Universidad Autónoma del Estado de México

Facultad de Ingeniería

Cálculo de la dispersión de píxeles en imágenes RGB para Esteganografía con base en la Teoría Fractal

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

Doctor en Ciencias de la Ingeniería

PRESENTA:

M.C.C. Héctor Caballero Hernández

COMITÉ DE TUTORES:

Dra. Vianney Muñoz Jiménez
Dr. Marco Antonio Ramos Corchado
Dr. Marcelo Romero Huertas

Toluca de Lerdo, México, diciembre, 2020.

Dedicatoria

Dedicado a:

Este trabajo está dedicado a todas las personas que me han ayudado de forma generosa y desinteresada durante estos tres años, no solo para lograr culminar este importante trabajo de investigación, sino también en mi crecimiento espiritual y humano, lo cual considero una bendición de Dios.

Agradecimientos

Este trabajo no pudo ser realizado sin la importante colaboración de mi directora de tesis la Dra. Vianney Muñoz Jiménez, gracias a su prudente dirección y sabios consejos. Agradezco también la Dra. Alicia Morales Reyes por todo su conocimiento donado, al Dr. Marco Antonio Ramos Corchado y al Dr. Marcelo por sus observaciones invaluables en el proceso de desarrollo de esta tesis.

Agradezco a CONACYT por la asignación de la beca con número de registro 445998 para estudios de posgrados, financiamiento sin el cual no hubiese sido posible la culminación de este proyecto de investigación.

Una virtud difícil de hallar es la generosidad, tengo muy presente la ayuda de mis padres para continuar superándome como ser humano y hacer de utilidad mi estancia en esta vida. Agradezco a Dios por la gracia de la sabiduría y paciencia.

Agradezco a todos aquellos que han contribuido con sus consejos y amistad, durante este periodo de estudios.

Resumen

La esteganografía es una rama de las ciencias computacionales que estudia cómo transportar la información de forma oculta en objetos digitales llamados objetos portadores, tales como imágenes, vídeo, audio, entre otros. En la actualidad, los métodos de esteganografía modifican los componentes que forman la secuencia de datos en los objetos portadores desde una perspectiva espacial, frecuencial o la combinación espacial-frecuencial, con la finalidad de crear patrones que permitan incrustar mensajes como texto o incluso imágenes dentro de dichos objetos.

La presente tesis aborda una nueva perspectiva sobre el transporte de información oculta en imágenes digitales con modelo de color RGB (*Red Blue Green*, por sus siglas en inglés) sin pérdida de información por compresión. El eje central de la investigación es presentar la combinación de esquemas de incrustación espacial y frecuencial para obtener altas cargas de datos incrustados, sin pérdida de datos al momento de recuperar la información, con la finalidad de proponer un método viable para esteganografía que destaque por su versatilidad para incrustar cualquier tipo de objeto digital, con estego-imágenes sin aparentes distorsiones visuales, y al mismo tiempo proveer un esquema de aseguramiento en la información para no comprometer el mensaje oculto.

El método de esteganografía propuesto, denominado como VVRSM (*Virtual Variable Redistribution Steganographic Method*), presenta cuatro partes importantes. La primera parte se enfoca en el tratamiento del mensaje a incrustar, el cual sin importar que tipo de mensaje digital sea, es tratado como una cadena de texto mediante compresión por el algoritmo LZW (*Lempel-Ziv-Welch*, por sus siglas en inglés) y la técnica de codificación base 64. El mensaje que ha sido representado en base 64 es trasformado por un algoritmo espacial, el cual es de desarrollo propio, y aprovecha la dimensión fraccionaria para la generación de vectores variables encargados de distribuir de forma aleatoria los símbolos del alfabeto de la base en la cual esta codificado el mensaje. La codificación por vectores fractales incrementa la seguridad del mensaje incrustado.

La segunda parte del método genera una redistribución virtual sobre la imagen de portada (receptora del mensaje) y permite cambiar la ubicación espacial de los píxeles, el resultado de la distribución espacial es una nueva imagen denominada como intermedia, y es la receptora del mensaje codificado, al finalizar la incrustación del mensaje los píxeles son nuevamente ubicados en su posición original. La tercera parte incrusta las reglas de recuperación del mensaje y el proceso inverso de la imagen intermedia. La cuarta parte aplica una máscara para eliminar los patrones de incrustación y logra disminuir la detección por estegoanálisis.

Los resultados obtenidos en la base experimental permiten cumplir con los objetivos inicialmente planteados, desde el punto de vista de alta incrustación de datos, logrando cargas de datos superiores a 6 bpp, y evasión de estegoanálisis con cargas cercanas a 1.7 bpp, sin distorsiones en las estego-imágenes y sin pérdida de información al momento de recuperar el mensaje original. Las estego-imágenes son evaluadas por métricas de calidad, tales como: PSNR (*Peak Signal Noise to Ratio*), SNR (*Signal Noise Ratio*), MSE (*Mean Square Error*), SSIM (*Structural Similarity Index Metric*), CC (*Cross Correlation*), entre otras.

Abstract

Steganography is a branch of Computer Science that studies how to transport information hidden in digital objects called cover objects, such as images, video and audio. Steganography algorithms modify the components that make up the data sequence in cover objects from a spatial, frequency or spatial-frequency combination, to create patterns that allow embedding messages like a text or even images within of such objects.

The present thesis addresses a new perspective on the transport of hidden information in digital images with RGB (*Red Blue Green*) color model with a format without loss of data by compression. The central axis of the research is to present the combination of spatial and frequency embedding schemes to obtain high loads of embedded data. Further, without loss of data when retrieving information to propose a viable method for steganography that stands out for its versatility to embed any digital object. The new method preserves the high quality in the stego-images, and at the same time to provide an assurance scheme in the information so as not to compromise the hidden message.

The proposed steganography method, called VVRSM (*Virtual Variable Redistribution Steganographic Method*), has four essential parts. The first part is focused on the treatment of the message to be embedded, which, regardless of what type of digital message it is, is treated as a text string through compression by the LZW (*Lempel-Ziv-Welch*) algorithm and base 64 encoding technique. The message in base 64 is transformed by a spatial algorithm which is its development, which takes advantage of the fractional dimension for the generation of variable vectors in charge of randomly distributing the symbols of the base alphabet in which the message is encrypted. Encoding by fractal vectors increases the security of the embedded message.

The second method part consists of generating a virtual redistribution on the cover image (receiver of the message) which allows changing the spatial location of the pixels, the result of the spatial distribution is a new image called as intermediate. The intermediate image receives the encoded message, and at the end of embedding the message, the pixels are then placing in its original position. The third part consists of embedding the message retrieval rules and the inverse process of the intermediate image. The fourth part consists of applying a mask to eliminate insert patterns and achieve a decrease by detection of steganalysis.

The results obtained on the experimental basis allow to know the objectives initially set, from high data embedding, achieving data loads greater than 6.0 bpp, and evasion of steganalysis with close loads 1.7 bpp, without distortions in the stego-images and without loss of information when recovering the original message. The stego-images are evaluated by quality metrics, such as: PSNR (*Peak Signal Noise to Ratio*), SNR (*Signal Noise Ratio*), MSE (*Mean Square Error*), SSIM (*Structural Similarity Index Metric*), CC (*Cross Correlation*), among others.

Contenido

Página

Índice de figuras

Índice de Tablas

Índice de Pseudocódigos

Acrónimos

Acrónimo	**Significado**
AES	Advanced Encryption Standard
AMBTC	Absolute Moment Block Truncation Coding Compression
BBPVD	Blocks Based Pixel Value Differencing
BER	Bit Error Rate
BCT	Block Truncation Coding
BPCS	Bit Plane Complexity Segmentation Steganography
BPIS	Secure Block Permutation Image Steganography
CMYK	Cyan, Magenta, Yellow and Key
CPLD	Complex Programmable Logic Device
DCT	Discrete Cosine Transformation
DFT	Discrete Fourier Transformation
DWT	Discrete Wavelet Transformation
EMD	Exploiting Modification Direction
ENMPP	Expected Number of Modifications per Pixel
ERBP	Enhanced Resilient Back-Propagation
FPGA	Field-programmable Gate Array
GIF	Graphics Interchange Format
HBC	Hidden behind corner
IDWT	Inverse Discrete Wavelet Transformation
JPEG	Joint Photographic Experts Group
JPEG2000	Joint Photographic Experts Group 2000
LSB	Least Significant Bit
LZW	Lempel-Ziv-Welch
M-LSB	Modified Least Significant Bit
MSE	Mean Squared Error
MSSIM	Medium Structural Similarity Index
NCC	Normalized Cross Correlation
NFC	Near Field Communication
ORPSA	Optimal Reference Point Selection Approach
PMM	Pixel Mapping Method
PRN	Pseudo Ramdom Number
PRNG	Pseudo Random Number Generator
PSNR	Peak Signal to Noise Ratio
PVD	Pixel Value Differencing
RGB	Red Blue Green
RS	Regular-singular
RSA	Rivest, Shamir y Adleman
SOM	Self-Organizing Maps
SSIM	Structural Similarity Index
SVD	Singular Value Descomposition
TIFF	Tagged Image File Format
TSM	Two-sides match
VQ	Vector Quantization
YCbCr	Luma Chrominance Blue and Red
YIQ	Y in-phase quadrature

Capítulo 1

Introducción

La seguridad informática ha tomado relevancia en la actualidad debido a la información importante que se maneja en las empresas, gobiernos y otras entidades. Con base en lo anterior, se han implementado distintos procedimientos que permiten garantizar la integridad de los datos, así como su confidencialidad [161]. Entre las áreas que han experimentado avances en la protección de la información se encuentra la criptografía y la esteganografía. La esteganografía toma como medio de transporte un objeto portador en el cual se oculta la información [5]. La diferencia entre la criptografía y la esteganografía es que la primera necesita enviar el mensaje sin que un tercero pueda comprenderlo, mientras que la segunda, se genera un estudio sobre los métodos y procedimientos para que la información pase inadvertida en el objeto portador ([5], [81]).

Los métodos clásicos que se han utilizado por parte de la esteganografía han ido evolucionando, éstos se basan en la incrustación de información en archivos digitales, por ejemplo: imágenes, video, sonido, entre otros, mediante técnicas capaces de manipular los archivos desde distintas perspectivas, tales como el dominio de la frecuencia y del espacio.

Una de las características frecuentes de las investigaciones reportadas, es que éstas se centran en la combinación de los diferentes métodos de esteganografía tanto en el dominio del espacio como el de la frecuencia, además de la combinación de mecanismos criptográficos que permitan reforzar la seguridad de la información contenida. Sin embargo, la información incrustada sigue expuesta al compararse con la estego-imagen (imagen con mensaje) versus la imagen de portada (imagen original) [68]. Se ha observado que los trabajos están enfocados en incrementar la seguridad de la información incrustada en la estego-imagen como se puede observar en lo reportados en [73] y en [116], teniendo en cuenta que las investigaciones recientes se centran en proponer nuevas formas de incrustación, aunque no precisamente en elevar la carga de datos en las estego-imágenes ([1], [87]), por otro lado, los algoritmos reportados siguen enfocándose mayoritariamente en imágenes en escala de gris, por lo cual no se aprovechan los canales que brindan los modelos a color.

Las investigaciones que explotan el área de los modelos de color hacen uso de las mismas técnicas y algoritmos de ocultamiento de información que se manejan en imágenes a escala de gris, proponiendo modificaciones sobre algoritmos existentes y sin aplicar técnicas de estegoanálisis (detección de mensajes ocultos) para validar sus resultados.

El presente trabajo de investigación está enfocado al estudio y desarrollo de un nuevo método esteganográfico que permita el ocultamiento de la información en imágenes digitales a color, considerando el modelo RGB (*Red Blue Green*) y sin pérdida de información. Los datos para incrustar

dentro de la imagen de portada podrán ser texto u otras imágenes. Esta investigación se apoya de la teoría fractal para determinar la secuencia y la lógica de transformación de datos que serán incrustados en una imagen digital, considerando el análisis de dispersión de píxeles y coeficientes que permita la incrustación de la información sin distorsionar la estego-imagen en relación con la imagen de portada. Adicionalmente, los resultados obtenidos se validan con estegoanálisis para dar validez al método propuesto.

1.1 Planteamiento del problema

Los métodos esteganográficos actuales que se emplean en imágenes digitales han hecho énfasis en la modificación de algoritmos como LSB (*Least Significant Bit*), PVD (*Pixel Value Differencing*), DWT (*Discrete Wavelet Transformation*), DCT (*Discrete Cosine Transformation*), entre otros, o la combinación de varios de estos, además incluyen técnicas criptográficas para reforzar la seguridad como recurso en caso de que la información oculta sea descubierta. Actualmente en esteganografía se desarrollan métodos enfocados en incrementar la seguridad de la información incrustada en la estego-imagen, pero en la mayoría de los trabajos presentados no se perciben grandes esfuerzos por incrementar la cantidad de información que pueda albergar la estego-imagen [101], otro punto a mencionar es que dichos algoritmos siguen enfocándose mayoritariamente en imágenes basadas en escala de gris y que en los modelos de color no se ven aprovechados [154] los canales adicionales para incrustar información de forma superior en comparación a los modelos basados en escala de gris.

Los algoritmos y métodos de esteganografía que explotan el área de los modelos de color, se apoyan ampliamente en las mismas técnicas y algoritmos de ocultamiento de información que se manejan en imágenes a escala de gris ([10], [67], [78]). Se observa que en su gran mayoría proponen modificaciones sobre algoritmos existentes, aunque los niveles de incrustación de información difícilmente superan a los modelos de escala de gris, además que normalmente los trabajos reportados no aplican técnicas de estegoanálisis para validar sus métodos.

1.2 Justificación

Los algoritmos actuales en el área de esteganografía aplicados en imágenes en su gran mayoría son modificaciones a los propuestos en la década de los 90′s del siglo pasado y de la primer década del presente siglo, ya sea que estén basados en el dominio espacial o de la frecuencia, aparentemente combinando con éxito técnicas criptográficas con las esteganográficas, pero son escasos los trabajos cuando se trata de ser sometidos a través de procesos de estegoanálisis moderno, además de que la cantidad de datos incrustados cuando se trabaja con imágenes a color no logra incrustación de datos con tasas elevadas. Por lo tanto, es necesario generar una propuesta novedosa de esteganografía con la capacidad de incrementar la información incrustada en imágenes digitales con modelo de color RGB sin descuidar la seguridad intrínseca que debe de llevar a cabo el método con la información.

En este trabajo de investigación se realiza un estudio de la teoría fractal con una codificación basada en gramáticas regulares aplicada tanto para texto como para imágenes, que permite generar codificaciones dinámicas favoreciendo el ocultamiento de información en la esteganografía. Se estudia la dispersión de píxeles y coeficientes de una imagen, para solventar los problemas existentes de la relación entre alta calidad de la imagen y la cantidad de datos incrustados en ella. La combinación del enfoque espacial y de frecuencia permite explotar las características deseables en un sistema de esteganografía donde se busca alta capacidad de incrustación de datos (basándose en el dominio del espacio) y robustez con el dominio de la frecuencia.

La contribución de esta investigación es el aprovechamiento de la dimensión fractal, en este sentido se puede explotar la dimensión fraccionaria para obtener una relación entre la información incrustada y su distribución para el proceso de esteganografía, considerando el número de reglas que se generan para la codificación de la información y la lógica de incrustación.

1.3 Objetivos

En este trabajo de investigación se definen los siguientes objetivos.

Objetivo general

Desarrollar un nuevo método para la esteganografía basado en la incrustación parcial de información en imágenes RGB mediante el cálculo de la dimensión fractal y de la estimación de la dispersión entre píxeles.

Objetivos específicos

- Combinar el enfoque espacial y del dominio de la frecuencia para la incrustación de datos ocultos.
- Aplicar teoría fractal para calcular la incrustación de píxeles en segmentos de espacios 2D.
- Validar el método de esteganografía propuesto con el análisis de técnicas de medición de calidad de imágenes y software de estegoanálisis basado en métodos estadísticos.

1.4 Hipótesis

Mediante la incrustación de información representativa en imágenes digitales RGB empleando teoría de fractales con cálculo de la dispersión de píxeles de la imagen, se aprobarán las pruebas de estegoanálisis de chi-cuadrada y RS manteniendo los niveles de PSNR superiores a 40 dB y una tasa de incrustación superior al 30 % con respecto a la relación bits por píxel de la imagen de portada.

1.5 Contribuciones

En los siguientes párrafos se muestran los artículos que se han publicado durante el desarrollo de la presente investigación, así como la participación en congresos.

Publicaciones

- "A Review of Steganography Techniques for Digital Information Transmission for Secure Channels with Digital Images", publicado en la revista IEEE Latin America Transactions con ISSN: 1548-0992, el 15/12/2019. Las métricas de la revista son Cite Score: 1.05, SJR:0.337 y SNIP: 0.606.
- "Algoritmo para Transmisión de Información Segura en Dispositivos NFC", publicado en la revista ReCIBE con ISSN: 2007-5484, el 20/11/2018. La revista esta indexada en los siguientes motores de búsqueda: Latindex, DOAJ, REDIB y Redalyc.

- "Método de esteganografía a través de la dimensión fractal y el algoritmo de LSB; una nueva perspectiva en imágenes RGB", presentado en el XXXI Congreso Nacional y XVII Congreso Internacional de Informática y Computación de la ANIEI y publicado en la revista Tecnología Educativa CONAIC con ISSN: 2395-9061, el 15/04/2019. La revista está indexada en los motores de búsqueda DOAJ y Latindex.
- "Aplicación de técnicas de dispersión de píxeles para esteganografía mediante expresiones regulares", publicado en la revista Komputer Sapiens con ISSN: 2007-0691, el 20/04/2020. La revista está indexada en los motores de búsqueda Latindex y en el sistema de revistas de CONACYT.
- "Spatial and Frequency Domain Combination based in Schemes to Steganography in RGB Digital Images using Cantor Set", publicado en la revista IJARCS con ISSN 0976-5697, el 20/06/2020. La revista esta indexada en los motores de búsqueda CROSSREF, EBSCO, HOST Index Copernicus, Open J Gate, Electronic Journal Library, NewJour, Science central.com, Ulrichs Web, Dayang Journal System y cuenta con los siguientes factores de impacto ICV: 66.64, SJIF: 5.845, ISRA JIF: 3.727, RGIF: 0.29, GIF: 0.7659 y Reaserch Gate: 0.80.

Ponencias

- Jornada Académica de Ingeniería Informática 2017 en el Tecnológico de Estudios Superiores de San Felipe del Progreso con el tema "Esteganografía con fractales. Una perspectiva lingüística", noviembre de 2017.
- Coloquio de Ingeniería en Computación organizado en el Centro Universitario UAEMex Atlacomulco con el tema "Esteganografía en imágenes digitales a color mediante conjuntos de datos", mayo de 2018.
- Primer Coloquio de Ingeniería Primera Edición del Coloquio de investigación en ingeniería y el 11º Curso-Taller "Temas actuales en ciencia del agua" organizado por UAEMex, octubre de 2018.
- Congreso internacional ANIEI 2018 CUCEI, como ponente del tema de investigación "Método de esteganografía a través de la teoría fractal para imágenes a color RGB", octubre de 2018.
- Segundo Coloquio de Ingeniería organizado por la UAEMex como Coloquio de investigación en ingeniería y el 11º Curso-Taller "Temas actuales en ciencia del agua", noviembre de 2019.

Estancia

Se realizó una estancia de investigación en el Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE) en el área de Ciencias de la Computación, con el objetivo de implementar la experimentación exhaustiva de métodos de esteganografía, durante el periodo de septiembre a octubre de 2019.

1.6 Estructura de la tesis

La tesis consta de cinco capítulos. En el capítulo 1, se presenta la introducción, la justificación, los objetivos del trabajo de investigación y la hipótesis.

En el capítulo 2, se presenta la descripción de los conceptos teóricos fundamentales para el entendimiento del desarrollo de esta investigación tales como: imagen, esteganografía y teoría de

fractales. En este capítulo también se aborda el estado del arte, presentando los trabajos actuales y de impacto en el área de esteganografía tanto para imágenes en escala de gris e imágenes a color.

El capítulo 3, presenta la metodología propuesta para alcanzar los objetivos de esta investigación.

El capítulo 4, describe los resultados obtenidos de la aplicación de la teoría fractal, la dispersión de píxeles y la representación de datos a través de técnicas de codificación con bases numéricas y técnicas de compresión.

En el capítulo 5, se presentan las conclusiones, los productos obtenidos en esta tesis de investigación y el trabajo futuro.

Finalmente, los últimos apartados son el anexo A, en el cual se presentan formas de compresión adicionales para datos, en el anexo B se presentan el cálculo de las dimensiones de distintos fractales deterministas, y el anexo C contiene pseudocódigos adicionales sobre manejo de información en las imágenes de portada y análisis de cadenas de datos, para la incrustación de las reglas que permiten la recuperación del mensaje oculto.

Capítulo 2

Marco Teórico y Revisión del Estado del Arte

En el presente capítulo se presentan los conceptos teóricos fundamentales para la comprensión del entorno de la esteganografía y las generalidades de los formatos de las imágenes que se empleando para incrustar información oculta en ellas. En la sección 2.1 se genera una narrativa sobre los aspectos generales de esteganografía, los métodos espaciales y en dominio de la frecuencia para incrustar información en objetos portadores. En la sección 2.2 se muestra la definición de imagen y posteriormente se muestran los formatos más conocidos de imágenes de digitales que se emplean para esteganografía, así como los modelos de color que emplean estos formatos. En la sección 2.3 se presentan los mecanismos de representación de datos mediante compresión y posterior a esto, se presenta la teoría general de fractales, en donde se explica la definición de fractales, definición de dimensión fractal, tipos de fractales y su aplicación actual. En la sección 2.4 se describen las métricas más importantes empleadas para la validación de estego-imágenes desde una perspectiva visual humana y estadística. En la sección 2.5 se presentan de forma concreta la clasificación de los métodos de estegoanálisis para la detección de información en estego-imágenes. Finalmente, en la sección 2.6 se presenta el estado del arte sobre trabajos de representación de datos mediante compresión, así como una revisión extensa de trabajos de esteganografía en imágenes digitales en escala de gris y con modelos de color.

2.1 Esteganografía

La esteganografía es el arte o bien la ciencia del ocultamiento de la información, que estudia los mecanismos de envío de la información para que esta pase desapercibida. Un sistema esteganográfico no requiere de un intercambio de información, es decir, no necesita una llave oculta llamada estego-llave para recuperar la información [76]. Se debe tomar en cuenta que dentro de un sistema de esteganografía existen atacantes pasivos y activos o maliciosos [131], tomando en cuenta que un atacante puede probar la existencia de un mensaje oculto dentro de una imagen de portada. Si el atacante quien desea obtener la información oculta no logra afirmar que existe un mensaje incrustado, entonces, el sistema puede considerarse seguro.

En esteganografía existen elementos importantes a identificar como la *imagen de portada*, que es la entidad en la cual se incrustará el objeto a ocultar y el estego objeto llamado $estego - imagen$, que corresponde al resultado de la fusión del mensaje oculto con la imagen de portada [101]. El proceso general de esteganografía se ilustra en la figura 1. La primera etapa, consiste en realiza un preprocesamiento a la imagen de portada para eliminar el posible ruido que contenga, posteriormente se incrusta el mensaje, después se valida que se incrustó correctamente el mensaje para finalmente recuperar el mensaje que se ha incrustado de forma oculta [154].

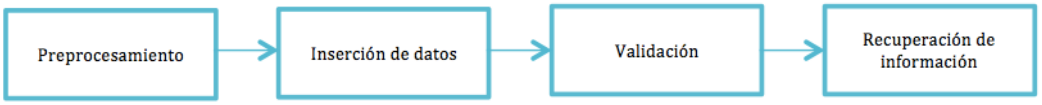


Figura 1: Diagrama general del proceso de esteganografía [131]

Un sistema de esteganografía es una quíntupla $\wp = (C, M, K, D_k, E_k)$, donde C es el conjunto de todas las imágenes de portada, M es el conjunto de mensajes secretos a ocultar, K es el conjunto de llaves secretas, $E_k : C \times M \times K \to C$, y $D_k : C \times K \to M$ [28] son dos funciones, la primera es la función de incrustación y sea la segunda la función de extracción, de tal forma que $D_k(E_k(c, m, k).k = m$ [43].

Sea P_c la probabilidad de distribución sobre el conjunto de todas las imágenes de portada y sea P_s la probabilidad de distribución de las estego-imágenes. La entropía relativa de Kullback-Leibler (distancia de Kullback Leibler o entropía relativa)) es la distancia entre dos probabilidades de distribución [50], y se definen en la ecuación (1):

$$D\left(P_C \parallel P_S\right) = \sum_{c \in C} P_C(c) \log\left(\frac{P_C(c)}{P_S(c)}\right) \tag{1}$$

Si $P_c = P_s$ entonces la distancia es 0.

Sea $\wp$ un estego-sistema y sea P_C y P_S la probabilidad de distribución de dos mensajes de portada y estego-objetos [75]. $\wp$ es llamado se llama ε-seguro contra atacantes pasivos si (2):

$$D\left(P_C \parallel P_S\right) \leq \varepsilon \tag{2}$$

Existen varias clasificaciones de los métodos de esteganografía [76] uno de ellos son los métodos de sustitución, los cuales se encargan de sustituir partes redundantes de un objeto en un mensaje secreto [143]. Existe una gran cantidad de métodos de este tipo. Los métodos más importantes que se emplean en la esteganografía son los de sustitución del bit menos significativo ([134], [135]) y los que utilizan métodos en el dominio de la frecuencia, estos últimos son ampliamente utilizados debido a la robustez que ofrecen contra ataques estáticos. A continuación, se describen las más destacadas.

El método **LSB** (*Least Significant Bit*) también conocido como bit menos significativo, consiste en modificar el bit de menor peso de un byte de la imagen de portada, dado que, la sustitución del bit menos significativo no distorsiona el objeto, desde el punto de percepción humano. Una de las ventajas que presenta este método es que no aumenta el tamaño del objeto modificado, sin embargo, este puede ser detectado bajo un análisis espectral o estadístico [34]. En la figura 2, se muestra un ejemplo de selección de los bits menos significativos de un píxel RGB.

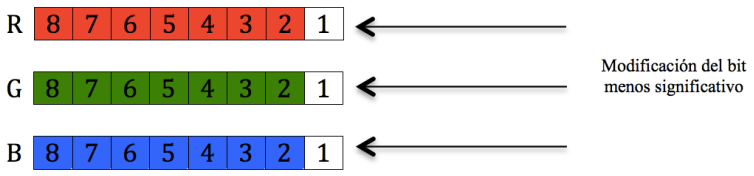


Figura 2: Selección del bit menos significativo de un píxel RGB para LSB

El método **PVD** (*Pixel Value Differencing*), modifica los valores de los píxeles en la imagen dado que el mecanismo de ocultamiento de la información está basado en sustituir el valor numérico de la diferencia obtenida entre los bloques de dos píxeles continuos dentro de una imagen, por otros similares en los cuales se incluyen bits de datos ocultos. Generalmente es aprovechado para imágenes a escala de gris ([39], [157]). El proceso de PVD se puede observar en la figura 3, en ésta se presentan cuatro casos de cómo se puede seleccionar el píxel para su modificación mediante PVD, se considera que las coordenadas de los píxeles están invertidas, siendo columna-renglón y no renglón-columna.

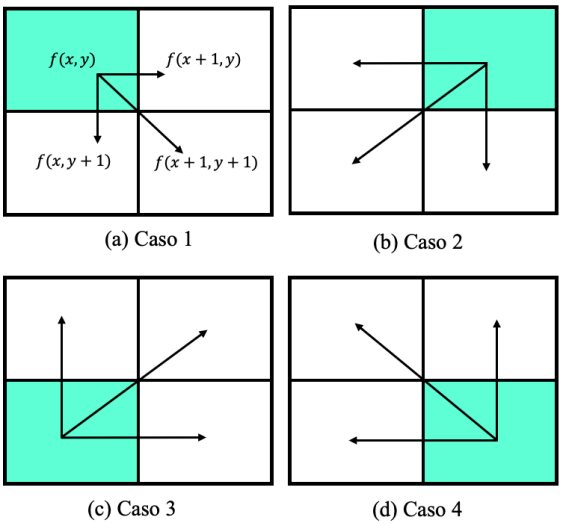


Figura 3: Proceso de modificación de píxeles utilizando PVD [70]

El método **PMM** (*Pixel Mapping Method*) [17], permite ocultar datos dentro de una imagen a escala de gris al seleccionar los píxeles en la cual incrusta la información, a través de funciones matemáticas dependientes de la intensidad del valor del píxel semilla y considerando una vecindad de 8 píxeles, se seleccionan en sentido contrario a las manecillas del reloj. Previamente, se verifica que los píxeles seleccionados, así como sus vecinos se localizan dentro de los bordes de la imagen, posteriormente de la incrustación de datos se realiza un mapeo de cada dos o cuatro bits del mensaje secreto para cada píxel vecino [18].

Los métodos más importantes que se emplean en el dominio de la frecuencia emplean transformadas, tales como: **DFT** (*Discrete Fourier Transformation*), **DCT** (*Discrete Cosine Transformation*) y **DWT** (*Discrete Wavelet Transformation*). Una ventaja de estos métodos sobre los métodos de dominio espacial es que la información está menos expuesta a la compresión, recorte y al procesamiento de la imagen [125].

DFT es un método basado en transformaciones matemáticas que convierten los píxeles para dar el efecto de difundir la ubicación de los valores de los píxeles sobre una región de la imagen. Con este tipo de método se descompone una función periódica en sus harmónicos (espectro de frecuencias). Al combinar las funciones harmónicas en base 2D es posible sintetizar funciones arbitrarias espaciales. Es un método basado en la desasociación (decorrelating) de la imagen, con esto se manipula el dominio de la frecuencia de la imagen e incrementar el número de coeficientes a transformar para incrementar la capacidad de incrustar datos. En la incrustación de los datos se cuantifican las magnitudes de las partes real e imaginaria de los coeficientes [60].

La *DCT* permite ocultar el mensaje secreto en el bit menos significativo del coeficiente del coseno discreto de una imagen digital [149], además se basa en descomponer la imagen en bloques

de 8×8 píxeles; de izquierda a derecha y de arriba hacia abajo, este método es aplicado en cada bloque que se obtiene de la imagen. Básicamente se toman 8×8 píxeles y se transforman en una matriz de 8×8 coeficientes DCT. Cada coeficiente de DCT representa cuanto se necesita escalar un determinado conjunto de funciones que tienen una base de dos dimensiones para generar los píxeles originales como se muestra en la figura 4.

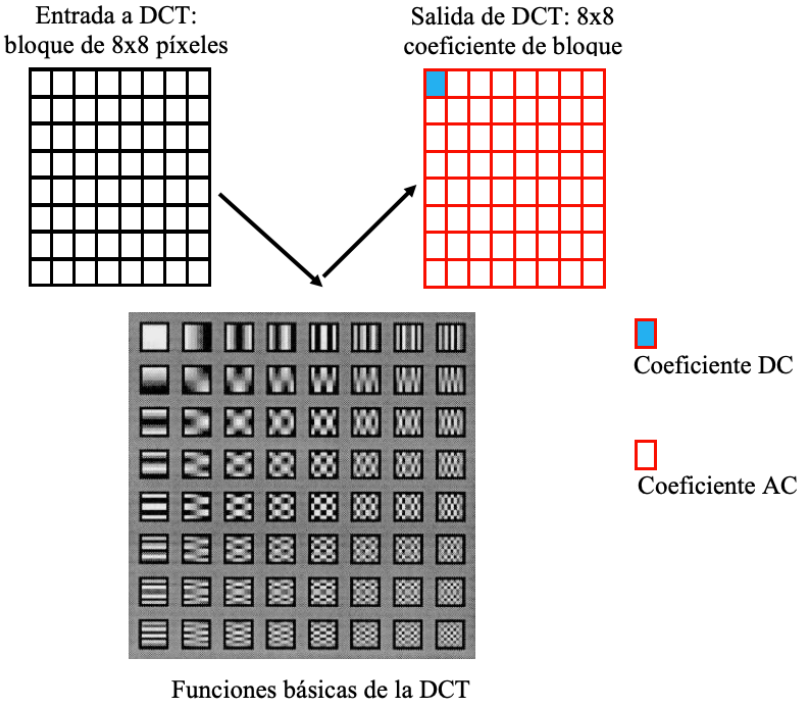


Figura 4: Diagrama de la DCT [141]

El proceso de descomposición mostrado en la figura 5, donde el coeficiente DC (muestra de valor promedio) representa el color promedio de la región 8×8. Los 63 coeficientes de AC (coeficientes con frecuencias no-cero) representan el cambio de color en todo el bloque.

