. Moerland, “Steganography and Steganalysis,” may 2003. [Citada en p. 24]
- [MvOV96] Alfred J. Menezes, Paul C. van Oorschot y Scott A. Vanstone, *Handbook of applied cryptography*, CRC Press, Boca Raton, Florida, 1996, ISBN 0849385237. [Citada en p. 3]
- [PJ96] Joan Puate y Fred Jordan, “Using fractal compression scheme to embed a digital signature into a image,” *SPIE-96 proceedings*, 1996. [Citada en p. 43, 88]

- [Ram05] Alejandro Martínez Ramírez, *Compresión fractal multi-resolución de imágenes*, Tesis doctoral, Instituto Nacional de Astrofísica, Óptica y electrónica, Puebla, México., feb 2005. [Citada en p. 39, 41]
- [Sch65] Gaspar Schotti, “Steganographica,” 1665, <http://www.petitcolas.net/fabien/steganography/steganographica/index.html>. [Citada en p. 2]
- [SRN01] Amhamed Saffor, Abd Rahman Ramli y Kwan-Hoong Ng, “A Comparative Study of Image Compression Between JPEG and Wavelet,” *Malaysian Journal of Computer Science*, tomo 14(1):páginas 39–45, 2001, ISSN 01279084. [Citada en p. 21]
- [VDM02] Christophe De Vleeschouwer, Jean-François Delaigle y Benoit Macq, “Invisibility and application functionalities in perceptual watermarking an overview,” en “Proceedings of the IEEE,” tomo 90, páginas 64–77, ene 2002. [Citada en p. 15]
- [VPP⁺01] S. Voloshynovskiy, S. Pereira, T. Pun, J.J.Eggers y J.K. Su, “Attacks on Digital Watermarks: Classification, Estimation-based Attacks and Benchmarks,” *IEEE Communications Magazine*, tomo 39(8), ago 2001. [Citada en p. 25]
- [Wei07] Eric W. Weisstein, “Affine Transformation.” MathWorld—A Wolfram Web Resource., 2007, <http://mathworld.wolfram.com/AffineTransformation.html>. [Citada en p. 98]
- [WP00] Andreas Westfeld y Andreas Pfitzmann, “Attacks on Steganographic Systems,” en “IH ’99: Proceedings of the Third International Workshop on Information Hiding,” páginas 61–76, Springer-Verlag, London, UK, 2000, ISBN 354067182X. [Citada en p. 24]

- [Yao03] Zhen Yao, “Fixed Point in Fractal Image Compression as Watermarking,” en “Proceedings IEEE International Conference on Image Processing (ICIP),” Barcelona, España, sep 2003. [Citada en p. 48, 89]
- [Zak06] Robert Hobbes Zakon, “Hobbes’ Internet Timeline,” nov 2006, <http://www.zakon.org/robert/internet/timeline/>. [Citada en p. 4]