El método DWT se aplica como una transformada matemática que permite la descomposición de imágenes basándose en transformación de pequeñas ondas llamadas ondículas de diferentes frecuencias. Los métodos en el dominio de la frecuencia son aprovechados no solo en imágenes digitales sino también en audio y en vídeo ([40], [41]). Las sub-bandas que genera esta transformada están espaciadas de forma logarítmica en frecuencia y representan la descomposición de banda de octava.

La sub-banda LL1 (conformada por LL2, HL2, LH2 y HH2) representa los componentes de baja frecuencia en sus posiciones horizontal y vertical de la imagen. La sub-banda HH1 representa los componentes de alta frecuencia de las posiciones horizontal y vertical de la imagen. La sub-banda LH1 contiene los componentes horizontales y verticales de alta frecuencia. Finalmente, la sub-banda HL1 contiene los componentes horizontales y verticales de alta y baja frecuencia.

Los métodos anteriores son los más comunes y utilizados en esteganografía, y a partir de ellos se han generado métodos que proponen diversas formas de incrustar datos en imágenes de portada, como ejemplo se tienen los siguientes.

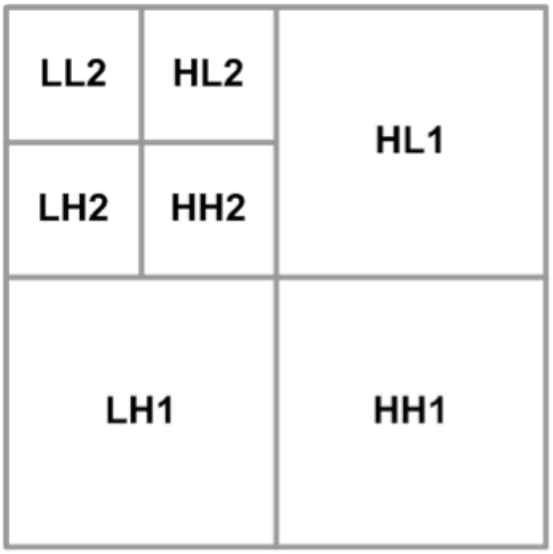


Figura 5: Diagrama de la DWT [40]

El algoritmo de **Watershed** se emplea para la segmentación de imágenes. Es empleado para encontrar determinadas regiones de las imágenes. Generalmente, se elige para imágenes que contienen texturas homogéneas y cuyo gradiente de intensidad es cercano a 0. Cabe señalar que es altamente utilizada en procesamiento de imágenes médicas, para el análisis de manchas de proteínas que se observan con técnicas de captura en geles de dos dimensiones [88].

BBPVD es una modificación del método de esteganografía PVD BB (PVD Blocks Based) cuya modificación radica en tomar bloques de dos píxeles con referencia de bloques de cuatro y ocho píxeles para incrementar la capacidad de incrustación de datos. Lo anterior mejora la capacidad de ocultamiento de información en la imagen de portada y la calidad general de la estego-imagen [72].

El método **BPCS** (*Bit Plane Complexity Steganography*) es un método de estenografía alternativo a los métodos de esteganografía como LSB. Sus autores muestran que la modificación de nitidez de la imagen permite incrustar significativamente más información y que los planos de bits para imágenes a escala de gris son mejores que los planos de bits binarios. Además, este método permite que los datos se oculten de forma aleatoria mediante una función de compresión para elevar la dificultad de ser localizadas por herramientas de estegoanálisis [80].

El método de esteganografía **EMD** (*Exploiting Modification Direction*) consiste en que cada dígito secreto en un sistema notacional del tipo $(2n+1)-ary$, se entiende como cualquier conjunto de mapas $f : G_n- > G$ de la n-ésima potencia de un plano cartesiano de G a G, es llevado por n píxeles en la imagen portada, donde n es un parámetro del sistema que incrementa o decrementa una unidad por píxel. Cada grupo de n píxeles contiene $2n$ posibles formas de modificación por lo que las n formadas de la modificación se suman, esto permite formar valores diferentes de un dígito secreto $(2n$ $+1)$ [167].

El método **TSM** (*Two-sides match*) consiste en usar la información lateral superior e izquierda de la imagen en los píxeles vecinos para ayudar a la estimación para la incrustación de modificaciones en ella. La estego-imagen está incrustada en el orden del barrido de la señal, a excepción de los píxeles de la primera fila y columna, como se muestra en la figura 6 [30].

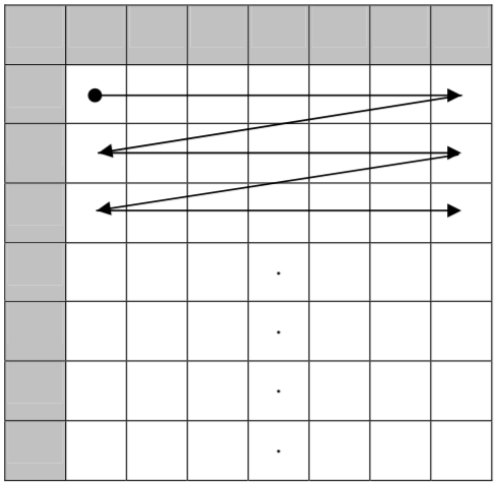


Figura 6: Diagrama de TSM [30]

2.2 Imagen digital

Una imagen digital es una función bidimensional $f(x, y)$, donde x y y representan la ubicación de un elemento de la imagen, mientras que f representa la función de intensidad en las coordenadas (x, y). Cuando los valores de x, y y la función de intensidad f son discretos, se dice que la imagen es una imagen digital. Una imagen digital es representada matricialmente a través de un número finito de elementos llamados píxeles. Cada píxel representa un valor numérico (intensidad), para imágenes en escala de gris, la intensidad del píxel está comprendido en el rango $[0, 255]$; para las imágenes en color, se almacenan tres valores (que representan la cantidad de rojo (R), verde (G) y azul (B)). Si una imagen tiene solo dos intensidades, entonces la imagen se conoce como imagen binaria [144].

2.2.1 Formatos de imágenes utilizados en esteganografía

Existen diferentes formatos utilizados en la esteganografía, a continuación, se enlistan los formatos que se utilizan en esta investigación.

- JPEG, es la extensión para archivos JPEG (Joint Photographic Experts Group), usa la compresión con pérdida.
- TIFF (*Tagged Image File Format*), es un formato sin perdidas, se aplica para imágenes desde 1 bit de color hasta 48 bits en modelos RGB, CMYK (*Cian, Magenta, Yellow and Key*), LAB (*Abreviación del modelo CIELAB*) o color indexado.
- GIF (*Graphics Interchange Format*), descarta todos los datos tipo exit, usa compresión LZW sin pérdidas, puede tener una paleta de colores de 24 bits, ofrece transparencia y animación. El color indexado se describe en la paleta de colores.
- PNG (*Portable Network Graphics*), utiliza compresión LZW, soporta transparencias y es un formato sin perdida por compresión, soporta imágenes a color con paletas de colores de 24 bits.

A continuación, se presentan los modelos de color más usuales en el área de imágenes [98].

- RGB (*Red, Blue, Green*). Es uno de los modelos de color más populares, se empezó a utilizar en monitores de rayo catódicos, consistía en combinar tres ases de luz que despedían los colores rojo, azul y verde, que combinados en distintas intensidades generaban los colores que se querían presentar a través de la variación de los voltajes, este modelo es ampliamente aceptado en sistemas operativos, formatos de imagen y lenguajes de programación. El modelo RGB presenta como limitación cuando el color no depende de la pantalla utilizada para crearlo o mostrarlo.

- CMYK. Cuando se agregan los tres colores primarios en proporciones iguales, se obtiene un color blanco, al restar proporciones iguales de los colores primarios, se puede crear un color negro. Para lograr esto se necesitan colores cian, magenta y amarillo, que son complementarios a RGB. El modelo CMY resultante es un complemento directo del modelo RGB. La conversión de RGB a CMYK viene dada por las siguientes ecuaciones: $C = 255 - R; M = 255 - G; Y = 255 - B$.

- YIQ (*In fase quadrature*). Se utiliza en transmisión de programas de televisión donde el color es una suma a la información proporcionado por la señal $Y : Y = g1 \times R + g2 \times G + g3 \times B$ donde $g1 + g2 + g3 = 1{,}0$, además se tienen dos vías adicionales de señal de ancho de banda que transportan la información de color en forma de diferencia ponderada entre la señal real y el componente Y.

- HLS (*Hue-Luminosity-Saturation*). Para generar los colores, se debe de elegir un color puro (*hue*), se incrusta un pigmento blanco para que pierda individualidad (saturación), posteriormente se agrega un pigmento negro, para que no sea muy brillante (luminosidad), los colores puros se encuentran en el borde exterior de un círculo de color horizontal, el tono se interpreta como un ángulo polar, que pasa por 0 grados para el rojo, 120 grados para el verde y 240 grados para el azul a rojo pasando de cualquier punto dentro del círculo representa algún color. Al estar más cerca del centro se aproxima más al color blanco. Existen otros círculos colocados de forma cilíndrica para indicar que cuanto más bajos son, más oscuros se vuelven. La coordenada Hue se describe con un ángulo entre 0 y 360 grados, la luminosidad (eje vertical) y la saturación (distancia radial desde el eje de luminosidad) con valores entre 0.0 y 1.0. La saturación de color máxima posible depende de la luminosidad. A medio camino de la parte superior ($L = 0.5$), la saturación máxima (en otras palabras, el radio del círculo) es $S = 1.0$. El color rojo, por ejemplo, tiene coordenadas como $H = 0$ grados, $L = 0.5$ y $S = 1.0$.

- HSV (*Hue-Saturation-Value*). Es similar a HSL solo que, en lugar de luminosidad, la coordenada vertical recibe el nombre de valor.

- HVC (textitHue-Value-Chroma). Se basa en un conjunto de muestras, cuya producción se controla cuidadosamente para garantizar la repetibilidad. Las muestras se han seleccionado de tal manera que, desde el punto de vista de la visión humana, muestrean los colores visibles de forma equidistante. Es bueno en los casos en que se solicita una impresión subjetiva del color, pero menos útil en el caso de los sistemas técnicos.

El grupo de modelos de color HLS, HSV y HVC permite mostrar mejor la visión del color al ojo humano que modelos como RGB, CMYK y YIQ, los cuales fueron desarrollados para aplicaciones de hardware, mientras que los primeros fueron desarrollados específicamente para ser apreciados por el ojo humano [71].

2.3 Preprocesamiento de datos: métodos de compresión de datos y teoría fractal

Existen dos clasificaciones en el área de compresión de datos, métodos con pérdida de datos y métodos sin pérdida de datos.

Los métodos de compresión con perdida tienen mayor eficiencia para reducir el tamaño de una imagen en cantidad de bits, por lo tanto, son ampliamente utilizados, además de que, la imagen que es comprimida es similar a la imagen original.

Las principales consideraciones de rendimiento de un esquema de compresión con pérdida incluyen [125].

- Relación de compresión.
- Relación señal/ruido.
- Velocidad de codificación y decodificación.

A continuación, se mencionan los métodos que se utilizan compresión con pérdida de datos [59].

- Transformar la codificación. Comienza dividiendo la imagen original en bloques generalmente de 8×8 píxeles. Se calculan lo coeficientes de transformación para cada bloque de píxeles convirtiendo la matriz original de 8×8 píxeles en una matriz de coeficientes con valores en la esquina superior izquierda. La esquina superior izquierda contiene la mayor parte de la información necesaria para cuantificar y codificar la imagen con baja distorsión perceptual. Los coeficientes resultantes se cuantifican y la salida del cuantificador se obtiene mediante métodos de codificación de símbolos para producir el flujo de bits de salida, representando con ello la imagen codificada. Para decodificar se desarrolla el proceso inverso.
- Cuantización de vectores. Se desarrolla un diccionario de vectores fijos que forman bloques de valores de píxeles. La imagen se divide en bloques que no se solapan y son denominados vectores de imagen. Cada bloque está representado en el diccionario y sus índices se utilizan para codificar la imagen original, la imagen está representada por la secuencia de índices que permita codificar con mayor entropía.
- Compresión fractal. En determinadas imágenes ciertas secciones se repiten en otras partes dentro de la misma, lo cual permite tomar las formas geométricas en datos matemáticos llamados códigos fractales que se utilizan para recrear la imagen codificada. Una vez que una imagen se ha convertido en código fractal se pierde su relación con una resolución específica y se convierte en una resolución independiente.
- Codificación de truncamiento de bloque. La imagen se divide en bloques de píxeles que no se solapan. Para cada bloque se determinan los valores de umbral y reconstrucción. El umbral suele ser la media de los valores de píxel en el bloque. Posteriormente se obtiene un mapa de bits del bloque reemplazando todos los píxeles por un valor de 1 o 0. En cada segmento (grupo de 1s y 0s) en el mapa de bits se determina el valor de reconstrucción, éste es el promedio de los valores de los píxeles correspondientes en el bloque original.
- Codificación de sub-banda. La imagen se analiza para producir los componentes que contienen frecuencias en bandas bien definidas. Posteriormente, la cuantificación y la codificación se aplica a cada una de las bandas. La ventaja de este esquema es que la cuantificación y codificación adecuadas para cada una de las sub-bandas se pueden diseñar por separado.

En los métodos de compresión sin pérdida se recupera la información sin pérdida de datos de la imagen comprimida, también se le conoce como señal silenciosa porque no añade ruido y se utilizan métodos de descomposición para minimizar la redundancia [25]. Los siguientes métodos están incluidos en la compresión sin pérdidas.

- Codificación de longitud. Reemplaza las secuencias de píxeles idénticos por símbolos de menor longitud. Para imágenes en escala de gris se representan como una secuencia V_i, R_i donde V_i es la intensidad del píxel y R_i se refiere al número de píxeles consecutivos con la intensidad V_i. Si tanto V_i como R_i están representados por un byte en un conjunto de 12 píxeles se codifica utilizando ocho bytes, produciendo una relación de compresión de 1:5 [69].

- Codificación RLE (*Run-Length Enconding*). Si un elemento de datos d aparece n veces consecutivas en el flujo de entrada, se reemplazan las n apariciones con el par único nd. Las n ocurrencias consecutivas de un elemento de datos se denominan una longitud de ejecución de n [104].

- Codificación de Huffman. La codificación de Huffman se ha utilizado en texto, imagen, compresión de vídeo y sistema de conferencia como JPEG, MPEG-2, MPEG-4 y H.263, entre otros, esta codificación se encarga de recopilar símbolos únicos de la imagen de origen y calcula su valor de probabilidad para cada símbolo ordenando los símbolos en función de su valor de probabilidad. Para este tipo de codificación se genera un árbol binario, el cual se construye clasificando los símbolos con menor y mayor probabilidad de aparición [52].

- Codificación LZW (*Lempel-Ziv-Welch*). Se basa en un diccionario el cual puede ser estático o dinámico. En la codificación de diccionario estático, el diccionario se prepara durante los procesos de codificación y decodificación. En la codificación dinámica, conforme se encuentran nuevos símbolos el diccionario se va actualizando automáticamente [162]. Un proceso de compresión por LZW, se puede describir de la siguiente forma: sean m y n números enteros. Sea $w_1, ..., w_n$ un conjunto de secuencias de tamaño m. Solo hay tres formas diferentes de organizar los datos procedentes de M_i de la secuencia $w_1, ..., w_n$ si no se permite que w_i sea una secuencia inmediatamente posterior a w_j en tres de las diferentes cadenas de secuencias para cada par de $1 \leq i \leq n$ y $1 \leq j \leq n$.

- Codificación aritmética. Define un método que proporciona palabras de código con una longitud ideal. Este método requiere conocer la probabilidad de la aparición de los símbolos individuales. La longitud promedio del código es muy cercana al mínimo posible dado por la teoría de la información, se asigna un intervalo a cada símbolo cuyo tamaño refleja la probabilidad de que aparezca este símbolo. La palabra clave de un símbolo es un número racional arbitrario que pertenece al intervalo correspondiente [120].

- Codificación de área. Busca encontrar regiones rectangulares con las mismas características. Estas regiones están codificadas en forma descriptiva como un elemento con dos puntos y una estructura.

Un fractal es una imagen o estructura que puede ser generada por computadora, el cual se definen como un objeto con figura geométrica en donde su estructura básica se encuentra fragmentada y se repite en varias escalas, término propuesto por Benoit Mandelbrot [125]. Una de las propiedades principales que presenta el fractal es que su dimensión métrica es representada por un número fraccionario.

Los fractales se pueden representar como **conjuntos matemáticos** cuyos patrones son similares entre sí y pueden ser exactamente iguales en todas las escalas. Estos difieren de las figuras geométricas regulares por su escalado dimensional fractal. Los fractales tienen una dimensión fraccionaria que generalmente excede su dimensión topológica y está comprendida entre números enteros y número fraccionarios. La teoría de sistemas dinámicos está ligada con la geometría fractal, los atractores de fractales de sistemas iterativos tienen un sistema dinámico naturalmente asociado, el cual es caótico [2] debido a que variaciones infinitesimales en sus condiciones iniciales pueden generar cambios importantes en ellos.

La dimensión topológica del espacio euclidiano n-dimensional es N_1. Es una dimensión entera, que describe objetos geométricos. La dimensión topológica de un punto es igual a 0, la dimensión topológica de una línea o curva es igual a 1, la dimensión topológica de un área es igual a 2. La dimensión topológica determina el número mínimo de parámetros necesarios para determinar con precisión la posición de un objeto en el espacio.

Una dimensión fractal indica el nivel de segmentación de un objeto utilizando una dimensión no entera. La forma de una red de valles está formada por líneas incrustadas en el plano, y la dimensión fractal describe hasta qué punto se llena el espacio en el plano de la línea, alcanzando así valores en el intervalo abierto (1, 2).

La dimensión fractal es un término sugerido por Felix Hausdorff en 1919, la cual especifica una propiedad de un objeto que indica la capacidad de cubrir el espacio en el que este distribuido y puede tomar valores continuos en un espacio de números reales, que van de 0 a 3. La dimensión de Hausdorff permite explicar cómo estructuras biológicas son capaces de ocupar espacios de volumen determinado, este tipo de aprovechamiento lo podemos observar en estructuras de órganos como pulmones, corazón, entre otros [48]. La dimensión fractal se calcula a través de la ecuación (3).

$$D_{MB} = \lim_{\varepsilon \to 0} \frac{\log N_1(\varepsilon)}{\log \frac{l}{\varepsilon}} \tag{3}$$

Donde D_{MB} es la dimensión Minkowski-Bouligand para espacios geométricos fractales, l es el número de dimensión, N_1 es el número de objetos autosimilares (cuyo patrón base se repite a distintas escalas), y ε es el lado lineal.

Existen dos clases de fractales, las cuales son:

- Los fractales lineales son aquellos que se construyen con un simple cambio en la variación de sus escalas y son exactamente idénticos en todas sus escalas hasta el infinito [14].
- Los fractales no lineales, en cambio, son aquellos que se generan a partir de distorsiones complejas y usando un término proveniente de la matemática caótica, distorsiones no lineales [146]. La mayoría de los objetos fractales puramente matemáticos y naturales son no lineales.

Una de las aplicaciones de los fractales es en la creación de antenas para celulares, las cuales tienen la capacidad de conectar a distintas redes de telecomunicaciones [38].

El estudio de teoría de fractales ha permitido avances para la computación, lo cual se puede apreciar en la generación de paisajes artificiales, tanto para simulaciones como para videojuegos, o escenarios de películas, sin procesos costosos que permitan reproducir modelos de la realidad, como

lo demostró Benoit Mandelbrot [95].

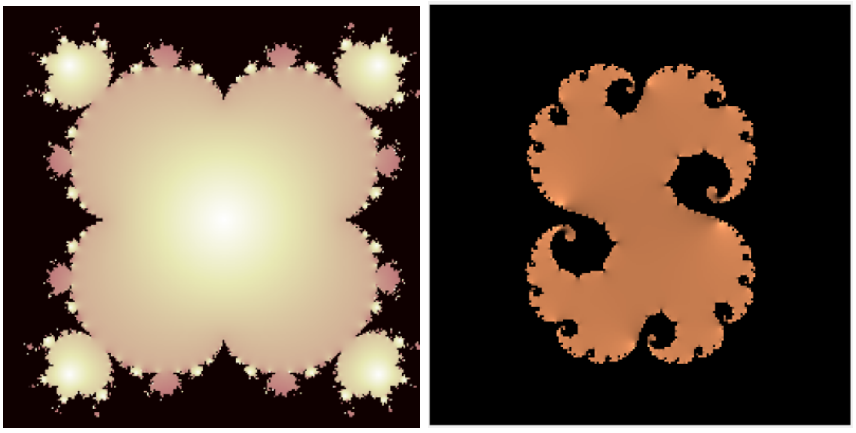
Entre los ejemplos de fractales más conocidos se tiene a los conjuntos matemáticos (fractales) Mandelbrot y Julia [2] y se presentan a continuación.

Mandelbrot. Es una función matemática de puntos que se encuentran localizados en el plano complejo, y el fractal es formado a través del borde de este. Este conjunto se construye a través de la ecuación (4), donde c es un número complejo, z_0 es el termino inicial y z_{n+1} es la relación de inducción. La figura 7 (a) representa el conjunto de Mandelbrot [121].

$$f(z,c) = \left\{ \begin{matrix} z_0{=}0 \\ z_{n+1}{=}z_n^2{+}c \end{matrix} \right\} \tag{4}$$

Julia. El conjunto matemático de Julia, permite obtener fractales a través de funciones cuadráticas como $f_c(z) = z^2 + c$, donde c es un número complejo. Sí el resultado de la ecuación anterior queda acotado entonces z pertenece al conjunto de Julia con parámetro c, denotado por J_c, de lo contrario z queda fuera de este. La ecuación (5) presenta el conjunto de Julia y gráficamente se visualiza en la figura 7 (b) [94].

$$f_c(z) = \left\{ \begin{matrix} z_0{=}z \\ z_{n+1}{=}z_n^2{+}c \end{matrix} \right\} \tag{5}$$



(a) Mandelbrot (b) Julia

Figura 7: Ejemplos de fractales

Existen otros fractales de gran importancia, entre ellos está el triángulo de Sierpinski, en honor al matemático W. Sierpinski. En este fractal generalmente se construye empleando un triángulo equilátero, el cual es dividido en cuatro regiones para unir los puntos medios de los datos del triángulo base, y posteriormente se elimina el triángulo central. Para cada triángulo restante el proceso se repite mediante la división y eliminación, como se hizo en el triángulo base [35]. Un ejemplo gráfico del triángulo de Sierpinski se muestra en la figura 8. En la ecuación (6) se presenta la forma general del triángulo de Sierpinski.

$$T = 3^{k-1} \tag{6}$$

Donde T es el número de triángulos, k es el número de iteraciones y $k \geq 1$.

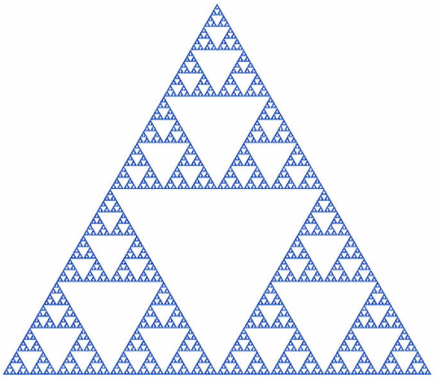


Figura 8: Triángulo de Sierpinski [137]

Otro fractal importante es la alfombra de Sierpinski, y tienen un procedimiento de construcción similar al del triángulo de Sierpinski, el cual consiste en tomar un cuadrado como base, y dividiendo en nueve cuadrados de menor tamaño, posteriormente cada uno de lado con una longitud $1/3$ y se elimina el cuadrado central. El proceso se repite dividiendo cada cuadrado en otros nueves cuadrados, cada uno con una longitud de un tercio y se vuelve a eliminar el cuadrado central. En la figura 9, se muestra un ejemplo de alfombra de Sierpinski [15].

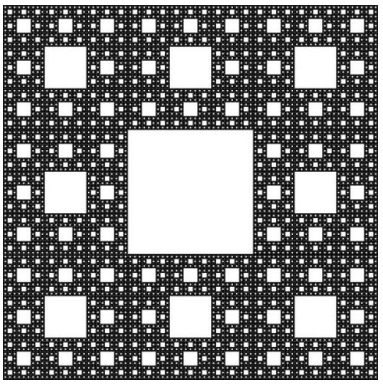


Figura 9: Alfombra de Sierpinski [6]

Para obtener una representación gráfica de la alfombra de Sierpinski, sea N_n el número de cajas negras, L_n la longitud de un lado de una caja blanca y A_n el área fraccionaria de las cajas negras después de la enésima iteración. Lo anterior está representado en las ecuaciones (7), (8) y (9).

$$N_n = 8^n \tag{7}$$

$$L_n = 3^{-n} \tag{8}$$

$$A_n = L_n^2 N_n = \left(\frac{8}{9}\right)^n \tag{9}$$

Una de las partes interesantes de los fractales es que su dimensión topológica no es exacta como lo podrían ser las figuras geométricas como los cuadrados, pentágonos, entre otros. Un ejemplo de lo anterior es que tomando en cuenta las ecuaciones (7) y (8) se tiene que el número de celdas de la alfombra de Sierpinski después $n = 0, 1, 2, ...$, y de las iteraciones dadas en 1, 8, 64, 512, 4096, 32768, 262144, en la ecuación (9) se tiene que su capacidad de dimensión fraccionaria esta expresada en la ecuación (10) [155].

$$D_{MB} = -\lim_{n\to\infty} \frac{\ln N_n}{\ln L_n} \tag{10}$$

Teniendo en cuenta la ecuación (10). La dimensión D_{MB} calculada para la alfombra de Sierpinski es ≈ 1.892789260.

2.4 Métricas de validación de calidad para imágenes

Dentro del tratamiento de imágenes al momento de realizar una serie de operaciones sobre ellas existen deformaciones inevitables dentro de su estructura interna, por tal motivo se han diseñado métricas para medir los errores que se generan. En los siguientes párrafos se muestran las métricas de mayor uso en la literatura.

El histograma de una imagen digital con niveles de intensidad en el rango $[0, L-1]$ es una función discreta $r_k = n_k$ donde r_k es el kth valor de intensidad y n_k es la frecuencia del píxel en la imagen. Es una práctica común normalizar un histograma dividiendo cada uno de sus componentes por el número total de píxeles en la imagen, denotado por el producto de $M \times N$, que corresponde al número de píxeles que contiene la imagen. Así, una normalización del histograma está dada por $p(r_k) = n_k/M \times N$ para $k = \{0, 1, 2, ..., L-1\}$, por lo tanto $p(r_k)$ es el estimado de la probabilidad de concurrencias del nivel de intensidad r_k en una imagen. La suma de todos los componentes de un histograma normalizado es igual a 1 [57].

Una de las métricas desarrolladas para comprobar la integridad estructural de una imagen es el **MSE** (*Mean Squared Error*) [125], el cual se define como error cuadrático medio, donde $f(x, y)$ es una señal portadora (imagen de portada), $\hat{f}(x, y)$ es una señal procesada (estego-imagen), $M\times N$ es el tamaño de la señal en 2D [53]. La ecuación (11) representa el cálculo del MSE.

$$MSE = \frac{1}{M \times N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[f(x, y) - \hat{f}(x, y) \right]^2 \tag{11}$$

La relación existente entre la señal y el ruido denominada como **SNR** (*Signal to Noise Ratio*) se establece entre la proporción que existe entre la señal de la potencia que se transmite y la potencia de la señal de ruido que la descompone [20]. Una señal SNR al alcanzar entre los 35 dB y 38 dB la imagen presenta sus detalles originales (sin efectos de distorsiones apreciables), mientras que de 30 dB a 35 dB la calidad la imagen pierde cualidades visuales, y un valor menor 30 dB la calidad es deficiente. Esta relación se mide en decibeles y se define por la ecuación (12).

$$SNR = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}(x, y)^2}{MSE} \tag{12}$$

Una de las métricas con mayor uso en la literatura para determinar la calidad de la imagen es el **PSNR** (*Peak Signal to Noise Ratio*) que se define como un límite, en donde se aproxima la relación con el receptor de errores a través del sistema de visión humano. Un PSNR mayor a 30 dB implica que la semejanza entre la imagen de portada y la imagen reconstruida es alta. El PSNR es adimensional, ya que las unidades tanto del numerador como del denominador son valores de píxeles [42]. Se debe de tener en cuenta que al obtener una imagen 45 dB de PSNR, normalmente, una persona no puede notar la diferencia, en un rango de 40 dB a 45 dB la imagen presenta visualmente la mayor parte de sus características visuales, de 38 dB a 40 dB la calidad es suficientemente aceptable y puntajes menores a 30 dB indican que la calidad es deficiente. Debido al uso del logaritmo en el PSNR, este se expresa en decibeles y se define por la ecuación (13).

$$PSNR(dB) = 10log_{10}L^2/MSE \tag{13}$$

El termino **BER** (*Bit Error Rate*) define el número de bits erróneos por unidad de tiempo, esta métrica también es utilizada para analizar los errores que puede contener una imagen. El radio del bit de error es el número de bits erróneos (TNEB) divididos por el total de números de bits transferidos (TNB) durante un intervalo de tiempo fijo [150]. La ecuación (14) permite definir el BER, el valor obtenido se representa en porcentaje.

$$\%BER = \frac{TNEB}{TNB} \tag{14}$$

La métrica **SSIM** (*Structural Similarity Index*) es un método utilizado para determinar la similitud entre dos imágenes, esta métrica permite medir o predecir la calidad de la imagen, basándose en una imagen inicial no comprimida o sin distorsión (imagen de portada) [153]. Generalmente, se utiliza el índice MSSIM (SSIM medio) para evaluar la calidad general de una imagen. $f(x, y)$ representa a la imagen portada y $\hat{f}(x, y)$ a la imagen distorsionada, f_j y $\hat{f}_j$ son el contenido de la ventana local jth, y W es el número de ventanas locales de la imagen. La ecuación (15) representa a MSSIM [152].

$$MSSIM\left(f(x, y), \hat{f}(x, y)\right) = \frac{1}{W}\sum_{j=1}^{W} SSIM\left(f_j, \hat{f}_j\right) \tag{15}$$

En la métrica SSIM los resultados indican que cuando tenemos de 1 a 0.95 puntos, la calidad de la imagen no presenta errores visualmente, de menores a 0.95 hasta 0.90 puntos se pueden apreciar diferencias con las imágenes de portada, y por debajo de 0.90 puntos es deficiente la calidad de la imagen.

La prueba χ^2 de Pearson también conocida como chi-cuadrada es una prueba no paramétrica que mide la diferencia entre una distribución observada y una teórica, con el objetivo de definir las diferencias existentes entre ambas. Se utiliza también para probar la independencia de dos variables entre sí. La ecuación (16) define el cálculo de la prueba de Pearson [112].

$$\chi^2 = \sum \frac{(f_0 - f_e)^2}{f_e} \tag{16}$$

Donde f_0= frecuencia del valor observado y f_e= frecuencia del valor esperado.

La correlación es un vínculo recíproco que existe entre dos o más elementos, estadísticamente se refiere a la proporcionalidad y la relación lineal entre variables [13]. Se dice que, si los valores que toma una variable son modificados de forma sistemática con respecto a los de otra, entonces están correlacionadas.

La correlación cruzada es una medida de similitud entre dos señales, frecuentemente empleada para encontrar características relevantes en una señal desconocida por medio de la comparación con otra que si se conoce. En estadística, el término correlación cruzada es usada para referirse a la covarianza $cov(X, Y)$ [118].

Coeficiente de correlación. El coeficiente de correlación de dos variables aleatorias es una medida de dependencia lineal [22]. Si cada variable tiene n observaciones escalares, el coeficiente de

correlación de Pearson se define como en la ecuación (17). Donde A representa la primera variable y B la segunda variable, siendo m y n las secciones a evaluar entre las variables aleatorias [128].

$$r = \frac{\sum_m \sum_n \left(A_{mn} - \overline{A}\right)\left(B_{mn} - \overline{B}\right)}{\sqrt{\left(\sum_m \sum_n \left(A_{mn} - \overline{A}\right)^2\right)\left(\sum_m \sum_n \left(B_{mn} - \overline{B}\right)^2\right)}} \tag{17}$$

La entropía es una medida estadística de aleatoriedad que se puede utilizar para caracterizar la textura de la imagen de entrada. Shannon define la entropía como una medida de incertidumbre de la información contenida en un sistema. La entropía se define por la ecuación (18), donde p contiene la suma de los datos de comparación entre la imagen de portada y la estego-imagen [61].

$$\Delta S = -\sum (p * \log 2(p)) \tag{18}$$

La desviación estándar se representa como σ y es una medida que se utiliza para cuantificar la variación o la dispersión de un conjunto de datos numéricos. Un valor bajo para la desviación estándar indica que la mayor parte de los datos se encuentran agrupados cerca la media de un conjunto de datos, por otro lado, si la desviación estándar es elevada, lo datos están extendidos en un rango amplio [7]. La ecuación (19) presenta la desviación estándar.

$$\sigma = \sqrt{\frac{\Sigma |x - \mu|^2}{N_2}} \tag{19}$$

Donde Σ significa la suma de x es un valor de un conjunto de datos, μ es la media del conjunto de datos y N_2 es el número de datos.

De las métricas antes expuestas las que generalmente se utilizan son PSNR y MSE, debido a que permiten determinar la calidad de una imagen, las más modernas son SSIM y chi-cuadrada. La métrica que ayuda a observar que existen cambios en la imagen de portada con respecto a la modificada es CC (Correlación Cruzada) debido a que permite observar la relación que existe en la imagen de portada y la estego-imagen desde el punto de vista de la relación que existe entre los valores originales y los valores modificados.

2.5 Métodos de estegoanálisis

El estegoanálisis consiste en detectar si una imagen contiene información oculta, por lo tanto, es el proceso inverso al que realiza la esteganografía. Existen dos tipos de estegoanálisis los cuales son dedicados y universales, los primeros se definen como detectores de espacio debido a que buscan un espacio de exploración para encontrar las modificaciones que generó un método de esteganografía empleado, y su principal ventaja es que su porcentaje de detección es mayor que los universales [91], el problema es que se debe conocer el método que fue utilizado para el proceso de ocultamiento. Los detectores universales permiten detectar más de dos tipos de métodos esteganográfico, desde una perspectiva espacial o de la frecuencia.

En estegoanálisis se tienen dos enfoques los cuales son activo y pasivo, el primero consiste en modificar los elementos que se transportan en el canal de comunicación, de tal forma que no afecte la visualización de la imagen y que recupere la información cuando sea posible. El enfoque pasivo consiste en analizar los objetos por medio de una función de detección, con una función de probabilidad que determine si es una estego-imagen o no [114].

Chandramouli [29] presenta una clasificación de los métodos de estegoanálisis como se muestra a continuación.

- Basado en aprendizaje supervisado. Consiste en detectar si un objeto contiene información oculta o no, estos pueden trabajar a través de redes neuronales, árboles de toma de decisiones, lógica difusa, etc. Este tipo de estegoanálisis requiere de un entrenamiento previo, esto da como resultado que con un conjunto de clasificadores adecuados se puede resolver satisfactoriamente una tarea, o puede ser que al equivocarse la elección del clasificador falle en la tarea.
- Identificación ciega (*blind identification based steganalysis*). Se basan en la estadística para detectar cambios en el objeto portador, como puede ser Spread Spectrum.
- Métodos estadísticos. Consiste en aplicar mecanismos estadísticos para detectar anomalías entre los componentes de la imagen.
- Basados en combinación de varios métodos. Se basan en combinar varios métodos para tratar de obtener mejores resultados.

Entre los métodos de estegoanálisis más comunes se tienen las siguientes.

- Chi-cuadrada. Evalúa los pares de valores de píxeles que son iguales bit a bit a excepción del bit menos significativo. Si los bits conservan una distribución uniforme, la frecuencia de los valores siguiente será la misma, de esta forma se observan las frecuencias esperadas con las obtenidas para evaluar si existen cambios detectables en la imagen [156].
- Umbralización (*thresholding*). Segmenta las imágenes escalares para crear una partición binaria de las intensidades de la imagen, esto se logra agrupando los píxeles con mayor intensidad, tomando como punto de partida el umbral de una clase determinada de píxeles y los demás píxeles en otra clase distinta [168].
- Región creciente. Permite extraer regiones de la imagen que están conectadas entre sí, se les denomina regiones adyacentes, y requiere de un píxel semilla que de la facilidad de detectar los cambios a través de un sistema matemático [105].
- Clasificadores. Los métodos clasificadores es un rango espacial de cualquier función de la imagen, siendo las intensidades de la imagen el más común de los espacios característicos [105].
- RS (*Regular- Singular*). Este método parte de una función f que mide la suavidad de los cambios de píxeles de una imagen, modificada por LSB, el valor aumenta al modificar el bit menos significativo de cada píxel, debido a que las diferencias se medirán de forma estadística entre los píxeles adyacentes. R y S son bloques que están relacionados entre sí y que aumentan el valor de f. Los bloques regulares son los que representan valores iguales según una función de estimación, y los bloques singulares son los que representan un cambio no esperado [51].

El proceso de la esteganografía es complejo, debido a que se vale de una gran cantidad de mecanismos para generar métodos que permitan el transporte de información de forma desapercibida, por lo tanto se busca lograr altas cargas de incrustación, y cuya primera premisa es evitar distorsiones en los objetos portadores donde va oculta la información, por otra parte, se debe de garantizar que la información no sea recuperada por parte de un atacante y que información sea resistente a ataques de diversos tipos.

En la sección 2.6 se presentan investigaciones actuales sobre esteganografía, tanto para imágenes a escala de gris y modelos de color, en las cuales se aplican distintas perspectivas sobre el dominio espacial y de la frecuencia o combinaciones de estas perspectivas.

2.6 Revisión del estado del arte

Los métodos de esteganografía aplicados en archivos digitales generalmente trabajan con archivos de audio, video e imágenes, debido a que permiten la incrustación de información sin dejar rastro desde la perspectiva de la visión humana, además de que permiten altas cargas de datos sobre todo cuando se trata de vídeo e imágenes.

Generalmente, el objeto digital que se emplea en esteganografía para el transporte de información oculta es la imagen digital, debido a que presenta una elevada flexibilidad pars este propósito. Esta sección está dividida en 3 subsecciones, la subsección 2.6.1 es una recopilación sobre métodos de compresión de datos, los métodos de compresión permiten reducir el tamaño de información así como generar que la información no sea directamente accesible, esto es aprovechado para incrementar la carga de datos que se transportan en un medio de comunicación, la subsección 2.6.2 es una recopilación de los trabajos cuya investigación está enfocada a la incrustación de datos en imágenes en escala de gris, en esta subsección se abordan métodos en el dominio del espacio y de la frecuencia, así como combinaciones de estos, finalmente, en la subsección 2.6.3, se abordan trabajos relevantes sobre incrustación de datos en imágenes con modelos de color, este tipo de imágenes se emplean los dominios espaciales o en dominio de la frecuencia, pero con la particularidad de que se busca aprovechar los canales adicionales para lograr mayores cargas de incrustaciones que las imágenes a escala de gris, aunque se debe de cuidar la variación de tonalidades en las imágenes, debido a que son en mayor proporción susceptibles a los cambios efectuados.

2.6.1 Trabajos de codificación y compresión

En la presente subsección se muestran trabajos relevantes en el área de compresión de datos para generar un análisis de los métodos que se emplean en la representación de mensajes, teniendo como punto de vista la reducción del mensaje y el transporte de ésta sin pérdida de datos.

Suarjaya [139] en el 2012, propone un nuevo algoritmo denominado JBE (J-bit encoding) enfocado en manipular los bits de los datos de un mensaje, con la finalidad de evitar perdida de datos seleccionando los conjuntos que no tienen repetición en las cadenas de datos, además lo combinan con métodos de compresión aritmética y RLE. Hosseini [65] 2012, analiza en su trabajo distintos métodos de compresión enfocándose en obtener cuales son el mayor consumo de recursos computacionales, donde el algoritmo LZW es uno de los que mejor comportamiento presenta en transmisión de datos por redes inalámbricas de computadoras.

En Wang et al. [151] en el 2013 presentan un trabajo enfocado a reducir el costo computacional para la compresión de datos que se transmiten en vídeo cuando se detecta movimiento, proponen un algoritmo de compresión sin pérdida para capturar secciones de video que presenten movimiento, mediante un método basado en la APP (*Alpha Parallelogram Predictor*) para estimar el DOF (*Grade of Free*) de los sujetos en movimiento, posteriormente encuentran el predictor del sistema mediante una de búsqueda, el método de compresión que utilizan es la compresión aritmética basada en contexto, logrando que no exista perdida de datos.

En Singh y Singh [133] en el 2013, presentan en su investigación un análisis de los métodos de compresión con perdida y sin pérdida de datos, en donde definen que las transformaciones BWT (*Burrows Wheeler Transform*) y MFT (*Move-to front Transform*) modifican el flujo de datos para que sea aprovechado por algoritmo de compresión. Los autores coinciden que la codificación aritmética arroja mejores resultados que la codificación Huffman, pero consume mayor tiempo en procesador. Los autores sugieren combinar BWT con RLE o Huffman para obtener compresiones

rápidas y eficaces. En Singh y Meenakshi [132] en el 2014 presentan un enfoque híbrido para la compresión de datos de texto mediante reducción dinámica de bits y codificación Huffman obteniendo ratios de compresión superiores a 53 %.

En Fajardo et a. [47] 2015, realizaron una investigación para la compresión de datos sísmicos mediante la transformada Wavelet aprovechando la capacidad de las ondículas de representación de señales sísmica, los autores emplean el método *Lifting* para representar la transformada de Wavelet 2D, empleando la codificación Huffman en vez de la codificación aritmética.

Kavitha [79] en el 2016, presentan un análisis sobre métodos de compresión con perdida y sin pérdida de datos, llegando a la conclusión de que los métodos de compresión sin pérdida de datos son mejores para objetos como texto, los cuales necesitan no perder datos y los métodos de compresión con pérdida de datos son mejores para archivos como audio y video, puesto que determinados componentes no son indispensables para la transmisión de este tipo de mensajes.

En Fritiya et al. [49] en el 2017 presentan el análisis de métodos de compresión como Huffman, Shannon Fano, Tunstall, Lempel Ziv Welch y codificación de longitud de ejecución y exponen los métodos que obtienen mejores resultados para la compresión de datos en forma de texto. La salida generada del archivo de compresión es más pequeña que el archivo original.

En Btoush y Dawahdeh [26] 2018, analizan la complejidad y la entropía de los métodos de compresión LZW, Huffman, CLV (*Código de Longitud Variable*), HFLC (*Huffman after using Fixed-Length Code*). Los autores concluyen que el algoritmo LZW obtuvo mejor comportamiento en relación con los demás algoritmos. Huang y Yuwen [66] en el 2018 presentan una investigación enfocada a la compresión de datos provenientes de sistemas de sensores múltiples que se utilizan en sectores agrícolas, militares, forestales y otros analizando el algoritmo DEFLATE, además combinan los algoritmos LZ77 y Huffman, para lograr encontrar las cadenas más largas para poder ser comprimidas.

De acuerdo con los trabajos anteriores se determina que los métodos que presentan mejor comportamiento en la compresión de texto sin pérdida de datos son LZW, Huffman y la compresión aritmética. Dichos métodos permiten representar no sólo símbolos derivados de cadenas de texto, si no también procedentes de otros tipos de archivos como video, audio, entre otros. Sin embargo, para éstos últimos la reducción de su tamaño no es el esperado, por lo que se sugiere el uso de métodos con pérdida de datos. Para la representación de datos de forma general es conveniente emplear codificaciones basadas ASCII como base 32 o 64, para evitar el uso de distintas bases de codificación.

2.6.2 Esteganografía para imágenes en escala de gris

Diversas investigaciones se han enfocado en la generación de algoritmos esteganográficos, entre ellos está el trabajo de Wu et al. [159] en el 2011, proponen un modelo adaptativo basándose en la diferencia de valores de píxeles y descomposición de esquemas, logrando baja distorsión en la imagen y obtienen mejores resultados que PVD, EMD, TSM, entre otros, mientras tanto las investigaciones de Lee et al. [89] en el 2011, se basan en la utilización de TPVD (*Try way Pixel Value Differencing*) para el ocultamiento de información, emplean imágenes en formato JPEG2000 (*Joint Photographic Experts Group 2000*) y logran incrustar imágenes en escala de gris, reduciendo el tamaño de los vectores, además de utilizar VQ (*Vector Quantization*) para recuperar el valor residual de la compresión generada por JPEG2000, demostrando que la aplicación de TPVD en la imagen de portada son imperceptibles dentro de la estego-imagen.

En el 2012 Gajendra et al. [54] trabajan con aplicaciones web para generar un algoritmo que emplea una clave como identificador único, además de aplicar el método BPCS (*Bit-Plane Complexity Segmentation steganography*) guardando los datos en una estego-imagen, con ello, se aseguran de que los datos se transporten de forma desapercibida y que se requiera forzosamente la clave única para su recuperación.

En el 2013 se propone el método BPIS (*Secure Block Permutation Image Steganography*) combinada con LSB en el trabajo de Al-Bahadili [3], con la finalidad de reforzar la seguridad de los datos ocultos. En el experimento se seleccionaron imágenes en formato BMP (*Windows bitmap*), y obtuvieron resultados del PSNR superiores a 40 dB. El método BIPS se basa en LSB para ocultar datos, pero se apoya en la generación de números aleatorios para modificar el orden en cómo se escribieron los datos en los bloques de la imagen, la información que oculta está codificada en ASCII.

En el 2013, Elgabar et al., [44] realizan un estudio de esteganografía en donde se comprueba que las imágenes con formato GIF (*Graphics Interchange Format*) obtienen mejores resultados que en JPEG (*Joint Photographic Experts Group*), sin embargo, los autores no especifican la cantidad de datos que se incrustan en las imágenes de prueba que presentan.

BBPVD trabaja tanto con imágenes en escalas de gris como de color, se basa en la diferencia del valor de los píxeles, y es presentada por Patil et al. [109] en el 2013, mediante la extensión de bloques basados en el algoritmo PVD para dos, cuatro y ocho píxeles. Los autores logran incrementar la capacidad de incrustación en el texto entre la estego-imagen y la imagen de portada sin diferencias visuales. Cabe resaltar que trabajos de esteganografía emplean distintos métodos de generación de secuencias aleatorias como se muestra en Jyoti et al. [72] en el 2014.

Nehete y Bhide [103] en el 2014, aplican el análisis de segmentación de texturas y color en imágenes, la base principal de la investigación está en el aprovechamiento de las variantes de los tonos de piel en un conjunto de rostros humanos, el método de esteganografía aplicado es DWT sobre el modelo YCbCr. En Kumar et al. [85] en el 2014 emplearon modificaciones del LSB, con bloques de 64×64 píxeles, como hardware de procesamiento gráfico se utilizó un FPGA (*Field Programmable Gate Array*). En el trabajo obtienen que las distintas tonalidades en los rostros humanos permiten incrustar información que pase desapercibida gracias al empleo de DWT por análisis estadístico, aunque: visualmente existe un efecto de cuadriculado en las imágenes.

Muhammad et al. [99] en el 2015, proponen el empleo del método M-LSB (*Modified Least Significant Bit*) y el algoritmo PBSA (*Pattern based Bits Shuffling Algorithm*), este algoritmo consiste en permitir convertir los datos secretos en representaciones de bits y los mezcla según un patrón específico con una estego-clave, ofreciendo que la extracción de los datos originales sea difícil para los atacantes. En sus resultados obtienen un promedio de 44.58 dB de PSNR en las estego-imágenes, además proponen una serie de métricas para el análisis de datos, siendo las principales NCC y SSIM.

Otros trabajos incluyen formas de segmentación de imágenes diferentes a las anteriores como en Puri y Deep [115], en el 2015 proponen una combinación de métodos de incrustación de datos aplicando eliminación de ruido de la imagen, posteriormente se segmenta la imagen con Watershed y se genera una búsqueda de valores en los píxeles, adicionando el cifrado RSA. Los valores de PSNR obtenidos en las estego-imágenes son superiores a 65 dB.

En Khandappalavar et al. [82], emplean la transformada de Arnold para poder incrustar datos a través del LSB. Un logro importante es que los datos alterados por LSB presentan alta resistencia

contra los ataques de tipo estático. La aplicación de la transformada permite el encriptado de los datos incrustados.

En Hernández et al. [64] en el 2015, toman como método de ocultamiento de datos el algoritmo TPVD, utilizando bloques de bits de 2x2 píxeles, dentro de sus experimentos realizan la comparación con el algoritmo propuesto por F. Peng [111], quien utiliza Wavelet Haar, y los resultados generalmente arrojan un PSNR promedio de 36 dB en comparación con los 30 dB promedio que se obtiene con la propuesta de F. Peng.

Diversas variantes del LSB se pueden observar en los estudios de Gulve y Joshi de 2015. [58], se desarrolla un método en donde mediante pares de bloques de píxeles se oculta información, la base del trabajo está desarrollada en LSB, el tamaño de los bloques empleados es de 2x3 píxeles. La validación de su algoritmo lo hacen a través de las diferencias en los histogramas de la imagen de portada y la estego-imagen, los cuales no reportan diferencia alguna. El nivel de PSNR generalmente es superior a 32 dB. Un punto por destacar es que se comparan con los trabajos de Wu [158], Chang [31] y Xin Liao [160] con resultados muy similares o ligeramente superiores ya sea en calidad o en capacidad para incrustar datos. En el trabajo de Atawneh et al. [11] del 2015, presentan una descomposición de la imagen en partes significativas en el dominio del espacio y de bandas aplicando Haar-DWT, LSB y DCT.

En Rafat et al. [68] en el 2016, proponen como mecanismo de ocultamiento utilizar el algoritmo LSB y criptografía a través de operaciones XOR y el algoritmo SHA (*Secure Hash Algorithm*) de 256 bits, además agregan *pseudo ramdom number*, más la inclusión de técnicas *Hash* para ocultar información. Swain [140] propone la combinación de los métodos PVD y LSB para incrementar la cantidad de datos incrustados y la robustez del método, utilizando bloques sobrelapados de 2x2 píxeles.

En 2017 Kakade et al. [74], trabajan con la detección de patrones en códigos QR (*Quick Response*), emplean LSB y DWT en la etapa de ocultamiento de datos, mientras que en la etapa de recuperación emplean IDWT. Por otro lado, Al-Farraji [4] propone una revisión sobre LSB y generar una división de las imágenes seleccionadas, incrustan datos mediante la segmentación de datos y generan la codificación binaria sobre la información oculta, mientras tanto Dahiy [32] en el 2017, asevera que los métodos de compresión de datos son recomendables para los procesos de esteganografía.

En el 2017 Darabkh et al. [33], proponen algoritmos basándose en variantes de PVD denominadas Quinnary PVD y Octa-PVD ambas en combinación de MLSB (*Mayor LSB*), con el primer método utilizan una partición de bloques de 3x3 píxeles en 5 parejas para cada uno de ellos, los bloques de partición y las parejas representan los píxeles vecinos, los cuales son seleccionados para realizar el ocultamiento de datos. Además, se emplea el método de ORPSA (*Optimal Reference Point Selection Approach*) con la cual minimizan el MSE y la distorsión de la estego-imagen. Para sus pruebas, utilizan imágenes de 512×512 píxeles, reportando que el valor del PSNR para la estego-imagen es de 38 dB.

En el 2017 Muke et al. [100] proponen un sistema de autenticación para acceso, empleando el ocultamiento de claves en imágenes a escala de gris mediante esteganografía en combinación de teléfonos inteligentes con la capacidad de manejar NFC (*Near Field Communication*).

En el trabajo de Soleymani y Taherinia [136] en el 2018, presentan un método para esteganografía de un documento de texto en la imagen del objeto portador, aprovechan la propiedad dispersa de

los documentos escaneados. Los documentos escaneados pasan de nivel de gris a valores binarios de medio tono, posteriormente las partes incluidas en la información se extrajeron utilizando quadtree, la cual es una técnica que pondera una región de una imagen en donde se da prioridad a un cierto nivel de intensidad, color o textura, en este caso son las áreas donde se encuentran las secciones donde existe texto e imágenes. Separan las áreas de interés para comprimir las partes extraídas, proponen un algoritmo basado en la lectura de los bits de cadena binarios, ignorando el cero detrás del siguiente digito y convirtiéndolos a valores decimales. La capacidad de incrustación es generalmente superior a 7 bits por píxel de la imagen portadora, para la mayoría de las estego-imágenes que presentan superan los 37 dB.

En Zenati et al. 2019 [165] proponen un sistema de esteganografía para incrustar imágenes en escala de grises, en documentos convertidos a imágenes en escala de grises mediante un modelo llamado Beta Elliptic. El sistema tiene dos fases principales, la primera es la incrustación de datos mediante esteganografía, utilizan un detector de puntos clave nominado BRISK (*detector Binary Robust Invariant Scalable Key*) para identificar las posiciones de incrustación en la imagen del documento anfitrión. La firma Beta Elliptic se convierte en una secuencia de bits secretos a través de la posesión previa de la firma. La segunda parte consiste en utilizar un sistema compuesto por transformación binaria y compresión Huffman. La secuencia obtenida se agrega en el primer bit menos significativo de las posiciones de inserción en la imagen del documento anfitrión. La propuesta de los autores permite incrustar directamente la firma elíptica Beta basada en el dominio del espacio usando LSB. Los autores proponen como métricas de rendimiento PSNR, SSIM, HVS y BER, obteniendo resultados sobresalientes en PSNR con puntuaciones más altas que oscilan entre 80 dB y 92 dB en los tres conjuntos de datos que se utilizan, por otro lado, los promedios de los puntajes SSIM son mayores a 0.993 puntos.

En Yakovleva et al. [163] 2018 describen un sistema de esteganografía basado en DWT ortogonal con matrices, el cual genera matrices con diversas variantes con bloques de $N \times N$ donde $N = 6 \times 6$. Los autores proponen el proceso de Gram-Schmidt para construir nuevas bases sobre las matrices generadas y poder embeber imágenes en las matrices que se generaron con la aplicación de DWT.

En Mangla et al. [96] 2019 proponen un sistema de esteganografía para ocultar datos en imágenes de portada con una resolución de 256×256 bits. Utilizan LSB y emplean una binarización del mensaje a ocultar. Emplean el enfoque de segmentación, el cual permite dividir la imagen de acuerdo con el tamaño de la información en bits, por lo que no hay degradación de la imagen y no hay problemas de desvanecimiento o borrosidad del color. Este enfoque también lo hace más seguro y robusto debido a la calidad de imagen obtenido [96].

En Luo et al. [93] 2019, proponen un método donde incrustan una imagen como mensaje en la imagen de portada, en donde la imagen secreta se incrusta independientemente de la diferencia de los valores de píxeles, y se anexa la información de un operador para incrustar con diferentes números de bits de acuerdo con el nivel de la diferencia de valor de píxeles que existe entre la imagen que es el mensaje y la imagen de portada. En este trabajo se analiza la robustez del esquema propuesto bajo el ruido de sal y pimienta. Los ruidos de sal y pimienta que en los experimentos se aplican con diferente densidad, que va de 0 a 0.15 en la estego-imagen "Pimientos" de 256×256 píxeles. Los resultados del PSNR alcanzan un puntaje de 27 dB. Se incrustan diferentes cantidades de datos en diferentes categorías según el grado de suavidad o contraste. Los autores son capaces de evadir estegoanálisis por RS [93].

Bilal y Koyun en 2020 [19] proponen un método de esteganografía el cual consiste en detectar las

zonas de una imagen con mayores diferencias, mediante un mecanismo de generación de cuadriculado de la imagen para obtener bloques elegibles e incrustar información. Los autores proponen una primera fase de incrustación de 500 bytes, y posteriormente una incrustación de 1 bpp. Los resultados muestran un PSNR de 53 dB al incrustar 1 bpp. Cuando los autores proponen incrustar la carga de 500 bytes, obtienen un PSNR de 78 dB y un SSIM de 1 punto, además de un MSE de 0.001 puntos.

La tabla 1 presenta un resumen de los autores y los métodos propuestos de esteganografía mostrados en esta investigación sobre imágenes a escala de gris. Los trabajos han sido seleccionados en función de los resultados obtenido en general. Como se puede observar predominan los métodos modificados como LSB y PVD.

Tabla 1: Trabajos de esteganografía para imágenes en escala de gris, según los datos obtenidos de las métricas de calidad presentadas

Métodos empleados	**Autores**	**Año de publicación**	**Trabajo realizado**	**Niveles de MSE y PSNR obtenidos**
BPIS, LSB	Al-Bahadili [3]	2013	Combinan LSB con la permutación de bloques para dificultar la ubicación de la secuencia generada por LSB en imágenes BMP	MSE=ND PSNR=58.71 dB
DWT	Nehete y Bhide [103]	2014	Proponen un conjunto de imágenes de rostros de personas con distintos tonos de piel para ocultar datos a través de DWT	MSE=ND PSNR=68.73 dB
M-LSB y PBSA	Muhammad et al [99]	2015	Combinación de LSB con PBSA para generar una base de ocultamiento de patrones en la imagen de portada	MSE=ND PSNR=44.58 dB
Watershed	Puri y Deep [115]	2015	Proponen la eliminación de ruido en la imagen de portada y se buscan regiones de píxeles para incrustar datos. Utilizan RSA como segundo mecanismo de protección	MSE=0.01 PSNR=68.74 dB
TPVD	Hernández et al. [64].	2015	Aplican TPVD mediante bloques de píxeles de 2x2, obteniendo mejores resultados que Wavelet Haar empleada por F. Peng	MSE=ND PSNR=36.04 dB
LSB	Gulve et al. [58]	2015	Utilizan una variante de LSB con bloques de 2x3 píxeles y obtienen resultados aceptables en cuanto a la calidad de la estego–imagen	MSE=5.15 PSNR=41 dB
PVD con variante Quinnary y Octa, en combinación de MLSB	Darabkh et al. [33]	2017	Modificación de PVD con bloques de píxeles de 3x3 en agrupaciones de 5 u 8 bloques, y emplean ORPSA para minimizar el MSE	MSE=ND PSNR=40.03 dB
Quadtree, dimisnución de resolución	Soleymani y Taherinia [136].	2018	Los documentos escaneados pasan de nivel de gris a valores binarios por medio de medio tono, posteriormente las partes incluidas en la información se extrajeron utilizando quadtree, Separan las áreas de interés para comprimir las partes extraídas, proponen un algoritmo basado en la lectura de los bits de cadena binarios, ignorando el cero detrás del siguiente dígito y convirtiéndolos a valores decimales. La capacidad de incrustación es generalmente superior a 7 bits por píxel de la imagen portadora	MSE=ND PSNR=37 dB
Beta Elliptic, LSB, Transformación binaria	Zenati et al. [165]	2019	El sistema tiene dos fases principales, la primera es la incrustación de datos mediante esteganografía, utilizan un detector de puntos clave nominado BRISK (*detector Binary Robust Invariant Scalable Key*) para identificar las posiciones de incrustación en la imagen del documento anfitrión. La firma Beta Elliptic se convierte en una secuencia de bits secretos a través de la posesión previa de la firma.	MSE=ND PSNR=82 dB
LSB y segmentación de imagen	Bilal, H. y Koyun, A. [19].	2020	Segmentan la imagen en patrones que separan los elementos de la imagen que contienen zonas con amplias diferencias de valores con respecto a zonas homogéneas	MSE=0.3 PSNR=53 dB

ND (No disponible)

2.6.3 Esteganografía para imágenes a color

Chuang et al. [27] en el 2006 propuso un método de esteganografía para imágenes comprimidas con BTC (*Block Truncation Coding*). En el esquema propuesto, se usó una estrategia de programación dinámica para encontrar la solución óptima de la función de mapeo biyectivo para el reemplazo de LSB con tres bits para obtener una baja distorsión en la estego-imagen. Sun et al. [92] en el 2013

presentaron una modificación en AMBTC (*Absolute Moment Block Truncation Coding*), en este estudio propusieron un método de esteganografía en el que utilizan análisis de matriz con código de Hamming para incrustar un mensaje secreto en el flujo de bits AMBTC comprimido que muestra resultados aceptables en la incrustación capacidad, velocidad de bits y eficiencia de ocultamiento.

En el área de imágenes a color se tiene el trabajo de Nori y Al-Qassab [106], en el 2012 desarrollan el fractal Julia para incrustar datos en imágenes RGB, a través de la modificación de bits en cada canal de la imagen. Los resultados mostrados por los autores son de una estego-imagen con un PSNR superior a 79 dB y un MSE y BER despreciables. En Eswari et al. [46] en el 2014 aplican el algoritmo de Zhang para incrustar información dentro de imágenes fractales además de combinar el método con RSA. Los resultados obtenidos del PSNR son superiores a 42 dB.

Prabakaran et al. [113] en 2014 proponen DWT para la incrustación de datos y la IDWT (*Inverse DWT*) para obtención de los datos ocultos en la imagen, además de la incorporación de SVD (*Singular Value Descomposition*). Las pruebas se hicieron en imágenes con formatos JPEG y TIFF en Matlab, con matrices de 512×512 píxeles con el modelo RGB en bloques de 4 píxeles aprovechando operaciones de XOR y obteniendo un total del 25 % de capacidad de incrustación de datos con referente al tamaño de la imagen.

En el 2014 Meenakshi et al. [97], proponen una modificación de LSB para espacios de color YIQ en imágenes con formato JPEG aplicando rotaciones sobre RGB, realizaron pruebas extensivas sobre modelos CMYK, XYZ, YCbCr (*Luma Chrominance Blue and Red*) y YIQ, obteniendo un PSNR ligeramente superior a 30 dB para el modelo CMYK, para el resto de los modelos el PSNR es inferior a 20 dB.

Qazanfari y Safabakhsh en el 2014 [117] presentan el método LSB++ para mejorar el método LSB+ propuesto por Wu et al. [39], lograron conservar el histograma de la imagen en el dominio espacial incorporando bits adicionales en las imágenes. Sin embargo, produce distorsiones estadísticas y perceptivas al prohibir que algunos píxeles cambien. En su trabajo mejoraron el método LSB+ al distinguir los píxeles sensibles a modificaciones y protegerlos de la incrustación de bits adicionales, lo que provoca una menor distorsión en las matrices de coocurrencia. El método LSB++ ayuda a preservar el histograma de coeficientes DCT de imágenes JPEG y generalizar el método para el caso en donde se utilizan más de un bit de los píxeles de la imagen portadora. Los resultados experimentales muestran que el método LSB++ mejorado, produce una menor cantidad de distorsiones en las matrices de concurrencia que el método LSB++. Con su propuesta, se probó que los ataques basados en histogramas no pueden detectar correctamente las estego-imágenes producidas con o sin incrustación de bits adicionales.

La teoría de fractales se puede llevar a cabo para el desarrollo de métodos de esteganografía como se aplica en Desai et al. [36] en el 2014, los autores utilizan los principios de la criptografía y el watermarking conjuntamente con la teoría fractal de Mandelbrot para la compresión de la imagen a incrustar. En este trabajo la imagen a ocultar se divide en diferentes secciones de acuerdo con la ecuación fractal propuesta y posteriormente se incrusta mediante DTC. De acuerdo con los resultados que obtuvieron, las pruebas fueron realizadas sobre imágenes en escala de gris y en RGB, validando sus resultados a través del análisis del histograma en donde no existen diferencias entre las imágenes tratadas.

Shobana [130] en el 2015, explotan las propiedades de las sombras de las imágenes bajo el modelo CMYK, basándose en el método LSB de 4 bits para las modificaciones en píxeles. Los niveles de PSNR que reportan los autores son superiores a 44 dB.

Stoyanova y Tasheva [138] en el 2015 aplican una modificación sobre LSB, emplean una llave criptográfica la cual permite el control de la incrustación de datos y la recuperación de estos mediante el sistema Rijndael, el cual propone un algoritmo simétrico. El PSNR de las estego-imágenes es superior a 54 dB, tomando en cuenta que usan los 3 primeros bits menos significativos. Las métricas de comprobación de calidad de la imagen que aplican son MSE, SNR, PSNR y SSIM.

Otros trabajos como el de Kaur et al. [77] en el 2015 utilizaron imágenes de color con baja resolución, a las cuales se les aplica el método de la curva elíptica y el mecanismo de cifrado. Obtienen niveles de PSNR superiores a 70 dB para la estego-imagen, pero no se muestra el tamaño de datos incrustados, ni el tamaño de las imágenes que se utilizaron como base. Su propuesta es una variante de LSB para imágenes RGB y los datos están codificados en ASCII (*American Standard Code for Information Interchange*). Al ocultar una imagen de 128x128 píxeles en otra de 512×512 píxeles se obtiene un PSNR de 55.9 dB.

En Naoum et al. [102] 2015 emplean los métodos de DWT y ERBP (*Embeddim Phase the Combination*). La selección de la estego-imagen se basa en un proceso de redes neuronales artificiales, la primera red es del tipo SOM (*Self-Organizing Maps*) las cuales fueron propuestas por Kohonen en 1982 ([84], [56]). El modelo se conforma por una capa de N_1 neuronas llamada capa de entrada que recibe a t y transfiere a la capa de salida la información procedente del exterior, mientras que la capa de salida conformada por M_1 neuronas procesa la información formando el mapa de rasgos llamado Palmer [108]. La primera red neuronal es no supervisada y la segunda utiliza la retro propagación resiliente. Recurren a métodos de encriptación por llaves, se ocultan imágenes de 64×64 píxeles y de 128x128 píxeles en imágenes de 256×256 y 512×512 píxeles respectivamente. Los resultados del PSNR están por arriba de los 105 dB y un MSE inferior a $2.57e^{-5}$.

La adopción de LSB en el modelo RGB es altamente probada como lo demuestran en el 2016 con el trabajo de Ouyang et al. [107], en combinación con operaciones XOR obtiene resultados sobresalientes en imágenes de 512×512 píxeles al incrustar imágenes del tamaño de un 25 % en relación al de la imagen de portada, sus resultados arrojan niveles superiores de 55 dB de PSNR.

El desarrollo de Geethaa y Thamizhchelvy [142], incorpora varios elementos como el PRNG (*Pseudo Random Number Generator*) en su versión hardware debido a que genera una secuencia de números difíciles de predecir, por lo tanto, es altamente utilizable en la generación de llaves criptográficas. A través de la generación fractales se combina la teoría del caos y la aplicación de la secuencia Fibonacci, para posteriormente aprovecharlo como marca de agua en un archivo de imagen.

Una de las aplicaciones de la transformada de Radon se puede encontrar en el trabajo de Roy y Changder [123] en el 2016, generan un haz de luz paralelo para conseguir la reducción de datos incrustados en la imagen, en esta investigación también aplican un algoritmo pseudo aleatorio que permita cifrar los datos incrustados, y proponen una matriz de codificación de datos. El proceso de ocultamiento es a través del LSB y llaves criptográficas con *Hash*. Obtienen niveles de PSNR superiores a 56 dB.

Vaishali y Kajal [147] en 2016 propusieron un método de esteganografía que se apoya en la compresión LZW y en la segmentación de plano de bits (BPCS). El método propuesto integra LZW y BPCS, donde las imágenes se comprimen antes de ser transmitidas por la red. Su propuesta mejora la capacidad de ocultación de datos de la imagen en comparación con los métodos de esteganografía de imagen existentes, al tiempo que se conserva la calidad de la imagen después de incrustar el mensaje secreto en ella. Presentan cargas de inyección superiores a 370,000 bytes, logrando que los

niveles de PSNR estuvieran por encima de los 50 dB.

Hardikkumar y Apurva en el 2016 [62]}, realizan una investigación sobre ocultamiento de información basándose en la generación de un fractal de Mandelbrot para localizar los datos ocultos dentro de la imagen, en éste se muestra la incrustación de una imagen la cual contiene texto. El proceso de modificación se realiza sobre imágenes RGB y se manipulan los bits a alterar, obteniendo resultados satisfactorios en relación con otros métodos, pero tiene el problema de que la imagen a incrustar es menor a la resultante cuando se genera el fractal.

En el 2016 Umbarkar et al. [145], proponen un método basado en la incrustación de datos de forma aleatoria, dependiendo del tamaño del mensaje para cifrar se seleccionan regiones de la imagen idóneas para realizar el proceso utilizando LSB. Los resultados del PSNR en las imágenes de 512×512 píxeles con 26,214 bits incrustados son mayores a 61 dB, cuando se triplica la cantidad de datos para incrustar se puede obtener un PSNR mayor a 56 dB, los formatos que utilizaron son BMP, PNG, TIFF y JPEG. Los resultados obtenidos contra otros autores que utilizaron métodos como LSBM, LSMMR, EA-LSB (variantes de LSB), PVD, IPVD (variantes de PVD), y HBC (*Hidden behind corner*) son superiores.

En Desai et al. [37] en el 2016, realizan una investigación sobre ocultamiento de información basándose en la generación de un fractal de Mandelbrot para localizar los datos ocultos dentro de la imagen, en este trabajo se muestra la incrustación de una imagen la cual contiene texto. El proceso de modificación se realiza sobre imágenes RGB y se manipulan los bits a alterar, obteniendo resultado superiores entorno a un 10 % en relación con otros métodos, presentando el inconveniente de que la imagen a incrustar debe ser menor a la resultante cuando se genera el fractal.

Geetha et al. [55] en 2016 reporta una extensa investigación de aplicación de teoría del caos y teoría fractal. El análisis que presentan permite obtener una idea sobre los alcances que existen en el área de teoría fractal, sobre todo al emplear los métodos de compresión a través de fractales, así como en estos, y sobre todo las técnicas de reconocimiento de patrones de elementos cíclicos para esteganografía, aunque varias de estas propuestas permiten observar que se combinan con LSB, DCT, DWT, entre otras.

En Kim [83] en el 2016 se propone un sistema de autenticación que utilice un algoritmo simétrico de cifrado y un código de detección de modificaciones. El algoritmo de cifrado es AES [122], tanto el código de cifrado como el de autenticación se envían de forma independiente. El sistema propuesto utiliza un teléfono inteligente con un lector y grabador de tarjetas NFC y una estego-imagen para acceder a los datos grabados en la tarjeta.

En el trabajo de Sankpal et al. [126] 2017 proponen una metodología en la cual se genera un control de permisos de acceso, y posteriormente, la información se oculta incrustando el código de acceso en una fotografía que está almacenada en un teléfono celular. El hardware que emplea es un smartphone con sistema Android y un Arduino para el control de lectura de tarjetas NFC.

Estupiñan y Acosta en 2018 [45] presentan un método de esteganografía, en el cual proponen embeber imágenes RGB mediante la generación de imágenes mosaico, buscando las zonas ruidosas de las de la imagen portadora y de la representación de la imagen a incrustar mediante el cálculo de los coeficientes de desviaciones estándar de cada bloque dentro del rango establecido y los residuos de desbordamiento y subdesbordamiento, empleando filtros y operaciones diferenciales para reducir los efectos sobre la imagen mosaico. Los autores probaron su método con imágenes de 640x480 píxeles, debido a que les permite aplicar distintos tamaños de bloques para incrustar el mensaje

oculto. En los resultados experimentales el promedio de PSNR obtenido fue de 18.8 dB y al aplicar StegExpose obtuvieron un 84 % de evasión al analizar 100 imágenes.

En el trabajo de Shashikiran et al. en el 2019 [129] proponen un método utilizando DWT para descomponer las imágenes de portada en sus canales R, G y B y luego obtener subbandas LL, LH, HL y HH de la imagen de portada, las imágenes que se ocultarán se descomponen en sus canales R, G y el canal B para luego ser incrustados por LSB en el MSB (*Major Significant Bit*). En este documento, incrustan una imagen para cada canal de la imagen de portada y obtienen puntajes promedio de PSNR mayores de 35 dB para las estego-imágenes, mientras que las imágenes recuperadas de su promedio en PSNR tienen un promedio de 29 dB.

Una combinación de mecanismos de seguridad como RSA [24] permite que el mensaje tenga una capa adicional de seguridad como en Ambika et al. en el 2019 [8] donde aplican DWT para incrustar mensajes cifrados en imágenes en color con dimensiones de 512×512 píxeles, el mensaje oculto es imágenes en escala de grises. Obtienen 31.792 dB y 0.86612 puntos MSE para las estego-imágenes analizadas.

Analizando los trabajos presentados, podemos ver que la mayoría de estos trabajos presentan desarrollo en métodos espaciales como LSB, PVD, por otro lado, una de las contribuciones novedosas es el uso de fractales para la incrustación de datos. En la siguiente sección presentamos un método basado en el dominio del espacio y el dominio de frecuencia para incrustar datos en una imagen digital. Por otro lado, se ha analizado para aprovechar los métodos espaciales y en el dominio de frecuencia para ofrecer un método con la capacidad de incorporar altas velocidades de datos además de no perderse al aplicar la recuperación de información.

Recientemente, DL (*Deep Learning*) se ha utilizado para incrustar información en imágenes digitales, tal es el trabajo presentado por 2017 Baluja [12] en el 2017, el autor propone un método de esteganografía basado en una arquitectura de aprendizaje profundo que incorpora secuencias de ruido en imágenes JPEG para evadir ataques estadísticos. Este sistema se prueba incorporando imágenes como un mensaje secreto en imágenes JPEG de 64×64 píxeles, el dataset utilizado es Tiny (dataset empleado en el desafío de ImageNet), en los resultados que Baluja reporta valida que el método es ampliamente efectivo para evitar métodos de análisis estadístico, pero las estego-imágenes muestran variaciones en los tonos debido a la pérdida de calidad que ha sufrido. Otro trabajo de esquema esteganográfico basado en DL es el de Zhang et al. [166] 2019. Los resultados mostraron una carga útil efectiva de hasta 4.4 bits por píxel. La evaluación experimental se llevó a cabo con los dataset: COCO (*Common Objects in Context*) [90] y Div2k (*Diverse 2 K resolution*) [119].

Biswas y Bandyapadhay en 2020 [21] desarrollaron un método de esteganografía para ocultar información en imágenes de color empleando el dominio de la frecuencia con la implementación de un algoritmo genético (AG) de alta resistencia a ataques contra estego-imágenes. Su propuesta de esteganografía consiste en una segmentación de los componentes de la imagen en 4 bits, posteriormente se seleccionan múltiples bits de forma aleatoria. El ocultamiento de la información se realiza con una función Hash, además los datos se encriptan. Los autores proponen analizar sus estego-imágenes con StirMar 4.0, y herramientas de estegoanálisis estadístico. En sus estego-imágenes argumentan que visualmente no sufren distorsiones o perturbaciones apreciables.

Varalakshmi en el 2020 propone un método de esteganografía para ocultamiento de información en imágenes digitales RGB, en donde emplean la transformada de Karhunen-Loeve. El autor propone emplear imágenes de portada con una resolución de 474x277 píxeles con 96 DPI, en las cuales

se incrusta texto de forma aleatoria y obtienen valores de PNSR promedio de 82.308 dB y un MSE de 6.4 E^{-4} [148].

En la década del 2010 se presentan modificaciones en algoritmos tales como LSB, PVD, planos de bits, entre otros, incrementando el interés en ser aplicados para imágenes con modelos a color. En esteganografía los métodos de compresión de información se han empleado para incrementar los datos que pueden ser incrustados en los estego-objetos, además de que permiten incrementar la calidad de la imagen según los resultados de las métricas PSNR y MSE.

La dispersión de datos en matrices ha sido ampliamente estudiada por diversos autores, sobre todo en el aprovechamiento de espacio para reducción de cálculos aritméticos y de posicionamiento como es el caso de Yan et al. [164], Kumar et al. [86], Bell et al. [16], el control de la dispersión de información permite mejor dominio del flujo de datos, así como su reordenamiento, resultando útil en una gran diversidad de tareas.

Un punto para considerar es que la mayor parte de los métodos presentados para esteganografía son una serie de modificaciones sobre LSB, PVD o de análisis por capas de bits, por otro lado, el uso de métodos que están en el dominio de la frecuencia como DWT, DTC, Wavalets, son empleadas en combinación con los métodos de dominio espacial para aprovechar la cantidad de datos incrustados. La mayoría de los autores presenta resultados parciales mostrando poca claridad entre la relación de datos incrustados y la capacidad máxima de recepción del objeto portador, pero sí los resultados de métricas donde el nivel de calidad de las imágenes es aceptable tomando como métrica el PSNR.

La tabla 2 muestra un resumen de los autores y los métodos propuestos de esteganografía mostrados en esta investigación para imágenes a color, en relación con los datos que obtuvieron de las métricas de calidad, como se puede observar existen una fusión de métodos basados en el dominio del espacio y de la frecuencia, estos trabajos han sido elegido debido a que son los que presentan mejores resultados con respecto a los demás trabajos reportados. Se observa un claro predominio del algoritmo LSB en la mayoría de los trabajos.

Tabla 2: Trabajos de esteganografía para imágenes a color en base a los resultados obtenidos de las métricas de calidad

Métodos empleados	**Autores**	**Año de publicación**	**Trabajo realizado**	**Niveles de MSE y PSNR obtenidos**
Fractal Julia, LSB	Nori y Al-Qassab [106]	2012	Aprovechan las características de las imágenes resultantes de los fractales utilizando el conjunto de Julia e incrustar datos con LSB	MSE=0.27 PSNR=79.96 dB
DWT	Prabakaran et al. [113]	2014	Ocultan datos en imágenes con formato JPEG a través de DWT combinado con SVD, la imagen se segmenta en bloques de 4x4 de píxeles	MSE=0.25 PSNR=64.24 dB
LSB	Singh y Meenakshi [97]	2014	Aplican rotaciones en bloques de imágenes con modelo de color YIQ, los datos son ocultados con LSB	MSE= 58.86 PSNR= 30.43 dB
Fractal de Mandelbrot, DCT	Desai et al. [37]	2014	Generan imágenes con fractales obtenidos del conjunto de Mandelbrot, además se aprovecha la compresión fractal. Los datos son ocultos mediante DCT	MSE=ND PSNR=ND

Tabla 2. continuación: Trabajos de esteganografía para imágenes a color en base a los resultados obtenidos de las métricas de calidad

Métodos empleados	**Autores**	**Año de publicación**	**Trabajo realizado**	**Niveles de MSE y PSNR obtenidos**
PRNG, Fractal	Geethaa y Thamizhchelvy [55]	2014	Ocultan datos con una versión de números aleatorios para ocultar la secuencia de datos. En el fractal propuesto aprovechan la serie de Fibonacci para aplicar una marca de agua	MSE=ND PSNR=ND
LSB	Stoyanova y Tasheva [138]	2015	Proponen una combinación de LSB con sistemas criptográficos y para la recuperación de datos se usa Rijndael. Explotan los tres bits menos significativos de los píxeles	MSE=2.68 $e^{-}5$ PSNR=86.21 dB
DWT y ERBP	Naoum et al. [102]	2015	Combinan DWT y ERBP, proponen la elección de las imágenes de portada a través de redes neuronales	MSE= 2.57 e^{-5} PSNR= 105.68 dB
Transformada de Radon, LSB	Roy y Changder [123]	2016	En este trabajo realizan una combinación interesante de la transformada de Radon, para lograr compresión de datos, y a su vez aprovechando las ventajas de LSB para incrustar gran cantidad de datos, siendo resistente a ataques estáticos	MSE=0.01 PSNR=57.23 dB
LSB y operaciones XOR	Ouyang et al. [107]	2016	Combinan LSB con operaciones XOR, logran ocultar hasta un %25 de la capacidad total de la imagen	MSE=ND PSNR=55.88 dB
Deep Leraning y patrones de ruido	Baluja [12]	2017	Incrustan imágenes JPEG de forma aleatoria en imágenes de dimensiones iguales, con grandes resultados en evasión por estegoanálisis, con la desventaja de imágenes distorsionadas. Emplean imágenes del dataset Tiny	MSE=150 PSNR=27 dB
Mosaicos y zonas ruidosas	Estupiñan y Acosta [93].	2018	Incrustación de datos por matrices en zonas ruidosas en la imagen de portada mediante la sustitución de valores en píxeles empleando información representativa	RMSE=29.342 PSNR=18.8 dB
DWT	Shashikiran et al. [129].	2019	Descomponen las imágenes de portada en sus canales R, G y B y luego obtienen subbandas LL, LH, HL y HH de la imagen de portada, las imágenes que se ocultarán se descomponen en sus R, G y los canales B para luego ser incrustados por LSB en el MSB.Incrustan una imagen para cada canal de la imagen de portada	MSE=ND PSNR=35 dB
Transformada de Karhunen-Loeve	Varalakshmi 2020 [148].	2020	Las transformada de Karhunen-Loeve permite cifrar y descifrar el objeto a ocultar en las imágenes de portada. El autor propone emplear imágenes de portada con una resolución de 474x277 píxeles con 96 DPI (*Dots per inch*)	MSE=0.0006 PSNR=80 dB

ND= (No disponible)

Como bien se ha observado en los trabajos descritos sobre esteganografía, la tendencia es clara en el uso de métricas como PSNR y MSE, varios de estos trabajos con elevados puntajes obtenidos en estas métricas, lo cual no es un indicio de que todas las estego-imágenes obtenidas de estos métodos indiquen que a nivel visual no presentarán distorsiones, como lo indica Z. Wang [152], y por tal motivo es necesario aplicar un mayor número de pruebas y métricas que validen los resultados de un método de esteganografía, además un uso extensivo de dataset de imágenes para su respectiva validación.

Con respecto a los métodos analizados, se puede verificar que aquellos que emplean dominio espacial, como es el caso de Zenati et al. [165], Soleymani [136] y Taherinia y Gulve [58], permiten cargas que van de 1 bpp a más de 4 bpp, debido a que las modificaciones realizas sobre los píxeles permiten mayor cantidad de incrustación de datos, por otra parte los métodos enfocados en las modificaciones en dominio de la frecuencia como es el caso de Nehete [85], Naoum et al [102]. y Varalakshmi [148],presentan una menor tasa de incrustación de datos, debido a que las modificaciones realizadas requieren de una mayor cantidad de píxeles que al usar el dominio espacial. Por otra parte, se pueden presentar combinaciones de los enfoques espaciales y en dominio de la frecuencia para obtener cargas mayores de los que logran los segundos, pero con la ventaja de obtener resistencia contra ataques estadísticos como es el caso del trabajo de Qazanfari y Safabakhsh [117]. De

los métodos con mayor efectividad en la evasión que se han encontrado es el de Baluja [12], debido a que emplea DL para evadir estegoanálisis.

Con base en la información analizada en la revisión literaria se puede decir que los métodos de esteganografía actuales deben de buscar cargas de datos superiores a los 3 bpp y como meta evadir estegoanálisis, una de las formas más viables está en lograr una combinación de los enfoques espaciales y en dominio de la frecuencia, donde la información esté distribuida con un orden que sea inaccesible para un atacante.

Por otra parte, si bien los imágenes con modelos de color permiten aumentar la relación de datos incrustados por estego-imagen generada, se necesita equilibrar las cantidad de datos que se incrustan en determinadas zonas de las estego-imágenes, debido a que son mayormente susceptibles a cambios si se les compara con respecto a las imágenes en escala de gris, como se puede observar en los resultados de Baluja y en los resultados del trabajo de Zenati et al. (imágenes en escala e gris). Tomando en cuenta el análisis efectuado en este capítulo sobre esteganografía e investigaciones sobre incrustación de información en imágenes digitales se da paso a la propuesta de un método de esteganografía, el cual es propuesto en el siguiente capítulo.

Capítulo 3

Método de Esteganografía Propuesto: *VVRSM*

En el presente capítulo se describe el diseño del método propuesto de esteganografía denominado como VVRSM (*Virtual Variable Redistribution Steganographic Method*). Para el método propuesto se han considerado como premisas centrales: (a) alcanzar altas cargas de incrustación de datos en imágenes tipo RGB aprovechando los canales que estas imágenes brindan y (b) mantener la seguridad de los datos mediante una propuesta de codificación, la cual considera la dimensión de fractales deterministas en combinación con la dispersión temporal de píxeles, con ello, se forman imágenes virtuales encargadas de recibir el mensaje oculto. En los siguientes párrafos se detalla el método propuesto.

El método de esteganografía propuesto se centra en dos conceptos:

- Representación y reducción del mensaje. Resulta necesario analizar la representación y reducción del mensaje para transportar cualquier objeto digital en imágenes RGB sin pérdida de información. Adicionalmente, un sistema de codificación basado en la redefinición de los valores de los símbolos del alfabeto procedente del proceso de compresión es utilizado, mediante vectores formados por el cálculo de la dimensión de un fractal elegido en un proceso que extraiga el que presente mayores variaciones en las dimensiones fractales calculadas.

- Tratamiento de la imagen de portada. Se considera una transformación virtual de la imagen de portada original para generar un efecto de aparente eliminación del mecanismo de ocultamiento de información, al momento de que se obtiene la estego-imagen final. Se incluye el uso de una máscara de seguridad a través de la transforma de DWT con el objeto de reducir el efecto de aleatoriedad del proceso de incrustación, con ello, disminuyendo la probabilidad de detección del mensaje por parte de los métodos estadísticos de estegoanálisis.

El método VVRSM se divide en dos etapas:

- **Etapa 1:** *Transformación del mensaje*
- **Etapa 2:** *Manipulación de la imagen*

En las siguientes secciones se describirán las dos etapas que conforman el método de esteganografía propuesto.

3.1 Etapa 1: Transformación del mensaje

Está primer etapa correspondiente a la transformación del mensaje se conforma por cuatro fases ilustradas en la figura 10.

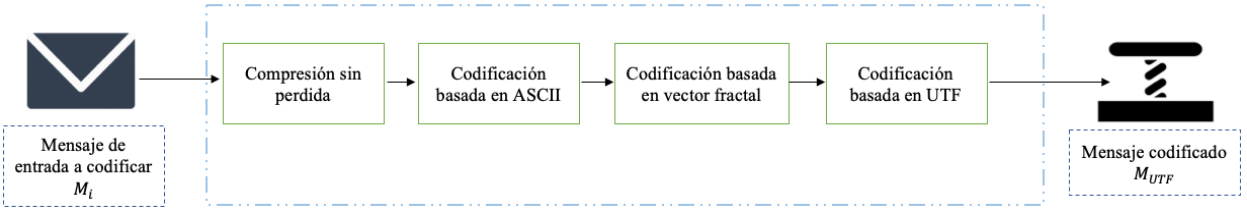


Figura 10: Método de esteganografía $VVRSM$ para la transformación del mensaje

Fase 1. Compresión sin pérdida de información. El mensaje de entrada M_i, es cualquier objeto digital que puede incrustarse en una imagen de portada. M_i se comprime a través del algoritmo de compresión LZW quien permite expresar cualquier objeto digital en una cadena de texto, sin la necesidad de realizar un proceso adicional y manteniendo la integridad del mensaje a ocultar.

LZW crea una tabla de clasificación de los símbolos de entrada asignando una posición y un valor. Posteriormente se forman pares de símbolos del mensaje que se está comprimiendo de tal manera que, si el último símbolo termina con valor w el próximo valor del par de símbolos es $w + 2$ (desplazamiento de posiciones), remplazando los pares de caracteres por un código numérico para comprimir el mensaje. El mensaje se transforma en cadena de texto, y codificado mediante la ecuación (20).

$$M_{LZW} = LZW(M_i) \tag{20}$$

Donde M_{LZW} representa una cadena sin pérdidas de datos transformada a través de la función de compresión $LZW()$.

Fase 2. Codificación basada en ASCII. La cadena M_{LZW} es codificada mediante la ecuación (21) a través de un codificador basado en ASCII (función $B_{64}()$) utilizando 64 caracteres, estos caracteres van de $a - z$, de $A - Z$, considerando los dígitos $0 - 9$ y los símbolos / y +. La salida de esta codificación genera una cadena M_{b64} obteniendo un alfabeto altamente reducido (64 símbolos). La elección de dicha codificación se debe a que permite representar cualquier mensaje digital con símbolos de un tamaño de 8 bits, esto es posible gracias al proceso previo de compresión por LZW, encargado de la reducción del tamaño de M_i.

$$M_{b64} = B_{64}(M_{LZW}) \tag{21}$$

M_{b64} es obtenida por la ecuación (22) formada a partir de los símbolos binarizados de M_{LZW} y representados en M_{cb}. Éstos son concatenados en secuencias de 8 en 8 bits, siendo k una variable de recorrido que representa el total de grupos de 8 bits de M_{cb} y $nc = k - 1$ es una variable de control de desplazamiento.

$$M_{b64} = \sum_{nc=0}^{nc \leq k-1} \left(M_{64} + C_o \left(\sum_{cb=1}^{cb \leq 8} M_{cb+(nc \times 8)} \right) \right) \tag{22}$$

La función $C_O()$ retorna el equivalente en base 64 de un símbolo, producto del recorrido de 8 bits que se efectúa en $\sum_{cb=1}^{cb \leq 8} M_{cb+(k \times 8)}$ para ser concatenado en M_{b64} como ASCI. Todo lo anterior se

puede observar en el planteamiento efectuado en el pseudocódigo 1.

Pseudocódigo 1 Obtención de la cadena M_{b64}

```
Inicio
b64="A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,
r,s,t,u,v,w,x,y,z,0,1,2,3,4,5,6,7,8,9,+,/"
temp = ' ', x = 0, M_LZW, M_b64 // M_LZW cadena comprimida, M_b64 cadena en base 64
M_cb= Binarizar(M_LZW) // Binarización de M_LZW
for x ≤ size_cadena(M_cb)-1, x ++ do
    temp+ = M_cb[x] // Concatenar subcadenas a variable temp
    if x%8==1 then
        M_b64 + = convertir_b64(temp) // se concatena el equivalente de temp al símbolo del
    arreglo b64
        temp = ' ' // limpia temp
    end if
end for
Almacenar M_b64
```

El pseudocódigo 1 presenta la codificación del mensaje comprimido en LZW, este pseudocódigo comienza con la inicialización de un alfabeto de 64 símbolos, posterior a esto se realiza la lectura de M_{LZW} para convertir todo el mensaje en binario y empezar a realizar la sustitución de las subcadenas del mensaje M_{cb} a base 64, de tal forma que se busca la relación entre las subcadenas de M_{cb} para generar la sustitución con respecto al alfabeto contenido en el arreglo $b64$, y concatenar todas las nuevas subcadenas obtenidas en M_{b64}. La representación de símbolos mediante el uso de 7 y 6 bits se puede observar en el anexo A.

Fase 3. Codificación basada en vector fractal. Los sistemas de codificación basados en ASCII y en otros estándares de representación de caracteres son ampliamente conocidos y utilizados en la literatura, por esta razón se propone obtener un vector denominado como v_f para que calculando la dimensión geométrica de diferentes fractales con base en la ecuación general (3) [110] (expuesta en el capítulo 2, sección 2.2.3) sea posible obtener combinaciones distintas sobre el valor posicional del alfabeto del mensaje a incrustar en las estego-imágenes, esto es con respecto al valor de los símbolos de este alfabeto para que cambien sus posiciones originales, sin perder la referencia para poder recuperar el mensaje mediante una secuencia de reglas.

Los fractales muestran variaciones en su dimensión geométrica (también conocida como D_{MB}) en cuanto mayor precisión se emplee antes del punto decimal es posible apreciar mayores variaciones de esta, por lo tanto, se ha diseñado un mecanismo para generar una lista que contenga ecuaciones de fractales deterministas denominada como U para calcular su dimensión, se han incluido las ecuaciones más importantes de fractales deterministas en el anexo B, y están incluidas en un ciclo que almacena el resultado de las operaciones. Se puede decir que v_f es capaz de almacenar n variaciones de D_{MB}, correspondiente a un fractal en específico.

En el pseudocódigo 2 se especifica la construcción de v_f, de tal forma que en un ciclo se calcula una cantidad de n variantes de dimensiones fractales D_{MB} tomando una lista U de fractales deterministas con su ecuación para calcular D_{MB} con una alta precisión de dígitos antes del punto flotante (50 dígitos antes punto decimal, para la detección de variaciones entre las dimensiones fractales). Como en U existen distintas ecuaciones disponibles, en un arreglo denominado $lista[]$ se almacenas todas las variaciones encontradas DMB obtenidas, especificando cuando termina el cálculo de una

proceso de obtención de dimensión fractal por ecuación con una etiqueta "Fin Fractal, $U[inicio]$". Finalizando el primer ciclo, se procede a agrupar los resultados de los resultados con las ecuaciones de mejor desempeño en variaciones de dimensiones fraccionarias para formar v_f. Finalmente, el último siguiente ciclo permite seleccionar una variante de D_{MB}, perteneciente a una sola ecuación de cálculo de la dimensión fraccionaria. Al seleccionar de v_f una variante de D_{MB}, se almacenan el número de la posición en donde se encuentra la variante de dimensión fractal y se forma un nuevo vector, denominado como v'_f. El número de elementos de v'_f debe ser el número de símbolos que contiene M_{b64}.

Pseudocódigo 2 Generación de v'_f como vector de dimensiones fractales

1: **Inicio**
2: v_f[], v'_f, U[]= D_{MB} // $lista$[] // v_f almacena las dimensiones fractales calculadas y U son las ecuaciones disponibles para calcular las dimensiones fractales, v'_f vector final
3: $i = 0$, n= 0, $inicio = 0$, M_{b64} // n representa el total ecuaciones de fractales a utilizar
4: $fractaL$ // es el total de ecuaciones empleadas en U
5: $grupos$[][], $cantidad$[][]
6: **for** $inicio \leq numero\_elementos(fractalL)$, inicio++ **do**
7: **for** $i \leq n$, i++ **do**
8: $lista$[i]= $Ejecucion\_D_{MB}(U[inicio])$ // Se almacenan las variantes de cada ecuación de la dimensión fractal
9: **end for**
10: $lista$[i]= "Fin fractal" + $U[inicio]$ // Etiqueta que marca el fin del cálculo de un conjunto de dimensiones procedentes de una ecuación de dimensión fractal
11: **end for**
12: $grupos = agrupar\_dimensiones(lista[])$("Fin fractal- inicio) // Obtener el grupo de dimensiones fractales
13: **for** $i \leq numero\_elementos(grupos)$ **do**
14: $cantidad[i][grupos[i]] = elementoRepetido(grupos[i])$
15: **end for**
16: v_f=$Obtener\_NumeroMayor(cantidad[]grupos[])$ // Obtiene la ecuación que obtuvo mayor variación en número de dimensiones fraccionarias
17: **for** $i \leq numero\_simbolos(M_{b64}$-1),i++ **do**
18: v'_f[i] = $posicion\_dimension(v_f[i])$ // Almacena una dimensión en específico y guarda el número de posición en donde se obtuvo
19: **end for**

En el pseudocódigo 3 se muestra el proceso para crear una nueva distribución del alfabeto de la cadena M_{b64}. En el primer ciclo de anidación se almacenan en $list2$[] los símbolos con un nuevo orden, tomando en cuenta que v'_f es el punto de referencia para reordena el alfabeto de M_{b64}, lo cual consiste en que en $list2$ se almacenan los símbolos, de acuerdo con las posiciones que estén almacenadas v'_f, de tal forma que si una casilla ya está ocupada, se sigue corriendo el ciclo para que el desplazamiento permita encontrar una localidad de memoria disponible en $list2$ y evitar la sobre escritura de datos, estos desplazamiento son posibles mediante las reglas que están sugeridas en las líneas 12, 17 y 20. En las últimas líneas del pseudocódigo 3 se busca reemplazar todo el contenido de por el nuevo orden del alfabeto en la cadena M_f.

Se debe de tener en cuenta que v'_f, puede contener valores superiores a 64, por los tanto se ejecutan las líneas 12, 17 y 20, anteriormente mencionadas en el pseudocódigo 3, para que $index$ nunca sobrepase el número de símbolos disponibles del alfabeto de M_{b64}, y de esta forma se almacenen

Pseudocódigo 3 Transformación de M_{b64} con vector fractal v'_f en cadena M_f

1: **Inicio**
2: $list[]$=$v'_f[]$, $list2[]$,$i = 0$, $e = 0$, $index = 0$, $s1$= $b64$ // $list[]$ contiene los valores de v'_f, $list2[]$ tendrá el nuevo orden del alfabeto, $s1$ representa el alfabeto ASCII, $index$ es una variable temporal de almacenamiento del símbolo en curso
3: **for** $e \leq size\_cadena(s1)$-1,e++ **do** //Relacionar símbolos
4: **if** $list[i] \leq size\_elementos(s1)$ **then**
5: $index = list[i]$
6: $list2[e] = s1[index]$
7: e++
8: **else**
9: $index$ = (($list[i]$ / $size\_cadena(s1)$) - $abs((list[i]/size\_cadena(s1)))$ * $size\_cadena(s1)$ + $abs((list[i]/size\_cadena(s1))$ // Permite ocupar las posiciones vacías que no se llenaron en la primera condición
10: inic:
11: **end if**
12: **if** $buscar(list2[e], index)$!=1 **then** // Si el símbolo que representa $index$ no está almacenado en list2[e] se almacena
13: $list2[e] = s1[index]$
14: e++
15: **else**
16: i++
17: index= ($list[i]$ / $size\_cadena(s1)$) - $abs(list[i]$/size_cadena(s1)) * size_cadena(s1)+abs (list[i] / $size\_caden(s1)$) // Permite ocupar las posiciones vacías que no se llenaron en la primera condición
18: $list2[e] = s1[index]$
19: **end if**
20: ir a etiqueta inic; // Etiqueta generada para saltar condición si no entra en la condición
21: $list2[e] = s1[index]$
22: i++
23: **if** $i \leq numero\_elementos(list)$-1 **then**
24: $M_f = equivalencia\_intercambio(M_{b64}, b64, list2[])$ //Función para el intercambio de relaciones entre el alfabeto original y el alfabeto empleando el vector fractal, v'_f contenido en $list2$
25: **end if**
26: **end for**
27: **Fin**

correctamente los valores del nuevo alfabeto.

En la figura 11 se presenta el proceso de asociación entre la reasignación de las posiciones del alfabeto original y el nuevo alfabeto de forma detallada, a_n representan todos los caracteres del mensaje codificado en ASCII y los valores de las posiciones de cada dimensión fraccionaria contenidas en v_f permitirán establecer la relación establecer un nuevo orden para el alfabeto de la cadena M_{64}. Dicha relación se designa mediante el valor del alfabeto original, y la posición de una serie de elementos en elegidos en v_f. Finalmente se obtiene el nuevo vector v'_f (almacén de valor de posiciones de una sola variante de D_{MB}), donde los símbolos de M_{b64} se asocian con el nuevo vector para obtener un alfabeto de a'_n.

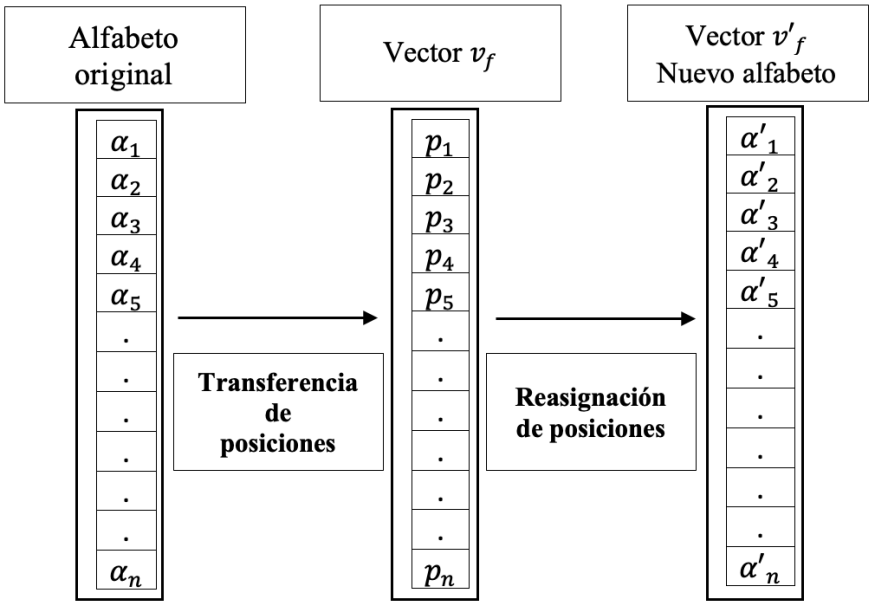


Figura 11: Intercambio de valores entre alfabeto original representado como α_n (procedentes de M_{b64}) y los nuevos valores representados en α'_n (encargados de conformar a M_f)

La figura 12 representa el desglose de la codificación vía vector fractal. A continuación, se describe el desglose de la dimensión fractal.

- Se selecciona la ecuación que generó mayor cantidad de variaciones en la dimensión fractal.
- Se selecciona la dimensión fractal de forma aleatoria, siempre y cuando no sea la primera que se generó del v_f, esto con la finalidad de elevar la seguridad de la propuesta.
- Posteriormente se obtienen las posiciones del vector en donde se encuentra dicha dimensión y se genera v'_f, v'_f es asociado con el alfabeto extraído de la cadena comprimida en base 64.
- Un ciclo es ejecutado para reasignar los valores del alfabeto y poder sustituir los valores del mensaje M_{b64}, relacionando el nuevo alfabeto con el alfabeto anterior siguiendo sus posiciones. Por ejemplo, el mensaje "abc" original pasa a a_{n+1} a_{n+2} a_{n+3}.
- Para volver a obtener de nuevo el mensaje original se obtiene del área de reglas la ecuación fractal, se calcula nuevamente v_f y se asocia con el alfabeto original, para realizar el proceso inverso de obtención de datos y generar el reemplazo de datos.

- En la codificación base 64 es importante asegurar que el mensaje sea integro, de lo contrario generará un error al momento de recuperar el mensaje.

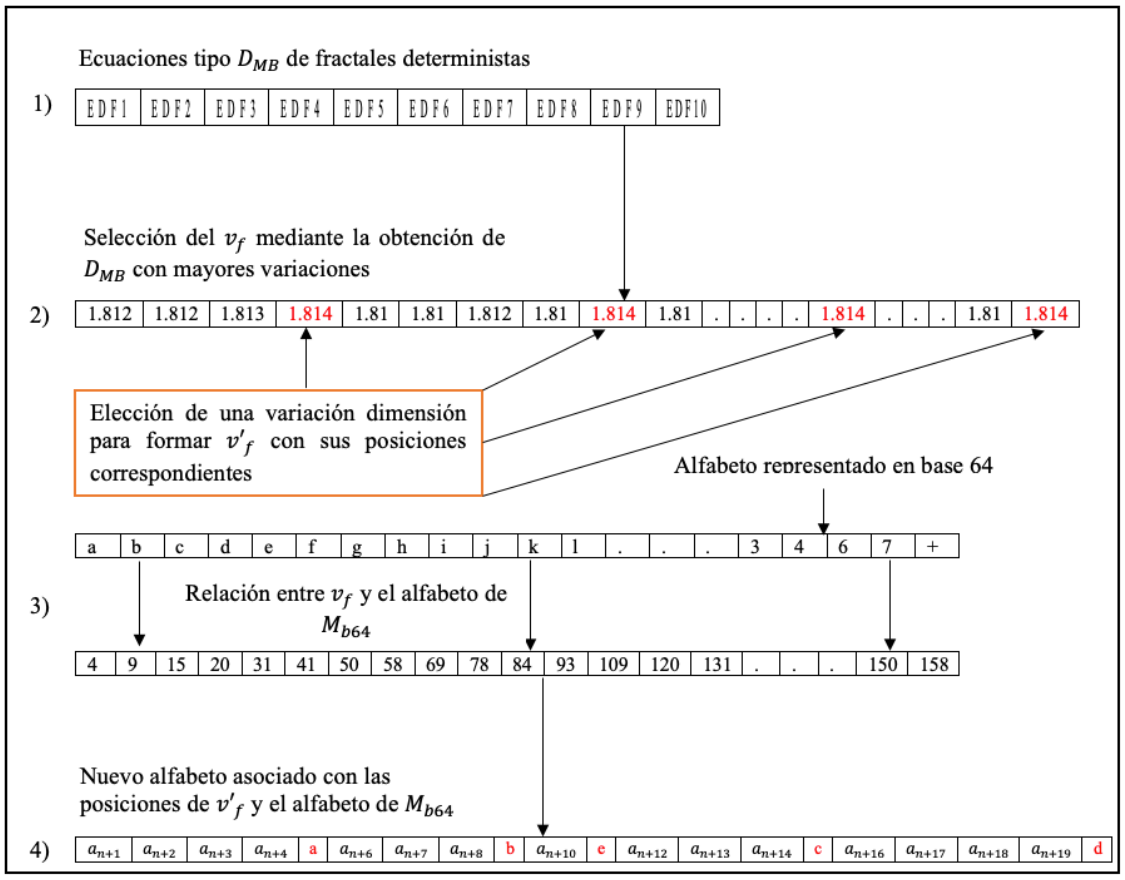


Figura 12: Desglose de la operación de la codificación vía vector fractal

Fase 4.- Codificación basada en UTF. Una vez que en M_f se han reemplazado todos los símbolos de M_{b64}, se reemplazan en M_f todos sus símbolos por una representación binaria UTF-6, con el objeto de reducir en aproximadamente un 25 % el tamaño de la cadena, y de esta forma obtener la cadena M_{UTF}. En el pseudocódigo 4 se muestra la transformación del mensaje correspondiente a la fase 4.

Pseudocódigo 4 Transformación general de M_f a M_{UTF}

```
1: Inicio
2: i = 0, M_f, M_UTF = ' '
3: while Size(M_f) ≤ i do // Cambio de tamaño de 8 bits a 6 bits
4:     M_UTF = M_UTF + Binarizar(M_f[i], (6)) // Solo se toman los 6 primeros bits de cada símbolo
5:     i ++
6: end while
7: Almacenar_datos(M_UTF) // Cadena con 6 bits por símbolo
8: Fin
```

En esta etapa, el modelo de codificación de UTF-6 se presenta en el Anexo A, donde la representación del alfabeto de base 64 puede ser representado con 7 bits o 6 bits, dependiendo el caso

que emplee, se puede obtener una disminución sobre el mensaje, que va del 12.5 % o del 25 %, según corresponda el caso. Al final de la etapa de transformación del mensaje, la imagen de portada se manipula para incrustar M_{UTF} en I'. Tarea correspondiente a la Etapa 2 *Manipulación del mensaje* del algoritmo propuesto.

3.2 Etapa 2: Manipulación de la imagen

La segunda etapa del método VVRSM corresponde a la manipulación de la imagen, la cual ocurre en cuatro fases que se ilustran en la figura 13 las cuales se describen a continuación.

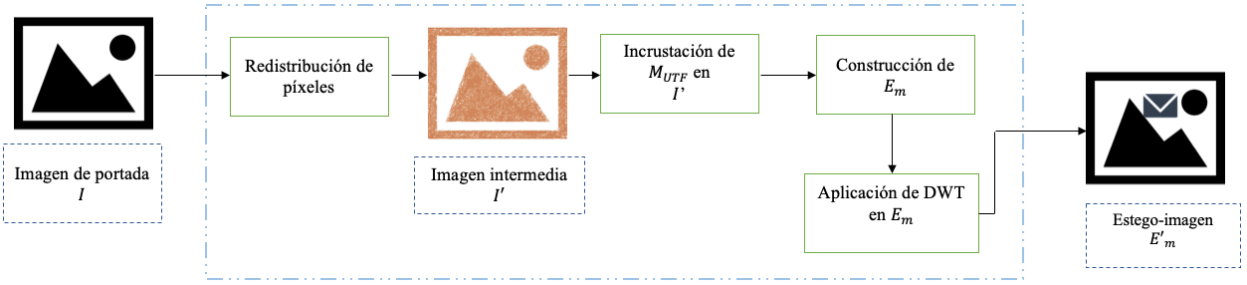


Figura 13: Diagrama general de bloques del método de esteganografía propuesto $VVRSM$ para manipulación de la imagen

Fase 1. Redistribución de píxeles. La redistribución de píxeles consiste en formar una nueva imagen a partir de los píxeles originales de la imagen de portada I para generar la imagen I' (imagen intermedia) destinada a recibir los datos de la cadena M_{UTF}.

En esta fase se almacenan los píxeles correspondientes a los canales R, G y B en una estructura de datos en la cual se indica un identificador representado como id, su posición (x, y), el valor de intensidad de píxeles representado en la variable e. Una vez almacenados los valores anteriores, se genera una nueva estructura de datos similar, con la diferencia que los datos de entrada corresponden a la función $g(id)$, que permite reordenar las coordenadas de los píxeles, manteniendo el valor original de su intensidad. La función $g()$, toma parámetro inicial id, debido a que este representa la posición actual, en esta función se debe de asociar una variable, como lo puede ser un conjunto de D_{MB} que permita que cambiar los valores de id, de forma única para todo el conjunto e píxeles disponibles en I. El proceso general de redistribución de píxeles se muestra en la figura 14.

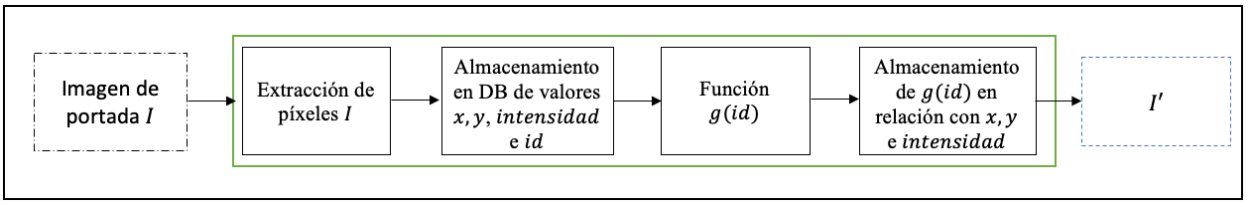


Figura 14: Fase de redistribución de píxeles y reconstrucción de imagen I'

La figura 15, muestra el proceso de la dispersión de píxeles, donde la base de datos permite la relación entre las posiciones originales y las posiciones temporales, para mantener la relación al construir E_m (primer estego-imagen). La indexación permite que la relación no desaparezca con los cambios de posiciones. Cuando I' se ha generado, la base datos o la estructura de datos es

eliminada, para evitar la recuperación de datos, y solo es recuperable la información al aplicar el proceso inverso de ocultamiento de datos sobre la estego-imagen final.

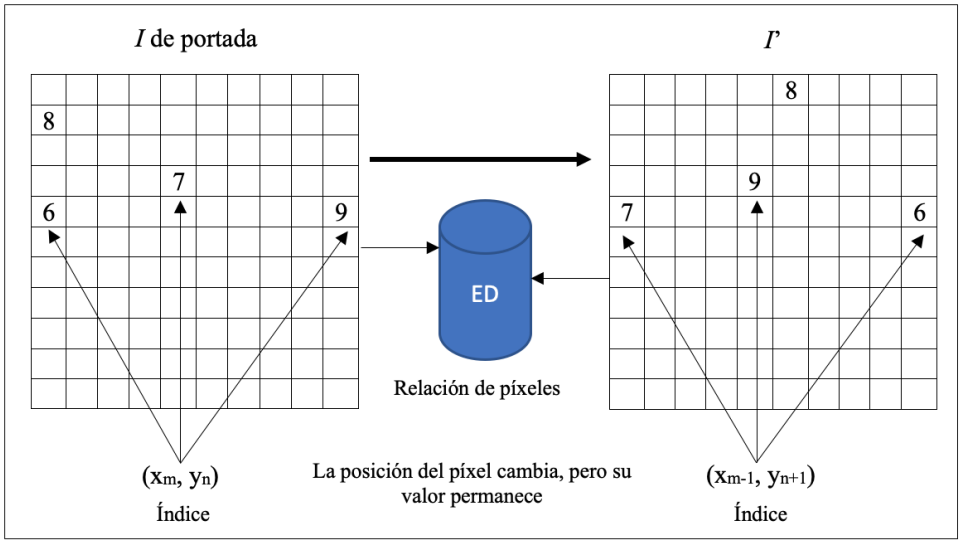


Figura 15: Esquema de redistribución de píxeles y reconstrucción de imagen I

El pseudocódigo 5 se expone la selección de la imagen de portada, y la generación de las estructuras que almacenen los valores originales de la imagen de portada y las estructuras que contendrán las modificaciones, esto se muestra de las líneas 3 a 6. De las líneas 7 y 11 se almacenan los canales de la imagen de portada con sus datos originales y de las líneas 12 a 16 se crean los canales virtuales los cuales están redistribuidos con una configuración diferente a la original. Finalmente, en la línea 17 se crea la imagen virtual en la cual se incrustarán los datos.

Pseudocódigo 5 Redistribución de píxeles para I

1: **Inicio**
2: $I = Extraccion(Imagen\_portada)$ // Obtener imagen de portada
3: Generar $ChannelsReal$, $ChannelsVirtual$, I' con campos id, x, y // Estructura de datos
4: $val = 0$, $val1 = 0$, $val2 = 0$, $i = 0$, $e = 0$, R, G, B // Valores inicializados
5: $A[i, e, canal] = I$ // Matriz inicializada para almacenar
6: **for** $x \leq M$ **do** // Ciclo de dispersión de píxeles
7: **for** $y \leq N$ **do**
8: $val = int(A[i, e, 0])$, $val1 = int(A[i, e, 1])$, $val2 = int(A[i, e, 2])$
9: $Insert\_values\ (ChannelsReal, id, i, e, val, val1, val2)$
10: **end for**
11: **end for**
12: **for** $i \leq M$ **do**
13: **for** $e \leq N$ **do**
14: Insertar valores en $ChannelsVirtual$ de $ChannelsReal$ ordenar por $g(ChannelsReal.id)$
15: **end for**
16: **end for**
17: $I' = ChannelsVirtual$ // Imagen virtual
18: **Fin**

Al terminar la redistribución de píxeles se genera un análisis previo sobre los píxeles, para determinar los cambios que puede generar la aplicación de DWT. Por lo general se ha considerado

que los píxeles con valores que están entre 0 y 1, provocan que el proceso no sea reversible, debido a que existe un desbordamiento de valores, por lo tanto, se realiza una lectura sobre los píxeles que presentan un riesgo potencial de que el mensaje incrustado no sea recuperado sin pérdida de datos, y por lo tanto se realiza una suma escalar para convertir estos píxeles a valores pares, para que al momento de que DWT genere valores fraccionarios puedan ser interpretados para ser reversibles sin pérdida de datos. Lo anterior se efectúa recorriendo todos los píxeles de la imagen, y al mismo tiempo se calcula la probabilidad que tienen de pasar de a un valor cerca el cual este esté en un intervalo de $[0:1]$ para evitar el desbordamiento de datos. Al terminar de generar la redistribución de los píxeles, se procede a seleccionar las zonas donde se incrustan los datos del mensaje y los datos de las reglas de recuperación del mensaje. De esta forma, en la tabla 3 se presentan los datos probables que pueden ser almacenados en imágenes I' con o sin compresión.

Tabla 3: Recomendaciones para incrustación de datos en I'

Dimensiones en píxeles	Total de píxeles disponibles	Bytes sugeridos a incrustar	Máximo sugerido de bytes a incrustar
64x64	12,288	6,144	8,601
128x128	49,152	24,576	34,406
256x256	196,608	98,304	137,625
392x392	460,992	230,496	322,694
512x512	786,432	393,216	550,502
1024x1024	3,145,728	1,572,864	2,202,009

En la tabla 3 se muestran los píxeles disponibles a incrustar en diferentes tamaños de imágenes, en la cual se sugiere incrustar el mensaje en al menos el 25 % del total de los píxeles I, la sugerencia indicada en la tabla 3 está basada en el análisis efectuado en la presente investigación, cuando se utiliza una técnica en dominio del espacio se pueden afectar dos bits sin perjudicar la calidad de la estego-imagen y 3 bits por píxel representa un límite entre el máximo almacenamiento y un PSNR cercano a los 34 dB. Por otro lado, si se emplea una técnica en el dominio de la frecuencia se puede considerar de dos a tres bits para insertar el mensaje, sin comprometer la calidad de la estego-imagen. En la última columna se muestra el máximo de bytes sugeridos a incrustar, basándose en los tres primeros bits disponibles en un píxel. El cálculo se puede obtener a través de la ecuación (23).

$$\mathrm{N_{uP}} = \mathrm{M} \times \mathrm{N} \times \mathrm{D_c} \times \mathrm{P_o} \tag{23}$$

Donde: N_{uP}, representa la cantidad de bytes sugeridos a incrustar, M: número de renglones de la imagen, N es el número de columnas de la imagen, D_c es el número de canales de la imagen y P_o representa el porcentaje de bits disponibles en la imagen de portada para modificar.

La cantidad de datos que pueden ser incrustados se calcula a través de la relación del mensaje que se deseen incrustar y del tamaño de la imagen que se emplee. Si no se consideran dichas variables, el proceso se puede detener por un error de escritura ocasionado por la falta de espacio de almacenamiento, o bien, provocando una degradación de la imagen de portada lo cual se puede apreciar con un análisis visual.

Tomando en cuenta las consideraciones antes mencionada, cuando I' ha sido formada (sin datos incrustados), se establece un desequilibrio de los píxeles pares e impares que permiten generar un patrón de confusión ante el uso de una herramienta de estegoanálisis, debido a que existen herramientas que buscan (como en RS) la cantidad de píxeles pares e impares que contiene una imagen, puesto que, en una imagen sin modificaciones, generalmente, la diferencia oscilan entre 3 % y 4 % con respecto a la cantidad de píxeles con valor par o impar. Con ello, se evita que las acciones del mecanismo de incrustación espacial sean detectadas, este proceso emplea la ecuación (24) donde Di_c es el canal de la imagen que almacena las modificaciones realizadas en la matriz C_{ij}, siendo h la variable encargada de realizar el factor de desajuste el cual puede ser positivo o negativo, al finalizar la operación de desequilibrio se genera I' con el factor de desajuste. El factor de desequilibrio se presenta en la ecuación (24).

$$Di_c = \sum_{j=0}^{M-1} \sum_{i=0}^{N-1} Ci, j + h \tag{24}$$

M y N son las dimensiones de la imagen, i e j son variables de inicio y $C_{i,j}$ es un píxel que recibe un valor de la variable h.

Mediante la ecuación (24) es posible realizar cambio en el orden en la cantidad de píxeles pares e impares, por lo tanto, y busca establecer un preámbulo que permita marcar una diferencia del 3 % con respecto a los píxeles pares e impares cuando se incruste un mensaje en la estego-imagen.

Este proceso es reversible, debido a que los píxeles que son modificados pueden ser nuevamente restablecidos, puesto que es una función espacial, la cual está destinada a sumar o restar una unidad sobre el valor del píxel, como máximo se sugiere una modificación del 33 % de los píxeles disponibles para cada canal.

Fase 2. Incrustación. Esta fase consiste en leer todos los píxeles de los canales I', con la excepción de los píxeles destinados a M_r (mensaje destinado a incluir reglas generales de recuperación de datos) y continuar con la lectura de la cadena M_{UTF}, donde el tamaño total de esta cadena se calcula con respecto a los píxeles disponibles en I', considerando como máximo los tres primeros bits de cada píxel para incrustar el mensaje; de lo contrario, sólo se podrá incrustar M_{UTF} de forma parcial. La cadena M_r, representa una concatenación de las básicas de operación, tales como, la ecuación fractal, redistribución de píxeles, entre otros, y está ligada intrínsecamente ligada con el analizador léxico y sintáctico propuesto en el anexo C. En la figura 16 se presenta el diagrama general de incrustación de datos, donde los bits disponibles M_{UTF} con incrustados en I', para que al terminar el proceso se genere la función inversa de I' y dar paso a la construcción de la estego-imagen E_m.

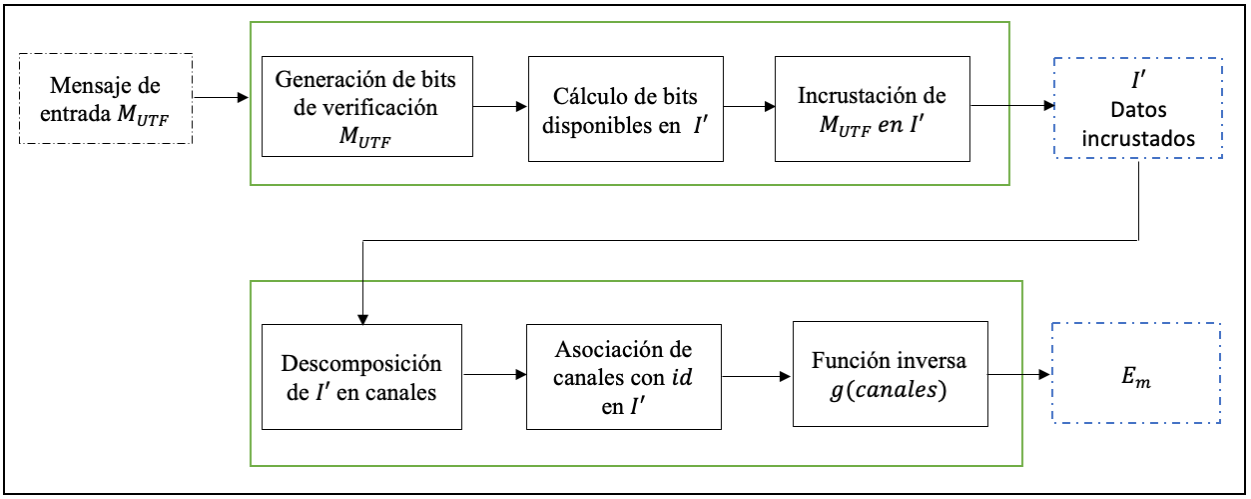


Figura 16: Incrustación de datos en I'

En el pseudocódigo 6 se lee I', la cual representa la imagen virtual, para descomponer sus canales, para posteriormente empezar a incrustar el mensaje que ha sido representado con símbolos de una longitud de 6 bits, y su vez almacenado en M_{UTF}, posterior a esto se genera un triple ciclo que esta expresado en las líneas 9 a la línea 20, donde el primer ciclo desplaza los canales de I'. Los siguientes ciclos permiten el desplazamiento a través de los canales para poder incrustar en el bit correspondiente M_{UTF} de cada píxel, hasta terminar con el mensaje de M_{UTF}. De las líneas 23 a 31 se inicia el proceso de redistribución de píxeles para formar la estego-imagen E_m.

Pseudocódigo 6 Incrustación de datos M_{UTF} en I'

Inicio
Leer (I')
Cargar estructuras $ChannelsVirtual$, $ChannelsReal$
R_{xy}, G_{xy}, B_{xy} = I' // Carga de canales de I'
$r = 0$, $Binarytemporal =$ ", $x = 0$, $y = 0$, $indice$, $seccion$ // inicializar variables
$bmaster$, E_m //desplazamiento por bits de cada píxel y la estego-imagen E_m inicializada
$M_r = Reglas\_fractal + Reglas\_marca$ // Se almacenan las reglas de indicio del vector fractal y condiciones generales
$M_{UTF} = leer(binaryUTF\text{-}6)$
for $r \leq size\_cadena(M_{UTF})$ **do** // Incrustar mensaje
    for $x \leq M$ **do**
        for $y \leq N$ **do**
            if ($temporary \leq M \times N$) **then**
                Recorrer canal $Binarytemporal$ = Binary(R_{xy}).position[0:bmaster] + $M_{UTF}[r]$
                $R_{xy} = Convertir\_Decimal(Binarytemporal)$
                $G_{xy} = Convertir\_Decimal(Binarytemporal)$
                $B_{xy} = Convertir\_Decimal(Binarytemporal)$
                $temporary = temporary + 1$, $r = r + 1$
            end if
        end for
    end for
    Suma bit $bmaster = bmaster + 1$
end for
I'=I'[y-seccion][x-seccion][canal] + M_r // Reglas de recuperación incrustadas en I'
for $canal \leq 3$ **do** // Ciclo de reordenamiento de valores de I' a estego-imagen E_m
    for $x \leq M$ **do**
        for $y \leq N$ **do**
            Buscar ($indice$, $ChannelsVirtual$ en $ChannelsReal$)
            $E_m[x][y][canal] = valor\_de(ChannelsVirtual, canal_{xy})$, $posicion(ChannelsReal)$
        end for
    end for
end for
E_m = // Estego-imagen formada de canales reordenados
$Destruir\_estructura\_datos()$// Se elimina todo el almacenamiento temporal y los canales virtuales
Fin

Fase 3. Reconstrucción. Una vez que se finalice la incrustación de M_{UTF}, se realiza el proceso de reconstrucción en I', que consiste en relacionar la primera estructura de datos que contiene las coordenadas originales provenientes de I, con respecto a I', de esta manera se restauran las coordenadas originales pero los valores de las intensidades de los píxeles cambian. Finalmente, el resultado obtenido es la estego-imagen E_m.

Antes de aplicar la DWT, las reglas de redistribución de píxeles se agregan a la cadena M_r (cadena que contiene reglas de recuperación del mensaje incrustado), donde se incluyen las ecuaciones de reordenamiento, los bits de terminación del mensaje y la constante b. El contenido final de M_r debe registrarse en los bordes de E_m, debido a que son regiones donde se puede extraer rápidamente las reglas de recuperación de información, además de que las reglas están contenidas en la zona del tercer bit menos significativo de cada píxel, siendo que estos píxeles no se modificaron para la incrustación del mensaje M_{UTF}.

Fase 4. Aplicación de la transformada DWT. En esta fase E_m es transformada mediante las transformadas DWT e IDWT para modificar el mapa de coeficientes en las cuatro subbandas que son generadas por DWT en la estego-imagen, adicionando una constante b en cada subbanda, la cual puede ser positiva o negativa para los píxeles de E_m. Finalmente se obtiene la imagen que contiene el mensaje incrustado E'_m siendo la estego-imagen final. Las ecuaciones (25), (26), (27) y (28) representan la adición de la constante b en las subbandas formadas por la transformada DWT. El pseudocódigo 7 muestra el proceso general de la aplicación de la técnica DWT, correspondiente a la cuarta fase de la etapa de *Manipulación de la imagen*.

$$C_{LL} = \sum_i \phi\left(n_1, n_2\right) + b \tag{25}$$

$$C_{LH} = \sum_i \psi^H\left(n_1, n_2\right) + b \tag{26}$$

$$C_{HL} = \sum_i \psi^V\left(n_1, n_2\right) + b \tag{27}$$

$$C_{HH} = \sum_i \psi^D\left(n_1, n_2\right) + b \tag{28}$$

Donde ϕ es la matriz que representa la subbanda LL, ψ^H es la subbanda LH, ψ^V es la subbanda HL y ψ^D es la subbanda HH, n_1 y n_2 son coordenadas de cada subbanda y b es una constante.

El pseudocódigo 7 muestra el proceso general de la aplicación de DWT. En este pseudocódigo se obtiene E_m para generar las subbandas HH, HL, LH, LL, posterior a esto en un ciclo se ingresa b, el cual puede tener valores negativos o positivos, pero no ambos al mismo tiempo, debido a que esto generaría un problema al recuperar los datos. Cuando b ha sido agregada en las subbandas se procede a se realiza el proceso de IDWT para generar E'_m, la cual es la estego-imagen definitiva.

Pseudocódigo 7 Aplicación de DWT en E_m

```
1: Inicio
2: E_m = Direccion(Estego − imagen) //Cargar estego-imagen E_m
3: Obtener [columnas, filas] = E_m, [HH, HL, LH, LL] = DWT(E_m) // Generar subbandas
4: i = 0, e = 0, b=0
5: for b = +/- .1, b <= +/-1 do
6:     for i ≤ columnas do
7:         for e ≤ filas do
8:             HH[i][e] = HH[i][e] + / − b
9:             HL[i][e] = HL[i][e] + / − b
10:            LL[i][e] = LL[i][e] + / − b
11:            LH[i][e] = LH[i][e] + / − b
12:        end for
13:    end for
14:    if Verifica_perdida(IDWT(HH, HL, LH, LL))==0 then
15:        Continua_ciclo()
16:    else
17:        Termina_ciclo()
18:    end if
19: end for
20: E'_m = IDWT(HH, HL, LH, LL) // Estego-imagen final almacenada en memoria
21: Fin
```

La figura 17 muestra el proceso general de la adición de la constante b en las subbandas de la técnica de DWT, cuando las subbandas son obtenidas de E_m para posteriormente generar la estego-imagen final E'_m.

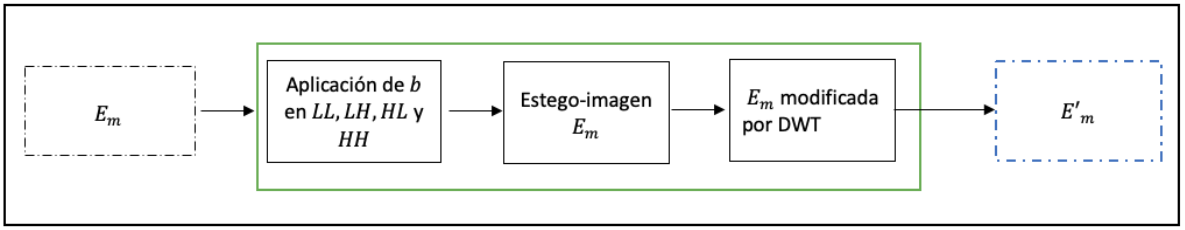


Figura 17: Aplicación de la transformada DWT en E_m

En la figura 18 se esquematiza de forma detallada la aplicación de DWT, cuando E_m (color azul) se ha construido, y b modifica las 4 subbandas de forma escalar (con adición o sustracción del valor asignado a b) para generar E'_m (color verde) como anteriormente se ha descrito en la fase aplicación de DWT en E_m.

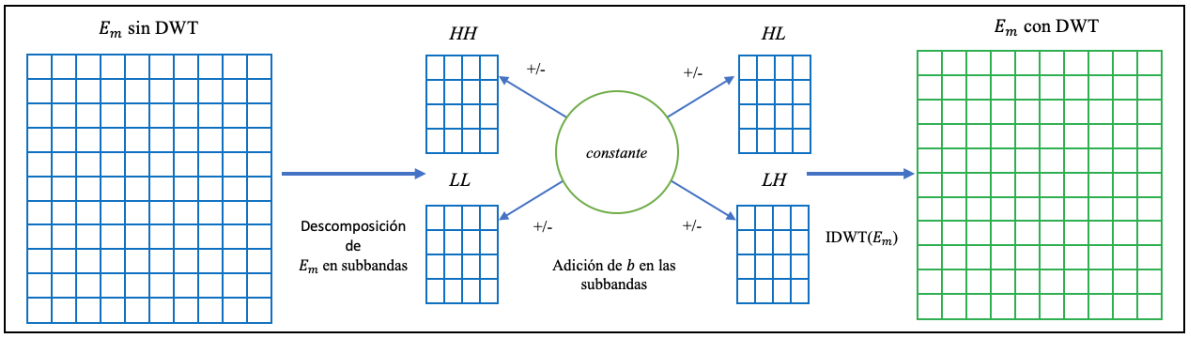


Figura 18: Diagrama detallado sobre la modificación de E'_m con DWT

En la figura 19 se muestra un diagrama esquemático de cómo se presentan las modificaciones en una estego-imagen al momento de que se han realizado los procesos de dispersión de píxeles, incrustación de la información, y la aplicación de la DWT.

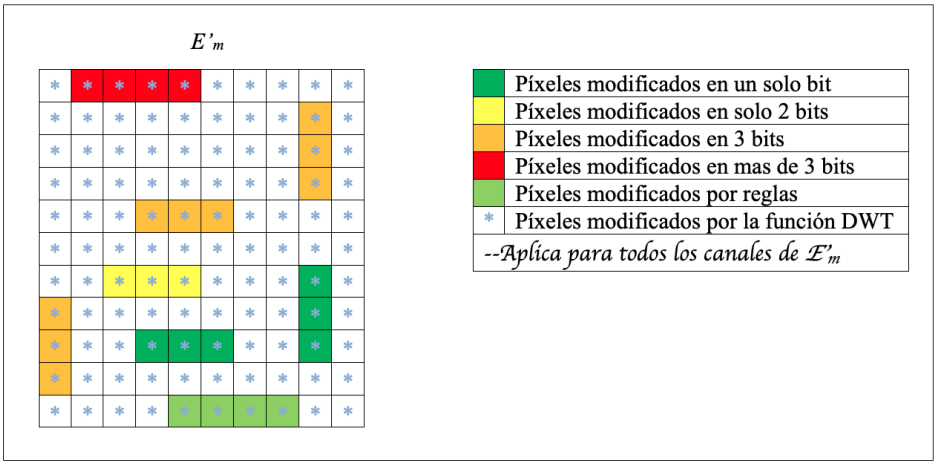


Figura 19: Diagrama representativo de modificación en E'_m al terminar de aplicar DWT

Como se puede observar en las etiquetas de la figura 19, el color verde intenso representa a todos lo píxeles que han sido modificados en el bit menos significativo, el color amarillo indica una modificación de dos bits por píxel, posteriormente el color amarillo intenso indica una modificación por tres bits en un píxel, el color

rojo indica los píxeles que han sido modificados en más de 3 bits, finalmente el color verde inferior representa todos los píxeles modificados para las reglas de recuperación. Por otra parte, estos casos se pueden presentar si y solo sí se ocupan todos los píxeles de cada canal, y es posible encontrar que exista en dos canales píxeles modificados en 1 o 2 bits, otro caso es una combinación de dos y tres bits modificados, el siguiente caso es píxeles modificados en tres o más bits, pero los cuatro casos no se observan en una estego-imagen. Los asteriscos indican la aplicación de b en todos los píxeles por DWT, los cuales permiten modificar los valores del mensaje para evitar una recuperación directa. Los píxeles que están en color blanco son píxeles que no han sido modificados, debido a que no es necesario modificar todos los valores si el mensaje no es lo suficientemente extenso.

En el anexo C, se presentan generalidades sobre la calidad de las estego-imágenes y el mecanismo para la validación de las reglas de recuperación del mensaje incrustado en las estego-imágenes, donde se da una propuesta general de un analizador léxico y sintáctico para la validación de cadenas.

3.3 Recuperación del mensaje en la estego-imagen

El proceso de recuperación del mensaje incrustado en la estego-imagen E'_m, consiste en:

- 1. Eliminar la constante b agregada en E'_m.
- 2. Lectura de la zona en donde se encuentran las reglas de incrustación de datos.
- 3. Reconstrucción del alfabeto utilizado para la codificación.
- 4. Generación del alfabeto original.
- 5. Recuperación y decodificación del mensaje.

La estego-imagen E'_m debe ser sometida al proceso inverso de la incrustación del mensaje para poder obtener sin pérdida de información los datos incrustado en la estego-imagen, para que suceda lo anterior, es necesario eliminar en primer lugar, la constante b agregada en E'_m. Posteriormente, se leen los bordes de la estego-imagen para extraer la ecuación fractal utilizada en el proceso de incrustación al igual que su dimensión. La figura 20 muestra de forma general el proceso de eliminación de la constante b en la estego-imagen E'_m, de tal forma que se puede interpretar que el proceso consiste en generar de forma ascendente o descendente el valor de b, el cual va de 0 a c_1, donde los valores de c_1 son menores a 0.1, con valores incrementales recomendados de 0.1 en 0.1 o de 0 a c_1 con valores recomendados de -0.1 a -0.1 de forma decremental, en este caso el valor de c_1 es mayor a -0.1.

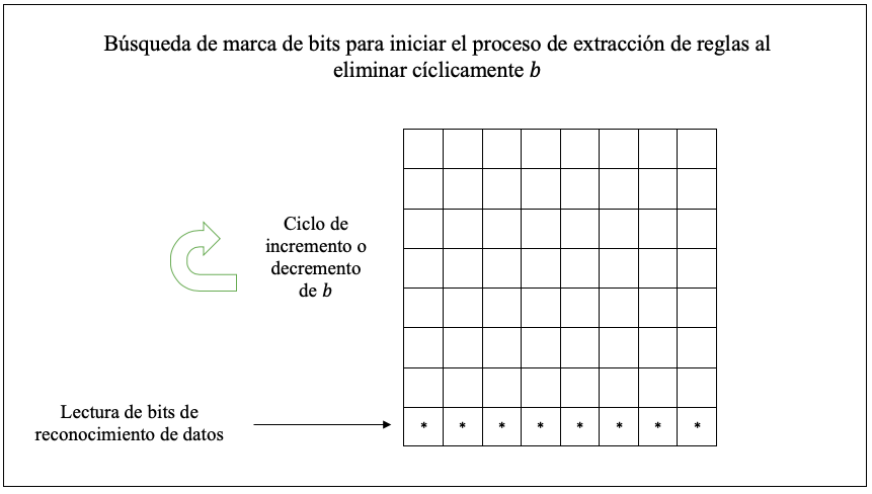


Figura 20: Búsqueda de marca de bits para iniciar el proceso de extracción de reglas al eliminar cíclicamente b

En cada iteración en la que se incrementa o decremental b a través de DWT, se leen los bits de la marca ubicada en los últimos píxeles de la estego-imagen correspondientes al tercer bit (indicador de 8 bytes) para

corroborar que la eliminación de máscara ha sido exitosa, y de esta forma poder empezar la recuperación de las reglas del mensaje, lo cual derivaría en la recuperación del mensaje incrustado. Las reglas de recuperación son analizadas mediante un analizador léxico y sintáctico propuesto en el anexo C, para validar la coherencia de las cadenas recuperadas.

Una vez obtenidas las reglas que se utilizaron en la incrustación de datos, se debe realizar la reestructuración tanto del alfabeto original de la codificación (como puede ser base 64) como de las reglas que se formaron a partir de la imagen intermedia. Al reconstruir la imagen intermedia, se lee la cadena de verificación para extraer M_{UTF}, el cual es decodificado usando el vector fractal y la base utilizada (base 64). Finalmente se realiza el proceso descompresión, para recuperar el mensaje incrustado, denominado como M_{rec}. El proceso de recuperación de datos se ilustra en la figura 21.

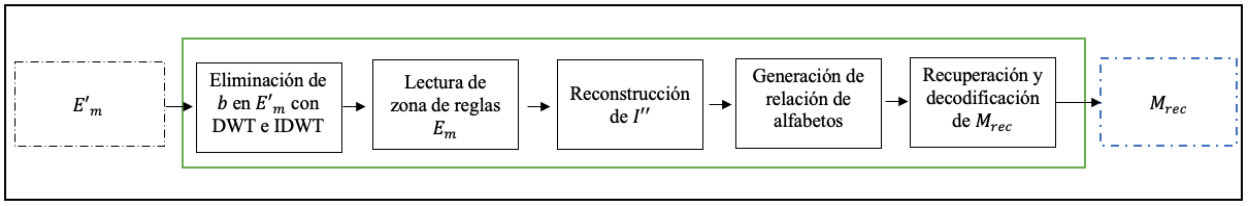


Figura 21: Recuperación de datos de la estego-imagen E'_m

3.4 Mejora del método a través de la implementación SVD con DWT

El método VVRSM está diseñado para lograr incrustaciones superiores a los 6.0 bpp cuando el mensaje a embeber es texto. Mientras que, cuando el mensaje a incrustar es vídeo, audio o imagen, el método propuesto logra incrustar 2.1 bpp en promedio.

Los resultados presentados por el método propuesto cumplen con el objetivo de incrustar 6 bpp, sin embargo, es posible lograr mayores cargas de incrustación cuando se aprovechan mecanismos basados en marcas de agua enfocados a perder la menor cantidad de datos al incrustarse en imágenes portadoras. Es por ello que se analizó la manera de adicionar una fase previa cuando los datos sobrepasan la modificación de 2 bits en la imagen de portada al emplear VVRSM, con ello, se incrementan las cualidades visuales de la estego-imagen, además de reducir la tasa de bits detectados por técnicas de estegoanálisis.

En esta fase de preprocesamiento se aprovechan las cualidades de SVD (*Single Value Decomposition*) y de DWT para incrustar un mensaje M_i en I. De forma general se tiene que M_i es transformado en una imagen de tipo RGB (si originalmente no lo es), la cual a su vez debe ser incrustada en una imagen RGB de baja resolución, lo cual daría como resultado una estego-imagen de un tamaño de entre 3 y 4 veces menor al tamaño original de M_i cuando ha pasado por el proceso de compresión. El proceso de SVD-DWT presenta la particularidad de que al recuperar la imagen incrustada existe pérdida de datos, y por tal motivo se deben de recuperar los datos perdidos, para que posterior a esto, se sumen tanto la estego-imagen y los datos que se perdieron en el proceso de recuperación. Lo anterior permite que de la suma de los dos elementos anteriores se obtenga un mensaje de menor tamaño con relación a M_i, lo cual es variable, pero se esperarían reducciones de hasta un 33 % con respecto al mensaje comprimido en una fase normal de VVRSM.

La figura 22 representa de forma general el proceso de incrustación de M_i en D_r mediante DWT y SVD, esta mejora del método se busca la obtención de una imagen intermedia de baja resolución, la cual sea capaz de albergar un mensaje convertido en imagen, con la finalidad de reducir el tamaño de M_i, este proceso presenta pérdida de datos, por tal motivo la estego-imagen resultante de SVD-DWT se suma a la pérdida, y estos dos elementos son nuevamente tratados como un mensaje M_i.

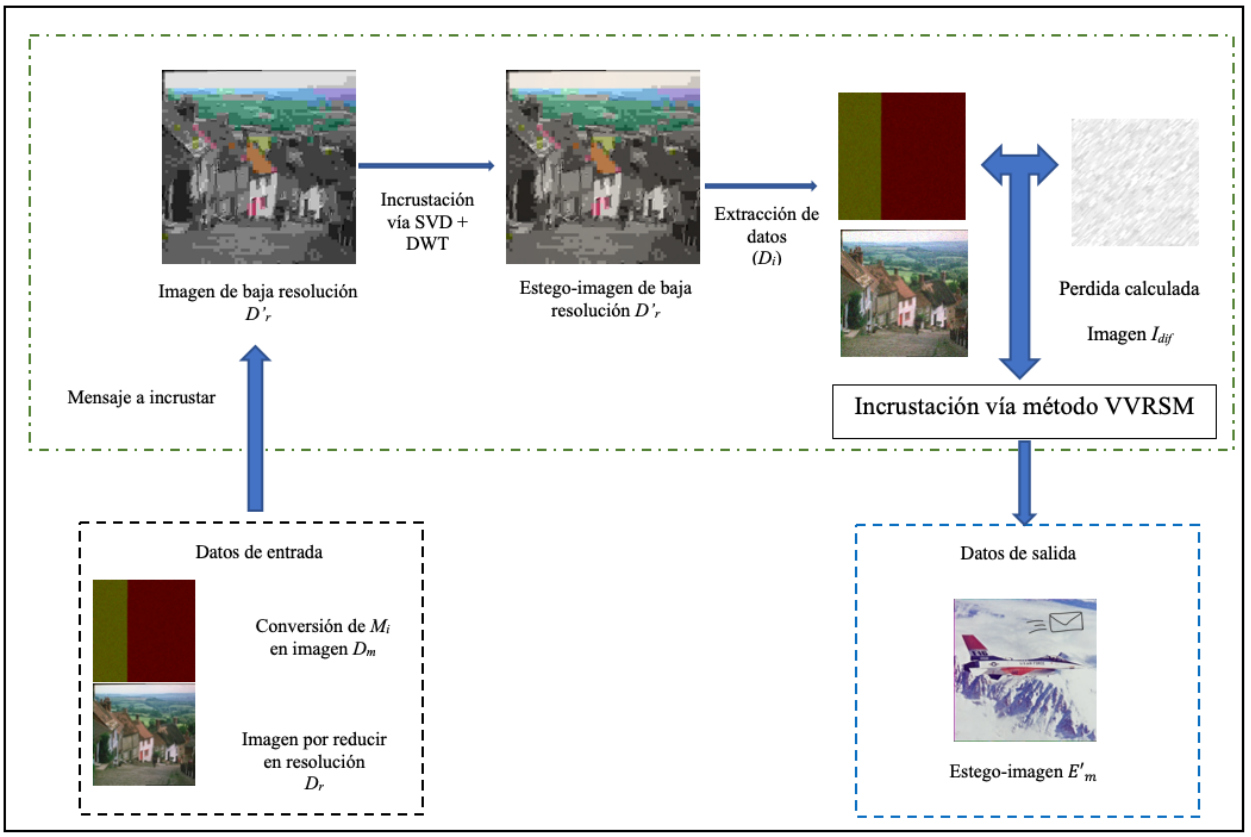


Figura 22: Diagrama de combinación de SVD y DWT

La etapa de preprocesamiento del mensaje contiene los siguientes pasos:

- El mensaje M_i es convertido en una matriz RGB, la cual representa una imagen con compresión sin pérdida de datos. M_i sigue el proceso de compresión con LZW y en base 64, para generar la cadena M_{b64}, de esta cadena, los símbolos son convertidos en su forma numérica y se representa un píxel por cada símbolo, para dar lugar a una nueva cadena denomina como M_n.
- M_n es transferida a una imagen denominada como D_m (la cual es incrustada vía SVD-DWT). Posterior a la generación del mensaje como imagen se procede a generar una imagen de portada denominada como D_r, la cual es una imagen de baja resolución, y puede ser con pérdida o sin pérdida por compresión, con el objetivo de obtener una estego-imagen con el menor tamaño posible con relación a M_i.
- La imagen D_m es incrustada en D_r con DWT, de tal forma que se extraen las wavelets de la imagen y posterior a esto se aplica un proceso de multiplicación mediante los coeficientes obtenidos de la DWT. El siguiente paso consiste en que las matrices obtenidas son modificadas por SVD para cada canal de D_r. Los canales de D_r son multiplicados por una constante c_i y mediante un proceso de *merging* se genera la estego-imagen D'_r y de esta forma se obtiene la primera mitad del mensaje.
- Al finalizar la incrustación de D_m, se extrae la información que se incrustó en D'_r y se calcula la pérdida de datos. Al calcular la pérdida de datos se genera una imagen, denominada como I_{dif}, la cual contiene los valores perdidos en el proceso de incrustación de D_m en D_r.

El pseudocódigo 8 presenta el mecanismo para incrustar M_i en una imagen D_r.

Pseudocódigo 8 Incrustación de M_i en D_r

Inicio
Leer(M_i) // Mensaje inicial
Leer(D_r) // Imagen a convertir en baja resolución
$D_m = Generar\_imagenRGB()$ // Imagen RGB vacía
$D'_r = Reducir\_resolucion(D_r)$ // Generar imagen con baja resolución
M=0, N=0, $canal$=0, D_i // D_i es la imagen a recuperar
if Tipo_mensaje(M_i) == "datos" o Tipo_mensaje(M_i)!= "imagen diferente" **then**
    M_{LZW} = LZW(M_i)
    M_{b64} = B64(M_{LZW})
end if
$i = 0$ // Variable de recorrido de datos
while $canal \leq 3$ **do**
    for $size\_cadena(M_{b64})$, M++ **do**
        for $size\_cadena(M_{b64})$, N++ **do**
            D_m[M][N][canal] = M_{b64}[i]
            i++ // Recorre cadena comprimida
        end for
    end for
    canal++ // Cambia de canal
end while
D'_r = DWT(D_r) * SVD(D_m) * c_i // Proceso de merging para crear D'_r
$D_i = D'_r$ * IDWT * ISVD // Inversa del proceso de incrustación
D_i // Mensaje recuperado
Fin

Al obtener D_i es necesario el porcentaje de pérdida de datos, debido a que el proceso de incrustación SVD-DWT presenta pérdida de información. El cálculo de la imagen que representa la pérdida de datos mediante la diferencia está dado por la ecuación (29).

$$I_{dif} = D_m - D_i \tag{29}$$

Donde D_m - D_i <>0.

Una consideración es que el mensaje M_i permita que la compresión aplicada vía LZW y por base 64 reduzca el tamaño del mensaje, en caso contrario, el resultado será un incremento de M_i y tendrá un efecto negativo con respecto a las cualidades de la estego-imagen final.

Por otro lado, el pseudocódigo 9 presenta la generación de la imagen I_{dif} para la recuperación del mensaje oculto en su totalidad.

Cuando se ha generado I_{dif}, esta imagen es convertida en una cadena de mediante el empleo de las funciones $LZW()$ y $B64()$, posterior a esto, D'_r es convertido en cadena y ambos resultados se concatenan, de tal forma que el resultado final se agrega a una nueva cadena denominada como M'_i, y es en este punto donde se reinicia el proceso de incrustación del mensaje como se realizó en el método VVRSM. Lo anterior se puede visualizar de forma general en la ecuación (30).

$$M'_i = B64(LZW(I_{dif})) + B64(LZW(TipoD)) + B64(LZW(D'_r)) \tag{30}$$

Donde la variable $TipoD$ especifica la separación entre los datos recuperados en I_{dif} y la estego-imagen D'_r para evitar la pérdida de datos cuando se efectúe el proceso de recuperación del proceso de incrustación. Todas las cadenas son concatenadas.

Pseudocódigo 9 Recuperación de pérdida de datos de D'_r

Inicio
Cargar(D_i) // Cargar imagen que representa el mensaje recuperado de D'_r
Leer(D_m) // Imagen con mensaje ASCII en base numérica
I_{dif} = = $Generar\_imagenRGB()$ // Imagen RGB vacía
$i = 0$, $e = 0$, $canal = 0$, $TipoD$ = "Segmento de datos"
while $canal \leq 3$ **do**
    [M, N] = D_i // extracción de dimensiones
    for $i \leq M$ **do**
        for $e \leq N$ **do**
            if D_i[i][e]!=D_m[i][e] **then**
                I_{dif}[i][e] = D_m[i][e]
            end if
        end for
    end for
    canal ++ // Incrementa canal
end while
Grabar I_{dif} // Imagen con datos recuperados
$M'_i = B_{b64}(LZW(I_{dif} + B64(LZW(TipoD)) + D'_r))$ //Mensaje a incrustar en la imagen de portada definitiva, el cual es comprimido y representado en base 64 a 6 bits por símbolo
Fin

Una de las consideraciones finales sobre el preprocesamiento de M_i es que, aunque D'_r consiga un tamaño de 10 a 15 veces menos que M_i comprimido, no es garantía de que el producto final en D'_r presente un tamaño inferior a M_i comprimido, debido a que se debe de tener en cuenta la pérdida que se obtiene y es por ello que se sugiere el pseudocódigo 10 para lograr el máximo ajuste posible, debido a que, sí la imagen D_r presenta una gran cantidad de zonas con tendencia heterogénea, el algoritmo de compresión empleado en la imagen, efectuaría una baja compresión en D_r, y por lo tanto D'_r sería de mayor tamaño que en lo logrado con VVRSM sin SVD-DWT.

Pseudocódigo 10 Ajuste de D_r para compresión

Inicio
Crear_imagen(D_r) // imagen de baja resolución
i, e // Coordenadas de D'_r
while $Size\_bytes(D'_r) > size(M_i)$ **do** // Tamaño en bits
    for $i \leq factor$ **do**
        for $e \leq Factory$ **do**
            $D'_r[i][e] = Recalcular(D'_r[i][e])$ // Se reduce el tamaño D'_r
            $D'_r[i][e] = D'_r()$ // Llamar nuevamente a pseudocódigo 9
        end for
    end for
end while
Mensaje: "Reducción adecuada"
Fin

Esta mejora del método tiene la particularidad que el proceso tanto de incrustación como de recuperación de información consuma un 50 % de recursos adicionales en relación con VVRSM sin SVD-DWT, en este sentido es necesario que se considere cuando las cargas de incrustación de datos se encuentran por arriba

de los 4 bpp, además de que afecte más de un 20 % del total de los tres bits disponibles, puesto que las modificaciones sobre estos bits generan distorsiones notables sobre las estego-imágenes.

En el siguiente capítulo se presenta el desarrollo de pruebas con el método VVRSM para verificar la capacidad de incrustación sobre conjuntos amplios de imágenes, y de esta forma verificar el comportamiento que puede tener distintas resoluciones con imágenes RGB. Las pruebas se enfocarán en medir la calidad de las estego-imágenes resultantes que sean procedentes de VVRSM, además este método se comparará con otras propuestas de esteganografía para tener puntos de referencia sobre su desempeño.

Capítulo 4

Pruebas y resultados

En este capítulo se presentan las pruebas y resultados obtenidos al aplicar el método VVRSM. El método VVRSM está codificado en Python versión 2.7.16 conjuntamente con Matlab 2015b, el gestor de Base de Datos utilizado para el almacenamiento temporal de los píxeles y sus coordenadas es SQLite versión 3. La selección del gestor de Base de Datos se hizo dado al bajo consumo de recursos de hardware que contempla. El equipo de cómputo empleado para la codificación de VVRSM es un MacBook Air Core i5 a 2.5 GHz, con 8 GB de memoria RAM y sistema operativo macOS 10.14.5.

4.1 Validación del método de esteganografía VVRSM

Con la finalidad de validar el método de VVRSM se realizaron tres escenarios para la realización de pruebas, con las siguientes características:

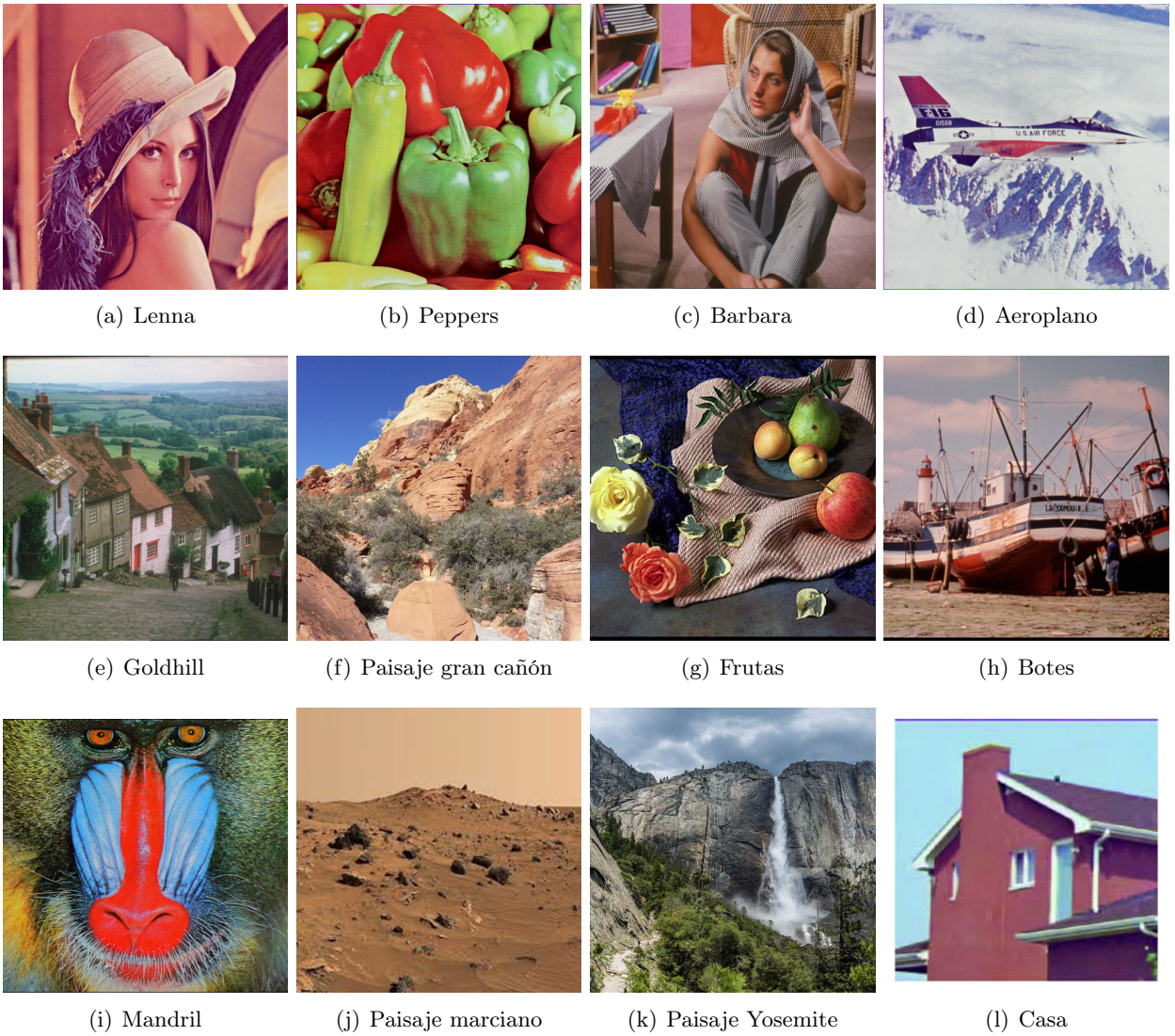
- **Escenario 1:** Selección de 12 imágenes de portada con modelo de color RGB, con una resolución de 512×512 píxeles, el mensaje seleccionado para incrustar es del tipo cadenas de texto. Las imágenes de portada que se emplearon en esta primera prueba se muestran en la figura 23.

- **Escenario 2:** Selección de tres subconjuntos de imágenes con distintas resoluciones:
  - Resolución 1: 1000 imágenes de tipo RGB con resolución de 64×64 píxeles.
  - Resolución 2: 1000 imágenes de tipo RGB con resolución de 128×128 píxeles.
  - Resolución 3: 1000 imágenes de tipo RGB con resolución de 256×256 píxeles.
  - Origen de las imágenes: Dataset Tiny y COCO ([124], [90])
  - Objetivo: Embeber texto con una carga máxima de 5 bpp.

- **Escenario 3:** Selección de 6000 imágenes con formato JPEG conservando el mismo tamaño de las imágenes de portada provenientes todas ellas de los dataset Tiny y COCO, la selección es aleatoria.
  - Resolución 1: 1000 imágenes de tipo RGB con resolución de 64×64 píxeles.
  - Resolución 2: 1000 imágenes de tipo RGB con resolución de 128×128 píxeles.
  - Resolución 3: 1000 imágenes de tipo RGB con resolución de 256×256 píxeles,
  - Origen de las Imágenes: Dataset Tiny y COCO ([90], [124])
  - Objetivo: Embeber una imagen de iguales dimensiones en una imagen por cada imagen de portada de forma aleatoria y obtener el mejor resultado posible en el desempeño de métricas.

Los canales R, G y B de las imágenes de portada, se trabajan de forma distinta. En el canal R se utiliza una función de ordenamiento del tipo seno hiperbólico, mientras que, para los canales G y B se emplea una función de ordenamiento del tipo tangente hiperbólico. Dicha selección se realiza con el objeto de que la distribución de los píxeles sea de forma mayoritariamente heterogénea para los canales B y G. Las funciones de ordenamiento son modificadas por factores pseudoaleatorios de tal forma que la distribución en la imagen virtual sea distinta.

La ecuación (31) es empleada para generar la distribución en el canal R, mientras que la ecuación (32) para generar la distribución en los canales G y B (esta ecuaciones son aplicaciones de la función $g()$ propuesta en el



(a) Lenna (b) Peppers (c) Barbara (d) Aeroplano

(e) Goldhill (f) Paisaje gran cañón (g) Frutas (h) Botes

(i) Mandril (j) Paisaje marciano (k) Paisaje Yosemite (l) Casa

Figura 23: Imágenes de portada empleadas en la validación del método de esteganografía VVRSM para el escenario de Pruebas 1

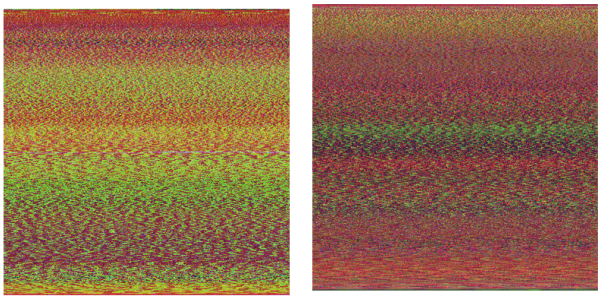
capítulo 3, etapa 2), en estas ecuaciones x e y son las posiciones de los píxeles e id es el identificador del píxel. El parámetro *random* que se muestra en las ecuaciones, se utiliza para lograr el cambio de la distribución variable en cada iteración, logrando que los cambios entre las imágenes virtuales sean distintos, el cual puede tomar valores procedentes de dimensiones fractales o de otras funciones que cambian su valor conforme se realiza un ciclo de ejecución. Se considera la lista de fractales proporcionados en el anexo B para lograr los intercambios de valores entre el alfabeto que es empleado. El fractal utilizado es el llamado pentácopo (cuya ecuación está definida en el anexo B), debido a que arroja cuatro variantes de su dimensión fractal, las cuales son: 1.8617159595009178...34, 1.8617159595009176...94, 1.8617159595009182...16 y 1.8617159595009180...75.

$$\text{abs}(\text{x} \times \text{id} \times 2) - (\text{x} \times \text{y} \times \text{id}), \text{abs}((\text{y} \times \text{x})/\text{random}) \tag{31}$$

$$\text{abs}((\text{id} \times \text{y})(\text{id} \times \text{x})) \times -\text{random} \tag{32}$$

Los resultados obtenidos en la ejecución de las ecuaciones anteriores son ordenados de forma descendente para el canal R, de forma ascendente para el canal G y de forma descendente para el canal B. Con ello, se ordenan los píxeles y se reconstruye la imagen de portada, de igual forma, el orden ascendente o descendente se intercala entre estego-imagen.

Un ejemplo de la aplicación de las funciones de dispersión (31) y (32) se observa en la figura 24; (a) se visualiza a la imagen Lenna y (b) que representa a la imagen Peppers. Como se observa, la distribución de los píxeles es totalmente distinta a las imágenes originales, pero la relación de las posiciones originales sigue guardándose en la base de datos temporal, sin que ello afecte las modificaciones aplicadas sobre los píxeles.



(a) Lena (b) Peppers

Figura 24: Imagen virtual de (a) Lenna y (b) peppers

4.1.1 Resultados del escenario de Pruebas 1

Se obtiene la evaluación de cada una de las imágenes generadas (estego-imágenes) utilizando las métricas de calidad reportadas en el capítulo 2. Adicionalmente, se utiliza la herramienta StegExpose [23] para calcular la capacidad de evasión del algoritmo propuesto en cada escenario de pruebas.

La tabla 4 presenta los resultados obtenidos al evaluar las estego-imágenes generadas por VVRSM, en esta tabla se especifican las métricas de análisis de calidad como lo son PSNR, SNR, MSE, SSIM y CC, por otro lado, las últimas tres columnas representan los métodos de estegoanálisis empleados para el análisis de la imagen, entre ellos se encuentra la entropía, la chi-cuadrada y el análisis RS.

Tabla 4: Resultados obtenidos en estego-imágenes al aplicar el método VVRSM

Estego-imagen	Bytes incrustados	PSNR	SNR	MSE	SSIM	CC	Entropía portada / estego	x^2	RS
Lenna	40,000	50.881	44.956	0.533	0.999	0.999	5.101 / 6.422	0	0.292
Peppers	119,000	47.019	40.464	1.294	0.999	0.999	7.273 / 7.330	0	0.491
Barbara	130,000	47.695	41.217	1.108	0.998	0.999	7.520 / 7.523	0	0.493
Aeroplano	130,000	46.192	43.499	1.566	0.988	0.999	6.576 / 6.584	0	0.574

Tabla 4 Continuación: Resultados obtenidos en estego-imágenes al aplicar el método VVRSM

Estego-imagen	Bytes incrustados	PSNR	SNR	MSE	SSIM	CC	Entropía portada / estego	x^2	RS
Goldhill	130,000	46.193	39.793	1.565	0.995	0.999	7.506 / 7.503	0	0.543
Paisaje gran cañón	110,000	47.708	42.855	1.105	0.999	0.999	7.577 / 7.575	0	0.557
Frutas	130,000	46.249	38.097	1.545	0.996	0.999	7.513 / 7.552	0	0.553
Casa	110,000	47.381	44.047	1.189	0.997	0.999	6.265 / 6.542	0.7	0.663
Botes	130,000	46.259	40.814	1.541	0.992	0.999	7.101 / 7.160	0	0.596
Mandril	130,000	46.190	39.417	1.566	0.998	0.999	7.613 / 7.611	0	0.530
Paisaje marciano	130, 000	46.154	46.154	1.577	0.988	0.999	6.576 / 6.584	0	0.654
Paisaje Yosemite	110, 000	47.717	41.459	1.102	0.998	0.999	7.735 / 7.736	0	0.584

En los resultados obtenidos que se muestran en la tabla 4, se observa de forma general que los resultados obtenidos al aplicar el método VVRSM son satisfactorios dado que aprueban las métricas de calidad y más aún, porque todos ellos lograron pasar las pruebas de estegoanálisis. Se observa que en la mayoría de las estego-imágenes el valor del PSNR es satisfactorio, superior a 46 dB, indicando que no se aprecian distorsiones significativas en las estego-imágenes.

El SNR obtenido en las estego-imágenes oscila entre los 38 dB y 44 dB, indicando que el nivel de ruido es no es apreciable de forma visual y que las modificaciones efectuadas en las estego-imágenes no las distorsionan de forma apreciable.

El MSE obtenido en la mayor parte de las estego-imágenes oscila entre 0.500 y 1.580 puntos, lo cual valida que el resultado es aceptable, puesto que no indica zonas con una distorsión que sea visible, en las regiones de la imagen donde los colores y las texturas sean continuas, además que en ninguna estego-imagen se obtienen más 2 puntos de MSE, lo cual supone que las modificaciones no son apreciables visualmente.

En cuanto a la métrica SSIM, se observa que se mantuvo por encima de los 0.987 puntos, indicando que las estego-imágenes se generaron con distorsiones no apreciables con respecto a la imagen de portada.

El cálculo del CC arrojó resultados de 0.999 en todos los casos, mientras que el cálculo de la entropía presenta una variación menor al 1 % por cada imagen. Considerando que entre más se aproxime el valor obtenido de la entropía de la estego-imagen con el valor obtenido de la imagen de portada, se afirma que no hay alteraciones importantes en la estructura de la imagen.

Sin embargo, la estego-imagen *casa* cuya tasa de incrustación es cercana a los 110,000 bytes. En las columnas de estegoanálisis no se observan resultados satisfactorios, dado que, esta imagen presenta un desequilibrio entre el número de píxeles pares e impares contenidos en ella, indicando que es susceptible a ser detectada por la métrica de chi-cuadrada. No obstante, dado a los resultados obtenidos, se puede concluir que las pruebas aplicadas a las estego-imágenes son satisfactorias con respecto a los objetivos planteados en este trabajo de investigación.

De acuerdo con los datos obtenidos en las pruebas, se observa que las estego-imágenes *Lenna* y *casa* presentan mayor variación con respecto a las demás imágenes. Dado que, dichas imágenes tienen una mayor disparidad entre los píxeles pares e impares (tienen mayor cantidad de píxeles pares) con una relación de 30 % aproximadamente, por lo tanto, al momento de incrustar una determinada cantidad de datos, se desbalancea aún más la cantidad de píxeles pares e impares. Por consiguiente, al aplicar la métrica RS la tasa de detección crece en mayor medida. En este tipo de casos particulares se dificulta que se apruebe exitosamente el estegoanálisis con altas tasas de incrustación de datos.

Observando la columna de la métrica RS se corrobora que existe una detección de bits incrustados, pero al momento de analizar dichos bits, se determina una diferencia inferior de 66 % con respecto a los

bits realmente incrustados. Lo cual permite aventajar el método VVRSM cuando los datos incrustados son menores a 80,000 bytes (sin compresión).

Por otro lado, la recuperación de los datos incrustados, se obtienen al extraer las reglas de operación y aplicando el proceso inverso tanto de la dispersión de píxeles como de la sustracción de valores agregados a los coeficientes generados por la DWT. El mensaje codificado por el fractal se obtuvo satisfactoriamente sin pérdida de información.

Recordando que, los datos incrustados presentan compresión mediante la técnica LZW, lo cual reduce la cantidad de datos representados en un 50 % aproximadamente. En las pruebas realizadas se descartó incrustar archivos del tipo audio, ejecutable, vídeo e imagen, debido a que el proceso de compresión y codificación con base 64 permiten que el resultado final se exprese en cadenas de texto. Por lo tanto, se ha corroborado que es posible recuperar los datos incrustados, así como aprobar la mayor cantidad de métricas de calidad y pruebas de estegoanálisis.

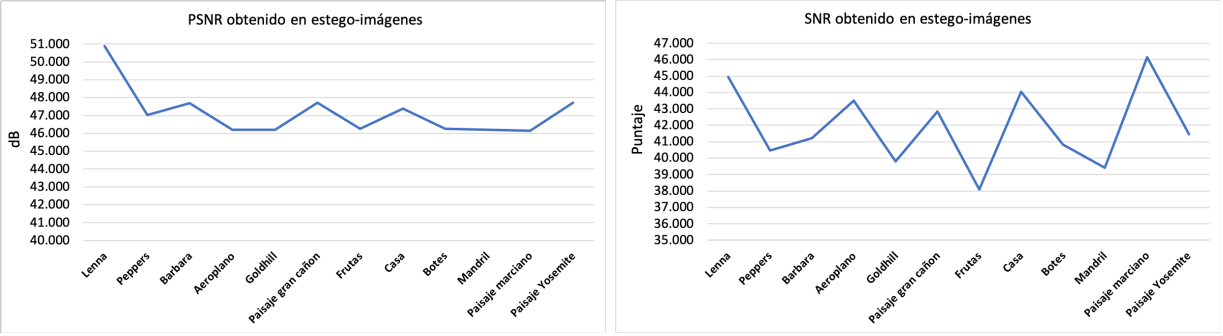
En la figura 25, se representan gráficamente los valores obtenidos de las métricas utilizadas. La figura 25 (a), observamos los valores obtenidos del PSNR en donde todas las estego-imágenes superan a los 40 dB, el valor mínimo aceptable para denotar alta calidad en una estego-imagen.

La figura 25 (b), se observan los valores del SNR, en donde hay una tendencia a superar los 40 dB, a excepción de la imagen *frutas* la cual está por debajo de este puntaje, sin embargo, las modificaciones no son perceptibles de manera general.

Por otro lado, la figura 25 (c) en la cual se muestran los resultados obtenidos para el MSE, se observa que la mayoría de las estego-imágenes superan el punto que se había propuesto como ideal (entorno a los dos puntos), además, cabe señalar que en la práctica no existen distorsiones apreciables.

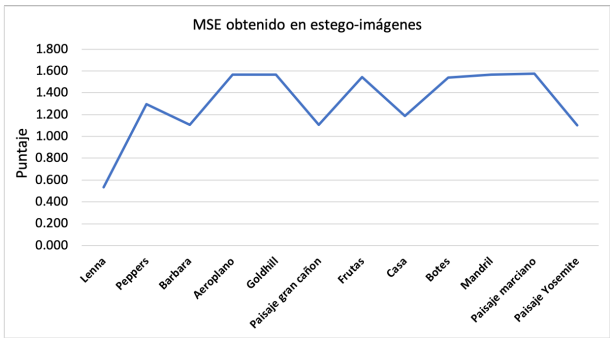
La figura 25 (d) representa la métrica SSIM en donde se observa una diferencia con respecto a lo ideal de un 0.050 % en la mayoría de las estego-imágenes, de forma similar esta tendencia se puede observar en los resultados de la gráfica 25 (e) de CC, aunque la variación máxima con respecto al ideal es de 0.005 %.

En la gráfica de la figura 25 (f) se compara la entropía de las imágenes de portada versus las estego-imágenes. En ésta, se observa que la imagen que presenta mayor variación es *Lena* y *casa*, por lo cual, los datos incrustados afectaron sensiblemente la distribución de las intensidades de los píxeles. Y con ello, las pruebas de estegoanálisis detectaron la existencia de datos incrustados. Por otro lado, en menor medida las estego-imágenes *peppers* y *botes* presentan una variación menor de su entropía con respecto a las imágenes de portada, pero siendo menor al 1 %, lo cual indica que no debe existir problemas al momento de ser sometidas a pruebas de estegoanálisis, al menos mediante chi-cuadrada.

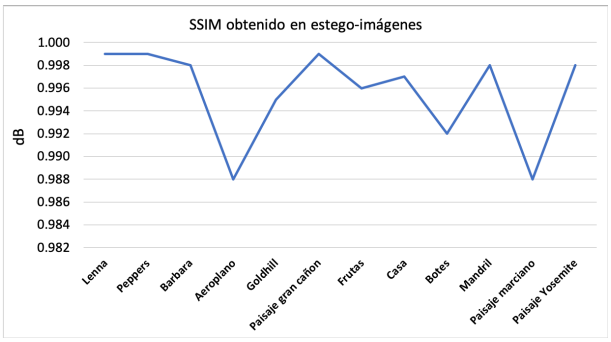


(a) PSNR (b) SNR

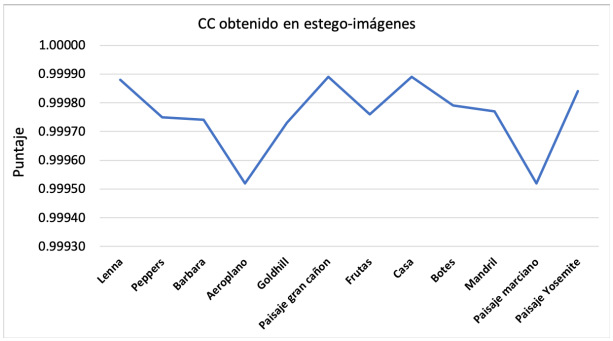
Figura 25: Resultados al aplicar las métricas de calidad, para el escenario de Pruebas 1



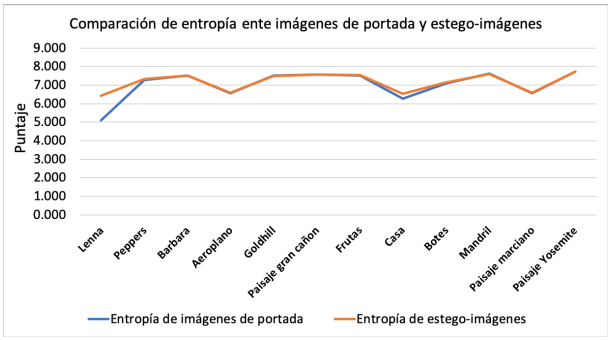
(c) MSE



(d) SSIM



(e) CC



(f) Entropía

Figura 25: Continuación. Resultados al aplicar las métricas de calidad, escenario de Pruebas 1

4.1.2 Resultados del escenario de Pruebas 2

En el escenario de pruebas VVRSM es comparado con LSB++ al incrustar texto en imágenes procedentes de los dataset Tiny y COCO, 1000 imágenes seleccionadas por cada dataset. En los resultados obtenidos del segundo escenario de pruebas se observa el comportamiento del método VVRSM en comparación con el método LSB++ al incrustar 5 bpp y 1 bpp, recordado que, LSB++ solo puede soportar una carga máxima de 1 bpp, respectivamente para cada método para los conjuntos Tiny y COCO, en las imágenes definidas con las tres diferentes resoluciones anteriormente propuestas. Los resultados se presentan a continuación.

En la tabla 5 se observa cuando la carga máxima ha sido de 5 bpp en VVRSM, el dataset Tiny muestra una distorsión menor que en el dataset COCO, tomando en cuenta los resultados obtenidos en PSNR, MSE y SSIM, esto posiblemente dado a que las imágenes del dataset Tiny presentan secciones con mayor cantidad de píxeles con valores similares lo cual permite obtener una diferencia de 0.97 dB de PSNR mayor al dataset COCO. Cuando se analizan los resultados del método LSB++ se observa la misma tendencia, Tiny presenta una diferencia a favor en PSNR de 0.1 dB, mientras que el resultado de MSE es una diferencia de 0.007, en SSIM y CC los resultados aparentemente son similares, estas diferencias, indican un desempeño marginal a favor del dataset Tiny.

En los resultados reportados de la tabla 5 se observa que, aunque la carga útil en el método VVRSM es cinco veces mayor que el método LSB++, los resultados de las métricas SSIM y CC tanto para Tiny como para COCO no remarcan diferencias. Si bien, los resultados obtenidos para el PSNR, SNR y MSE marcan diferencias, en las imágenes obtenidas por el método propuesto no presentan modificaciones estructurales, así como su luminosidad y su crominancia. La misma tendencia se puede observar en las tablas 6 y 7, donde las resoluciones de las estego-imágenes son más altas. Como se puede observar en los resultados de VVRSM, el PSNR se mantuvo por arriba de los 40 dB para las tres resoluciones de los conjuntos Tiny y COCO, por otra parte, oscilan entre los 34.6 dB y los 37.7 dB de SNR, siendo el valor más bajo para las estego-imágenes de la resolución 64×64 píxeles. En la tabla 5 se presenta un MSE con más de 5 puntos, por lo tanto, indica que

Tabla 5: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 64×64 píxeles aplicando el método de LSB++ y VVRSM

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
LSB++	Tiny	51.745	45.406	0.436	0.999	0.999	1.000
VVRSM	Tiny	**41.022**	**34.676**	**5.1433**	**0.992**	**0.998**	**5.000**
LSB++	COCO	51.665	45.409	0.443	0.999	0.999	1.000
VVRSM	COCO	**40.996**	**34.719**	**5.172**	**0.991**	**0.998**	**5.000**

por ser la resolución más baja, es el conjunto con mayor afectación, caso contrario, en los conjuntos con resoluciones 256×256 píxeles el MSE está por debajo de los 2.7 puntos, lo cual quiere decir que un cambio en una imagen de baja resolución es más perceptible, debido a que la continuidad de píxeles en un segmento determinado con un valor igual o similar cambia de forma más abrupta cuando la zona de representación se reduce.

Tabla 6: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 128×128 píxeles aplicando el método de LSB++ y VVRSM

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
LSB++	Tiny	51.820	45.481	0.429	0.999	0.999	1.000
VVRSM	Tiny	**42.797**	**46.453**	**3.417**	**0.992**	**0.999**	**5.000**
LSB++	COCO	51.474	45.284	0.463	0.998	0.999	1.000
VVRSM	COCO	**42.786**	**36.576**	**3.424**	**0.991**	**0.999**	**5.000**

Tabla 7: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 256×256 píxeles aplicando el método de LSB++ y VVRSM

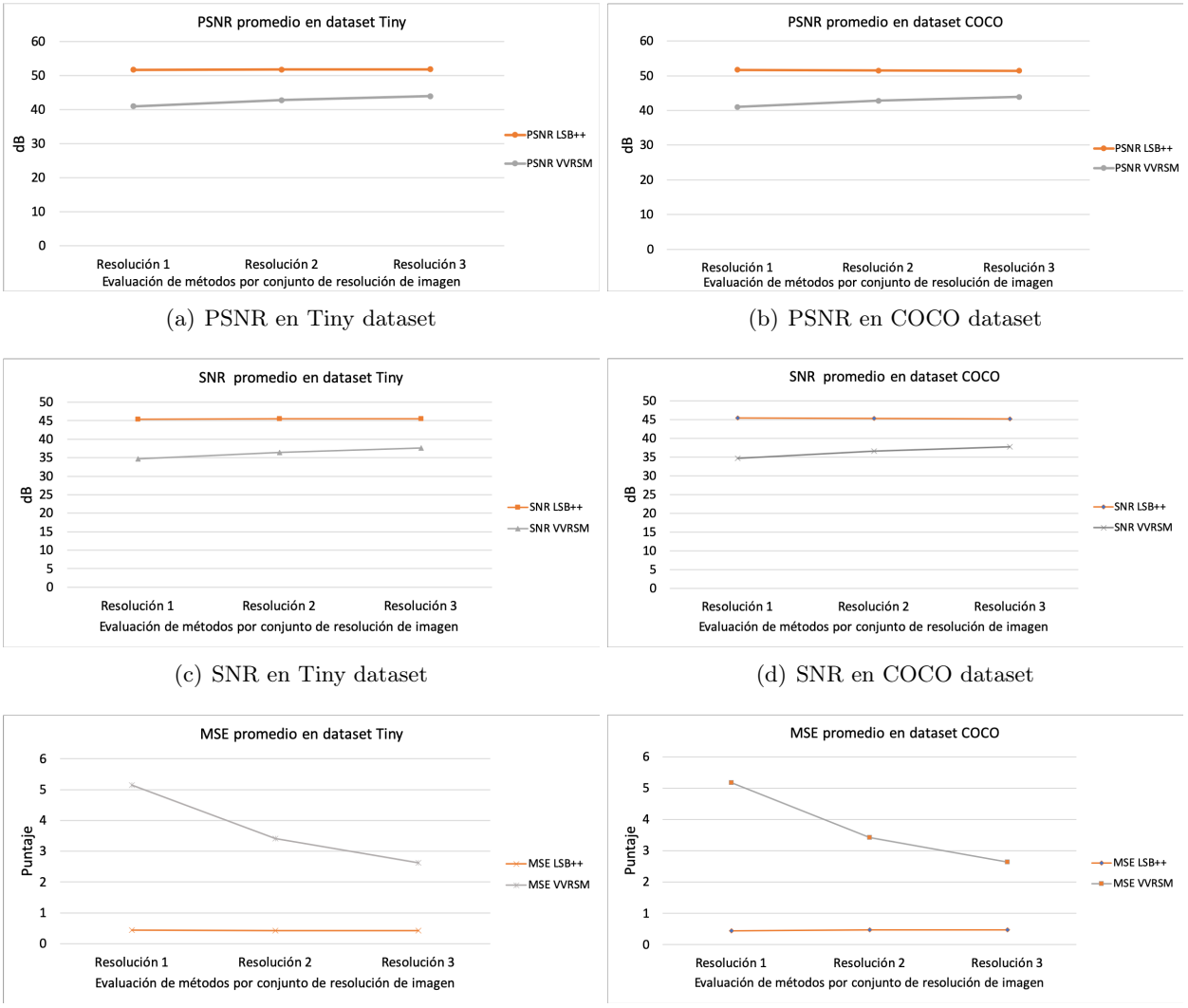
Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
LSB++	Tiny	51.872	45.892	0.428	0.999	0.999	1.000
VVRSM	Tiny	**43.93**	**37.586**	**2.631**	**0.993**	**0.999**	**5.000**
LSB++	COCO	51.433	45.351	0.468	0.998	0.999	1.000
VVRSM	COCO	**43.922**	**37.759**	**2.636**	**0.992**	**0.999**	**5.000**

Las métricas SSIM y CC para VVRSM se mantuvieron por arriba de los 0.990 puntos, lo cual indica que visualmente no se aprecian distorsiones significativas en las estego-imágenes para los conjuntos Tiny y COCO en las tres resoluciones propuestas. Mientras tanto en LSB++ los resultados se ubican en los parámetros idóneos por la carga de un 1 bpp, si se desea incrementar la carga de 1 bpp para LSB++, este método desprecia la carga que excede al valor anteriormente mencionado, por lo tanto, no es útil incrementarla.

EL método VVRSM permite la incrustación de texto con cargas mayores a 5 bpp, sin distorsión visual de las estego-imágenes y sin pérdida de datos, logrando con ello un método con la capacidad de altas cargas de datos, con la capacidad de aprobar distintas métricas de desempeño de calidad.

La figura 26 muestra gráficamente los resultados obtenidos de las tablas 5, 6 y 7. Al comparar ambos métodos, los resultados obtenidos por el método LSB++ al incrustar un bit por píxel, mantienen homogéneamente los valores del PSNR, SNR, MSE, SSIM y CC, para los conjuntos con resolución 1, 2 y 3. Por otro lado, los resultados mostrados por el método propuesto son ligeramente más bajos, sin embargo, las

cargas de incrustación son mayores, aproximadamente 5 veces más, lo cual perjudica en menor medida a los resultados obtenidos en las métricas de PSNR, SNR y MSE.



(a) PSNR en Tiny dataset

(b) PSNR en COCO dataset

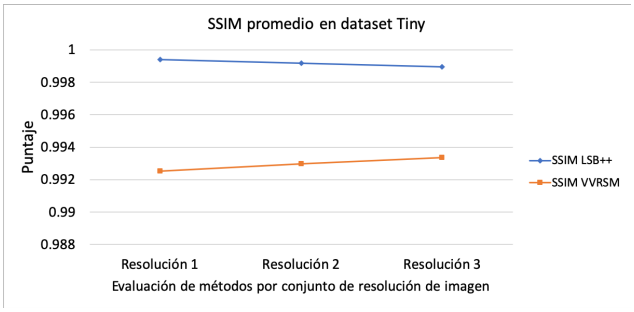
(c) SNR en Tiny dataset

(d) SNR en COCO dataset

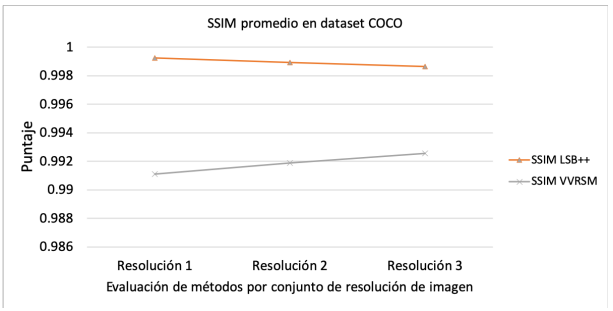
(e) MSE en Tiny dataset

(f) MSE en COCO dataset

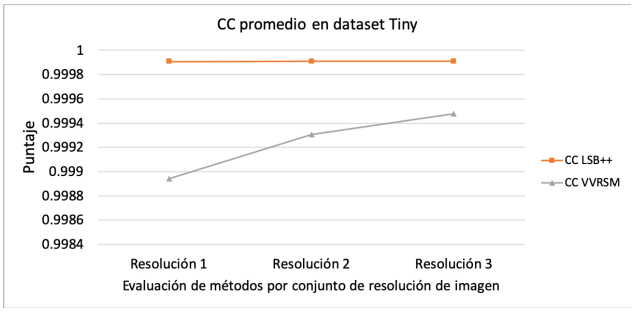
Figura 26: Resultados obtenidos al evaluar los métodos de LSB++ (Qazanfari) y el propuesto al embeber texto como mensaje en las resoluciones 1 (64×64), 2 (128×128) y 3 (256×256)



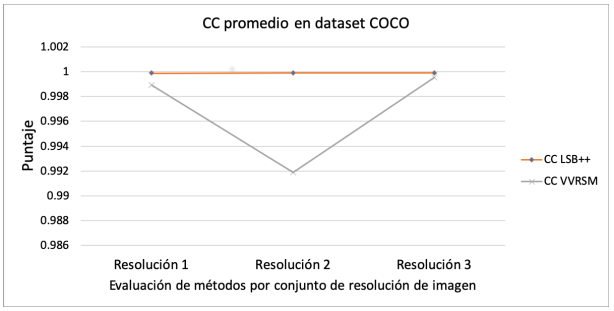
(g) SSIM en Tiny dataset



(h) SSIM en COCO dataset



(i) CC en Tiny dataset



(j) CC en COCO dataset

Figura 26: Continuación. Resultados obtenidos al evaluar los métodos de LSB++ (Qazanfari) y el propuesto al embeber texto como mensaje en las resoluciones 1 (64×64), 2 (128×128) y 3 (256×256)

Los datos reportados por Qazanfari y Safabakhsh (LSB++) [117] permiten evadir promedios de cargas de 0.8 bpp, sin embargo, analizando dichos datos se observa que, en su propuesta, sólo se puede incrustar un máximo de 1.0 bpp por imagen, de ahí que reporten resultados sobresalientes con respecto a las métricas PSNR y MSE, recordando que una salida de PSNR superior a 45 dB visualmente no es posible apreciar distorsiones, de igual forma los resultados de la métrica es menor 1.0 punto, por lo tanto la distorsión apreciada es técnicamente difícil de observar. Por otro lado, dichos resultados ante la evasión de técnicas de estegoanálisis se observa en la figura 26 que ambas propuestas obtienen resultados similares, recalcando que en nuestro método la carga propuesta es de 5.0 bpp.

Con respecto a los datos observados en las tablas 5, 6 y 7, se verifica que al incrustar valores de carga que van del 6 % al 26 % superiores a un 1 bpp en las imágenes de portada de los dataset Tiny y COCO, se obtendrían resultados similares o superiores a los registrados en LSB++ con 1 bpp, a continuación, se presentan los resultados obtenidos. En la tabla 8 se muestran los resultados al incrustar un 26 % más de bits en imágenes de 64×64 píxeles con respecto a LSB++, las métricas que muestran las diferencias con mayor notoriedad se encuentran en los resultados de PSNR y SNR, donde la diferencia a favor de VVRSM es de 1 y 2 dB para ambas métricas aproximadamente y la diferencia con respecto a MSE es de 0.1 a favor de VVRSM. El dataset mejor evaluado fue Tiny, ligeramente con respecto a COCO, con una diferencia marginal de 0.01 % con respeto a MSE y PSNR.

Para corroborar la dispersión de los datos con respecto a los promedios obtenidos de las métricas de calidad de las estego-imágenes obtenidas, se ha propuesto incluir la medición de la desviación estándar, y comprobar que lo resultados de forma general no se encuentren alejados en más de tres unidades para las métricas PSNR, SNR y MSE, de igual forma también se aplica en las métricas SSIM y CC. Al observar los resultados presentados en la tabla 9, se puede corroborar que la desviación estándar muestra que las variaciones entre los resultados de las métricas de calidad no presentan variaciones de más de un 1 % para PSNR en ambos métodos, mientras que en la medición de SNR es mayor con respecto a las demás métricas, pero es menor a 3 dB, aunque se puede ver que, en este caso, existe menor variación en la desviación estándar para los resultados de VVRSM al compararse con LSB++.

Tabla 8: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 64×64 píxeles aplicando el método de LSB++ y VVRSM con 1.260 bpp

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
LSB++	Tiny	51.745	45.406	0.436	0.999	0.999	1.000
VVRSM	Tiny	**53.034**	**46.706**	**0.323**	**0.999**	**0.999**	**1.260**
LSB++	COCO	51.665	45.409	0.443	0.999	0.999	1.000
VVRSM	COCO	**53.030**	**46.769**	**0.323**	**0.999**	**0.999**	**1.260**

Tabla 9: Resultados obtenidos en la evaluación de los dataset Tiny y COCO empleando la desviación estándar para las imágenes con resolución 64×64 píxeles aplicando el método de LSB++ y VVRSM con 1.260 bpp

Método	Dataset	PSNR σ	SNR σ	MSE σ	SSIM σ	CC σ	Carga en bpp
LSB++	Tiny	0.380	2.228	0.028	0.005×10^{-1}	0.007×10^{-2}	1.000
VVRSM	Tiny	**0.048**	**2.224**	**0.003**	$\mathbf{0.005} \times 10^{-1}$	$\mathbf{0.006} \times 10^{-2}$	1.260
LSB++	COCO	0.259	2.023	0.019	0.005×10^{-1}	0.005×10^{-2}	1.000
VVRSM	COCO	**0.047**	**2.001**	**0.003**	$\mathbf{0.004} \times 10^{-1}$	$\mathbf{0.005} \times 10^{-2}$	**1.260**

La tabla 10 presenta los resultados al incrustar 1.139 bpp en imágenes de 128×128 píxeles, y se observa una diferencia de 2 dB tanto para las métricas PSNR y SNR al comparar los métodos LSB++ y VVRSM, esto a favor del método propuesto. En el caso del MSE, VVRSM obtiene 0.2 puntos menos en comparación LSB++, e indica un menor grado de distorsión, por lo tanto, se puede decir que las estego-imágenes de VVRSM obtienen un mejor desempeño contra el método comparado. Con respecto a las métricas SSIM y CC, técnicamente son iguales, y se necesitaría de una precisión de 6 dígitos antes del punto decimal, siendo esta diferencia no significativa, por el reducido por la escala que se emplea de un punto.

Tabla 10: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 128×128 píxeles aplicando el método de LSB++ y VVRSM con 1.139 bpp

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
LSB++	Tiny	51.820	45.481	0.429	0.999	0.999	1.000
VVRSM	Tiny	**53.573**	**47.244**	**0.285**	**0.999**	**0.999**	**1.139**
LSB++	COCO	51.474	45.284	0.463	0.998	0.999	1.000
VVRSM	COCO	**53.572**	**47.377**	**0.285**	**0.999**	**0.999**	**1.139**

En los resultados de la tabla 11 se observa que se repite el comportamiento de resultados con respecto a la tabla 9, una desviación estándar que muestra variaciones por debajo de un1 dB para PSNR en ambos métodos, y en las demás métricas se observa que las diferencias son menores a décimas o centésimas para LSB++ y VVRSM respectivamente, para SSIM y CC baja a menos de milésimas para LSB++ y VVRSM, por lo tanto los resultados de forma general siguen cercanos al promedio obtenido.

En la tabla 12 se puede observar que las diferencias con respecto a las demás tablas se centran principalmente en las variaciones presentadas en PSNR y SNR, donde la diferencia es 3 dB para ambas métricas, una ganancia ligeramente mayor a las pruebas anteriores, esto es debido a que la carga incrustada es inferior (VVRSM) y que las imágenes tienen una mayor cantidad de píxeles, lo que permite mayor cantidad de modificaciones con menor repercusión en las modificaciones efectuadas a las estego-imágenes. En esta evaluación, las estego-imágenes del dataset Tiny obtuvo ligeramente un mejor desempeño con respecto a COCO, para

Tabla 11: Resultados obtenidos en la evaluación de sobre los dataset Tiny y COCO empleando la desviación estándar para las imágenes con resolución 128×128 píxeles aplicando el método de LSB++ y VVRSM con 1.139 bpp

Método	Dataset	PSNR σ	SNR σ	MSE σ	SSIM σ	CC σ	Carga en bpp
LSB++	Tiny	0.387	2.238	0.030	0.007×10^{-1}	0.008×10^{-2}	1.000
VVRSM	Tiny	**0.025**	**2.225**	**0.001**	$\mathbf{0.005}\times10^{-1}$	$\mathbf{0.006}\times10^{-1}$	**1.139**
LSB++	COCO	0.256	1.998	0.022	0.007×10^{-1}	0.006×10^{-2}	1.000
VVRSM	COCO	**0.026**	**1.975**	**0.001**	$\mathbf{0.005}\times10^{-1}$	$\mathbf{0.004}\times10^{-2}$	**1.139**

ambos métodos, posiblemente debido a que las imágenes tienen una mayor cantidad de píxeles consecutivos los cuales presentan valores que cambian en menor medida con relación a COCO.

Tabla 12: Resultados obtenidos en la evaluación de sobre los dataset Tiny y COCO para las imágenes con resolución 256×256 píxeles aplicando el método de LSB++ y VVRSM con 1.060 bpp

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
LSB++	Tiny	51.872	45.892	0.428	0.999	0.999	1.000
VVRSM	Tiny	**54.420**	**48.091**	**0.234**	**0.999**	**0.999**	**1.060**
LSB++	COCO	51.433	45.351	0.468	0.998	0.999	1.000
VVRSM	COCO	**54.418**	**48.270**	**0.235**	**0.999**	**0.999**	**1.060**

Los resultados de la desviación estándar de los conjuntos Tiny y COCO con resolución de 256×256 píxeles, se presenta en la tabla 13, al aplicar los métodos LSB++ y VVRSM, presentan el mismo patrón que las tablas 9 y 11 donde se registraron los datos obtenidos de esta medida, el resultado con mayor variación se encuentra en SNR para ambos métodos, y los resultados de PSNR, MSE, SSIM y CC, en el orden señalado presentar un valor escalar que va de décimas a diezmilésimas, lo cual indica que, presentan técnicamente los mismos valores según las métricas de calidad para las estego-imágenes de ambos conjuntos, y con mejor desempeño para VVRSM, en las tres resoluciones analizadas (64×64, 128×128 y 256×256), donde, por ejemplo la desviación estándar del PSNR está en 0.0135 dB. Estos resultados son debido a que como máximo se está modificando el primer bit de cada píxel.

En las métricas donde se registró menos variación, cuando se aplicó la desviación estándar fue en CC y SSIM, esto refuerza por una parte que VVRSM es capaz de alcanzar mejores resultados en la calidad de las estego-imágenes y la estabilidad de método LSB++ al incrustar un 1 bpp de datos para las estego-imágenes.

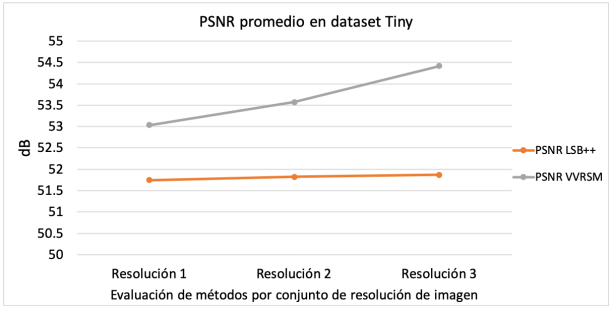
Tabla 13: Resultados obtenidos en la evaluación de los dataset Tiny y COCO empleando la desviación estándar para las imágenes con resolución 256×256 píxeles aplicando el método de LSB++ y VVRSM con 1.060 bpp

Método	Dataset	PSNR σ	SNR σ	MSE σ	SSIM σ	CC σ	Carga en bpp
LSB++	Tiny	0.389	2.239	0.029	0.008×10^{-1}	0.007×10^{-2}	1.000
VVRSM	Tiny	**0.014**	**2.225**	$\mathbf{0.007}\times10^{-1}$	$\mathbf{0.005}\times10^{-1}$	$\mathbf{0.005}\times10^{-1}$	**1.060**
LSB++	COCO	0.255	1.979	0.022	0.008×10^{-1}	0.005×10^{-2}	1.000
VVRSM	COCO	**0.013**	**1.957**	$\mathbf{0.007}\times10^{-1}$	$\mathbf{0.005}\times10^{-1}$	$\mathbf{0.003}\times10^{-2}$	**1.060**

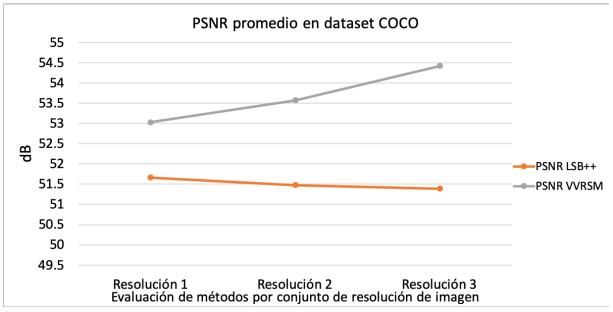
De acuerdo con los resultados anteriormente mostrados, entre LSB++ y VVRSM las estego-imágenes estuvieron por arriba de los 51 dB, este es un rango donde visualmente no se detectan distorsiones y de igual

forma las métricas SNR, MSE, SSIM y CC presentan valores que están arriba de los valores considerados como ideales, aunque en las mediciones de la desviación estándar los resultados para las métricas de calidad generalmente están más cerca de 0 que los resultados de LSB++, sobre todo en la evaluación de PSNR, mientras que los valores del SNR son los más elevados para ambos métodos, con relación a las demás métricas en relación a la desviación estándar, pero manteniéndose por debajo de los 3 dB.

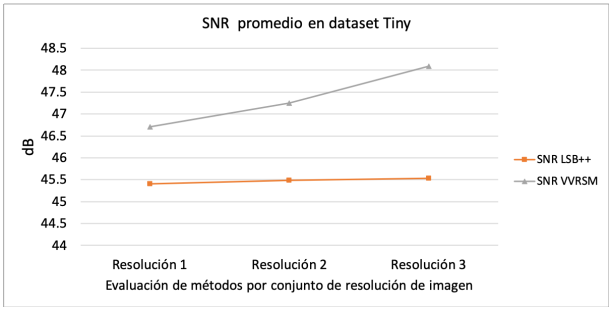
En la figura 27 se presentan los resultados graficados de las tablas de métricas de desempeño cuando se compara LSB++ versus VVRSM al incrustar 1 bpp, en donde se puede observar de mejor forma las diferencias que existen entre el método LSB++ y VVRSM, con respecto a esto las diferencias más notables están en los resultados del PSNR y SNR para ambos dataset, en donde la diferencia oscila entre los 3 dB y los 2 dB, dependiendo el conjunto de imágenes analizadas, lo cual representa una diferencia logarítmica multiplicada por 10 por cada unidad de diferencia que se presenta en los resultados de PSNR y SNR, debido a que una diferencia en dB no se puede medir de forma lineal.



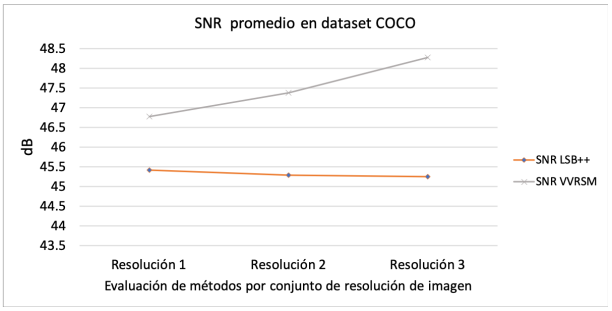
(a) PSNR en dataset Tiny



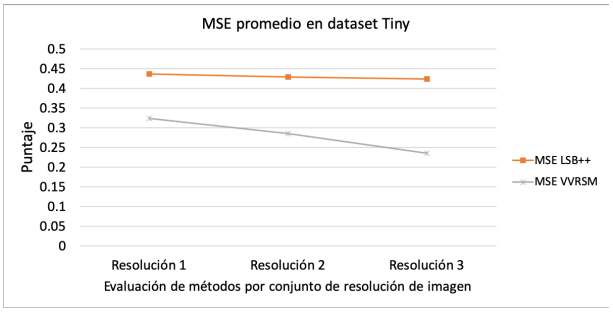
(b) PSNR en dataset COCO



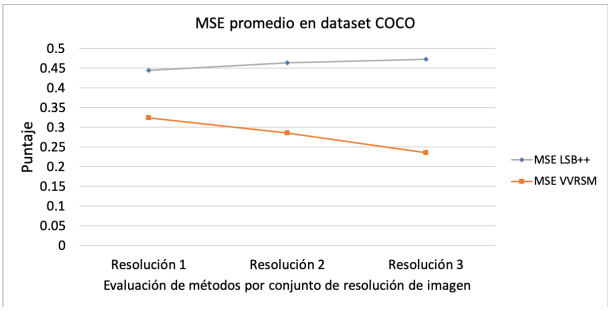
(c) SNR en dataset Tiny



(d) SNR en dataset COCO

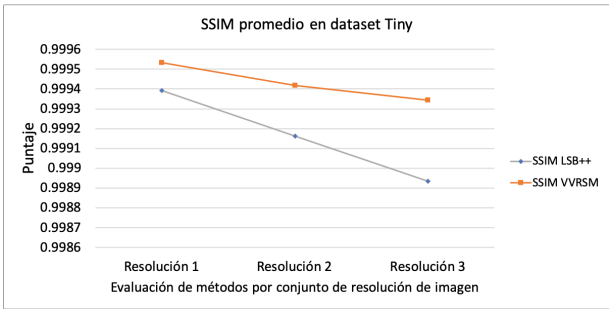


(e) MSE en dataset Tiny

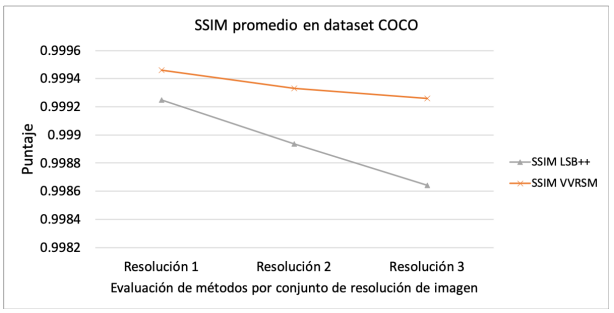


(f) MSE en dataset COCO

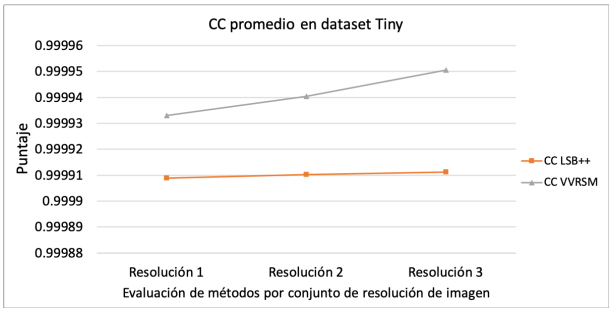
Figura 27: Resultados obtenidos al evaluar los métodos de LSB++ (Qazanfari) y VVRSM al embeber texto como mensaje en las resoluciones 1 (64×64), 2 (128×128) y 3 (256×256)



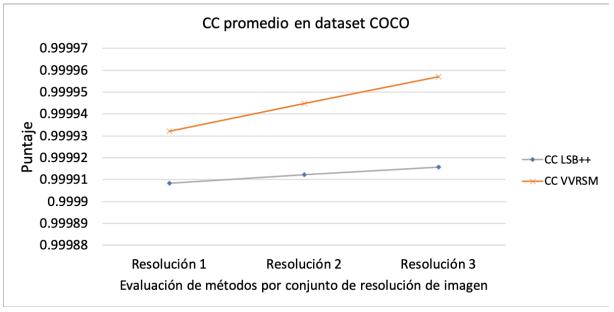
(g) SSIM en dataset Tiny



(h) SSIM en dataset COCO



(i) CC en dataset Tiny



(j) CC en dataset COCO

Figura 27: Continuación. Resultados obtenidos al evaluar los métodos de LSB++ (Qazanfari) y VVRSM al embeber texto como mensaje en las resoluciones 1 (64×64), 2 (128×128) y 3 (256×256)

Los resultados que se aplicaron al incrustar tasas superiores a 1 bpp corroboran que el método VVRSM tiene un mejor desempeño con respecto a LSB++ en el apartado visual, como se corroboró al aplicar las métricas de calidad PSNR, SNR, MSE y SSIM, aunque la diferencia son cercanas al 5 % de forma general para las métricas evaluadas, si es apreciable, sobre todo en las métricas cuyos resultados están en dB (PSNR y SNR, con variaciones cercanas a 2 dB) y en el MSE. Con respecto a CC las estego-imágenes están altamente correlacionadas con respecto a las imágenes portada. Estos resultados corroboran VVRSM es competente contra métodos dedicados a obtener un desempeño sobresaliente en métricas de calidad y en evasión de estegoanálisis como se valorará en la sección 5.2.

De forma general, el método VVRSM es competitivo desde la perspectiva visual, ya que las estego-imágenes no muestran distorsiones aparentes y esto es respaldado por los resultados presentados en las métricas PSNR, SNR, MSE, SSIM y CC, además, es efectivo cuando los promedios de bpp son cercanos a 1.7 bpp versus los 0.8 bpp que tienen como límite la técnica LSB++ para lograr la evasión de estegoanálisis.

4.1.3 Resultados del escenario de Pruebas 3

Para la tercera evaluación, se hizo una selección de imágenes aleatorias tanto del dataset Tiny (10,000 imágenes disponibles) como del dataset COCO (36,000 imágenes disponibles), para ambos conjuntos de datos se generó un conjunto A de 1000 imágenes para incrustar en otro conjunto llamado B de 1000 imágenes de portada, todas las imágenes son del tipo RGB.

De las imágenes seleccionadas, se generaron dos subconjuntos con dimensiones de 64×64 píxeles (resolución 1) para cada conjunto de datos, 2 subconjuntos con dimensiones de 128×128 píxeles (resolución 2) para cada conjunto de datos y 2 subconjuntos más para cada subconjunto de 256×256 píxeles (resolución 3).

Resumiendo:

A contiene 1000 imágenes para incrustar,
B contiene 1000 imágenes de portada,

2 subconjuntos de 64×64 provenientes de A y 2 subconjuntos de 64×64 provenientes de B,
2 subconjuntos de 128×128 provenientes de A y 2 subconjuntos de 128×128 provenientes de B y
2 subconjuntos de 256×256 provenientes de A y 2 subconjuntos de 256×256 provenientes de B

Los bits promedio de cada imagen incrustada con formato JPEG en cada conjunto de datos tiene aproximadamente las siguientes cargas expresadas en la tabla 14.

Tabla 14: Carga incrustada para fase 3 de pruebas

Dataset	Resolución	Carga incrustada en bits
Tiny	64×64	15,603
COCO	64×64	15,606
Tiny	128×128	57,385
COCO	128×128	41,098
Tiny	256×256	141,100
COCO	256×256	152,674

Para la tercera etapa de pruebas se utilizó el método de Baluja dado a que es un método ampliamente avanzado el cual emplea un mecanismo basado en Deep Learning con la capacidad de incrustar el mensaje en distintas zonas de la imagen para pasar pruebas por estegoanálisis.

La figura 28 muestra tres extractos de la evaluación del método de Baluja a las imágenes de prueba. La figura 28 (a) representan la imagen de portada *Bus* extraída de Tiny con dimensión de 64×64 píxeles. La figura 28 (b) es una estego-imagen obtenida por el método Baluja, en donde se visualiza que hubo un cambio de tonalidad altamente visible, de color verde. Finalmente, en la figura 28 (c) se visualiza la estego-imagen obtenida por el método VVRSM, comparando este resultado con el de Baluja, se observan menores cambios de tonalidades con base a la imagen de portada. El mismo comportamiento se observa en las imágenes de la figura 28 (e) y (h) (Cebra y Edificios, respectivamente) obtenidas por el método de Baluja versus las imágenes de la figura 28 (f) e (i) obtenidas con el método VVRSM.

(a) Imagen de portada Bus 64×64 (b) Estego-imagen 64×64 del método de Baluja (c) Estego-imagen Bus 64×64 VVRSM

Figura 28: Imágenes de portada (incisos (a), (d) y (g)) y estego-imágenes resultantes de aplicar los métodos de Baluja y VVRSM para Tiny con Resolución 1 (64×64), COCO con Resolución 2 (128×128) y COCO Resolución 3 (256×256)

(d) Imagen de portada Cebra 128×128

(e) Estego-imagen 128×128 del método de Baluja

(f) Estego-imagen Cebra 128×128 VVRSM

(g) Imagen de portada edificios 256×256

(h) Estego-imagen edificios 256×256 del método de Baluja

(i) Estego-imagen edificios 256×256 de VVRSM

Figura 28: Continuación. Imágenes de portada (incisos (a), (d) y (g)) y estego-imágenes resultantes de aplicar los métodos de Baluja y VVRSM para Tiny con Resolución 1 (64×64), COCO con Resolución 2 (128×128) y COCO Resolución 3 (256×256)

El método de Baluja [12] presenta una interesante propuesta para evadir las técnicas de estegoanálisis, concentrándose en la métrica de la chi-cuadrada, Baluja utiliza la convolución como la técnica principal de incrustación de datos, el método admite una imagen de dimensiones iguales con respecto a la imagen de portada, y una red neuronal que emplea Deep Learning para calcular la distribución de los datos a incrustar, lo cual le permite con distintos dataset, a pesar de que no se haya entrenado con ellos, volviéndolo un método versátil. Como en la propuesta de Baluja las imágenes a incrustar sufren una perdida bits, debido a que toman los 5 bits más significativos de cada imagen de portada, al recuperarse la información esta se presenta con pérdidas.

De acuerdo con los resultados presentados en la tabla 15, el método Baluja completa correctamente la inserción de datos en las imágenes de pruebas, sin embargo, se observa que en la evaluación de las métricas de PSNR y SNR los valores obtenidos son bajos (están por debajo de 30 dB para PSNR y de 25 dB para SNR) con respecto a los obtenidos por el VVRSM. Además, la métrica MSE está muy por encima de los 5 puntos (lo cual ya es un indicio de un error considerable en la imagen), y se corrobora en las imágenes de la figura 28, incisos (b), (e) y (h), pero mantiene estructuralmente su forma. Resulta interesante que los datos presentados en la medición de SSIM y CC obtienen un alto rendimiento e indica que la incrustación del mensaje tendrá mayor éxito al pasar una prueba de estegoanálisis.

Tabla 15: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 64×64 píxeles aplicando el método de Baluja y el método VVRSM

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
Baluja	Tiny	27.065	20.531	184.270	0.947	0.989	1.269
VVRSM	Tiny	**47.489**	**41.499**	**1.213**	**0.997**	**0.999**	**1.269**
Baluja	COCO	27.118	20.496	182.349	**0.994**	0.989	1.270
VVRSM	COCO	**43.358**	**37.190**	**3.281**	**0.994**	**0.999**	**1.270**

Cabe mencionar que el método VVRSM mantiene un equilibrio entre los resultados obtenidos de las métricas PSNR, SNR y MSE, ya que mantienen una estabilidad en los cambios de valores de píxeles, lo cual no sucede con el método de Baluja. Además, los resultados obtenidos para las métricas SSIM y CC son bastante similares, aunque en general es posible apreciar que numéricamente en las métricas de PSNR, SNR y SSIM presentan diferencias notorias en cuanto al método de Baluja.

En la tabla 16 se presentan los resultados de aplicar la desviación estándar a los resultados de las métricas de calidad para los conjuntos Tiny y COCO. Como se puede observar para el método de Baluja la mayor diferencia se encuentra en MSE, y es el resultado de desviación estándar más notorio para ambos conjuntos de imágenes, mientras que en SSIM y CC las diferencias varían por centésimas. En el caso de PSNR y SNR es menor a 4 dB. Para el método VVRSM, las variaciones en la desviación estándar son menores que en Baluja por 2 dB para PSNR y SNR, mientras en SSIM y CC los resultados se expresan en milésimas, esto indica que la fluctuación de cambios de valores entre estego-imágenes es menor.

Tabla 16: Resultados obtenidos en la evaluación de los dataset Tiny y COCO al obtener la desviación estándar para las imágenes con resolución 64×64 píxeles aplicando aplicando el método de Baluja y el método VVRSM

Método	Dataset	PSNR σ	SNR σ	MSE σ	SSIM σ	CC σ	Carga en bpp
Baluja	Tiny	3.704	3.783	187.777	0.053	0.014	1.269
VVRSM	Tiny	**1.886**	**2.127**	**0.222**	**0.007**	**0.001**	**1.269**
Baluja	COCO	3.600	3.577	176.794	$\mathbf{0.437}\times10^{-1}$	0.207×10^{-1}	1.270
VVRSM	COCO	**1.851**	**2.770**	**1.419**	$\mathbf{0.005}\times10^{-1}$	**0.001**	**1.270**

Por otro lado, cuando se analizan los resultados de la tabla 17, para imágenes de resolución 128×128 píxeles con el método de Baluja, se presenta la misma tendencia que en el análisis anterior, se observa un mejor rendimiento cuando se obtiene el resultado del MSE para el dataset COCO. El método VVRSM obtiene mejores resultados para ambos conjuntos de datos, especialmente para el dataset COCO. De acuerdo con los resultados obtenidos en la tabla 19 en el dataset COCO se obtienen resultados de las métricas de evaluación con mejor desempeño con respecto al dataset Tiny, debido a que la representación de datos para VVRSM fue más efectiva y puesto que la mayor resolución de las imágenes de portada permite que la variación entre píxeles sea menos abrupta entre las regiones de píxeles, esto redunda en las alteraciones son en menor medida apreciables a nivel visual y que estadísticamente su comportamiento sea menos notorio cuando se modifican los valores de los píxeles.

Tabla 17: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 128×128 píxeles aplicando el método de Baluja y el método VVRSM

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
Baluja	Tiny	26.882	20.380	185.600	0.926	0.989	1.159
VVRSM	Tiny	**40.959**	**34.970**	**5.794**	**0.992**	**0.998**	**1.159**
Baluja	COCO	27.801	21.254	139.788	0.923	0.989	0.834
VVRSM	COCO	**50.060**	**43.947**	**0.652**	**0.997**	**0.999**	**0.834**

En la tabla 18, para las resoluciones de 128×128 píxeles, nuevamente VVRSM presenta menor fluctuación con respecto a los resultados de la desviación estándar en las métricas de calidad con respecto al método de Baluja, siendo más notorio en el conjunto COCO, y la métrica MSE la que presenta puntajes superiores a 121 puntos. El que VVRSM presente menor cantidad de cambios en las evaluaciones por desviación estándar y es debido a que las estego-imágenes sufrieron menos cambios con respecto a las imágenes de portada.

Tabla 18: Resultados obtenidos en la evaluación de los dataset Tiny y COCO al obtener la desviación estándar para las imágenes con resolución 128×128 píxeles aplicando el método de Baluja y el método VVRSM

Método	Dataset	PSNR σ	SNR σ	MSE σ	SSIM σ	CC σ	Carga en bpp
Baluja	Tiny	3.464	3.381	184.261	0.051	0.014	1.159
VVRSM	Tiny	**0.171**	**2.138**	**0.133**	**0.006**	**0.006**$\times 10^{-1}$	**1.159**
Baluja	COCO	3.064	3.434	121.087	0.045	0.010	0.834
VVRSM	COCO	**0.572**	**2.085**	**0.078**	**0.009**$\times 10^{-1}$	**0.001**$\times 10^{-1}$	**0.834**

La tabla 19 presenta los resultados de la evaluación de los conjuntos Tiny y COCO con resoluciones de 256×256 píxeles, como se puede observar el método de Baluja sigue con la tendencia de valores por debajo de los 30 dB para PSNR y SNR en ambas evaluaciones, con un MSE superior a los 150 puntos, mientras que SSIM y CC se ubican por arriba de los 0.91 puntos, sobre todo en CC donde los resultados se encuentran por arriba de los 0.99 puntos. Por otra parte VVRSM presenta un mejor desempeño en Tiny con respecto a COCO, lo cual supone un mejor compresión de datos, esto es más notorio en los 3 dB de diferencia que existen entre las mediciones de PSNR y SNR, mientras que la diferencia con los resultados de MSE es de 1.6 puntos, lo cual indica una diferencia notable, y esto indica una menor afectación de por la incrustación de datos en las estego-imágenes, por otra parte, las imágenes de procedentes de COCO son en menor medida afectadas por el reescalamiento efectuado, debido a que al ser imágenes con resoluciones superiores a 600x400 píxeles, al momento de ser sometidas a una reducción de 256×256 píxeles pierden menos características visuales con respecto al dataset Tiny, donde las imágenes originales son de 72×72 píxeles.

Tabla 19: Resultados obtenidos en la evaluación de los dataset Tiny y COCO para las imágenes con resolución 256×256 píxeles aplicando el método de Baluja y el método VVRSM

Método	Dataset	PSNR promedio	SNR promedio	MSE promedio	SSIM promedio	CC promedio	Carga en bpp
Baluja	Tiny	27.437	20.924	168.379	0.917	0.990	0.712
VVRSM	Tiny	**43.870**	**37.881**	**2.938**	**0.994**	**0.999**	**0.712**
Baluja	COCO	27.447	20.807	158.374	0.915	0.992	0.776
VVRSM	COCO	**47.077**	**40.905**	**1.393**	**0.995**	**0.999**	**0.776**

En la tabla 20 se encuentra un comportamiento similar como en las tablas 16 y 18, con una desviación estándar bastante notoria en MSE por el elevado valor para el método de Baluja, con 179 y 135 para los

dataset Tiny y COCO, mientras VVRSM sigue presentando menos variaciones en los resultados de cada métrica, aunque los resultados para SNR son superiores a un 30 % a lo alcanzado en las demás pruebas con resoluciones inferiores, pero en las demás métricas los cambios son menos notorios.

Tabla 20: Resultados obtenidos en la evaluación de los dataset Tiny y COCO al obtener la desviación estándar para las imágenes con resolución 256×256 píxeles aplicando el método de Baluja y el método VVRSM

Método	Dataset	PSNR σ	SNR σ	MSE σ	SSIM σ	CC σ	Carga en bpp
Baluja	Tiny	3.595	3.535	179.123	0.057	0.0143	0.712
VVRSM	Tiny	**0.148**	**2.151**	**0.088**	**0.005**	$\mathbf{0.004}\times10^{-1}$	**0.712**
Baluja	COCO	3.430	3.569	135.330	0.049	0.009	0.776
VVRSM	COCO	**1.923**	**2.832**	**0.584**	$\mathbf{0.084}\times10^{-1}$	**0.001**	**0.776**

Los puntajes de las métricas PSNR, SNR, MSE, SSIM y CC se mantuvieron cercanos a los niveles idóneos (45 dB, 35 dB, 3, 0.990, 0.990 respectivamente) para VVRSM, mientras que, como se ha había mencionado anteriormente para Baluja solo el SSIM y CC están cercanos al valor idóneo, aunque de forma general al analizar la desviación estándar en las resoluciones de 256×256 píxeles para VVRSM es donde se observan las diferencias más notorias con respecto a los pruebas anteriores, donde las el PSNR, por tomar una referencia tiene 1.923 dB de desviación estándar, mientras que en el conjunto Tiny es de 0.148 dB, lo indica que existe mayor fluctuación de valores para este conjunto, lo mismo se puede observar en las demás métricas para este caso particular.

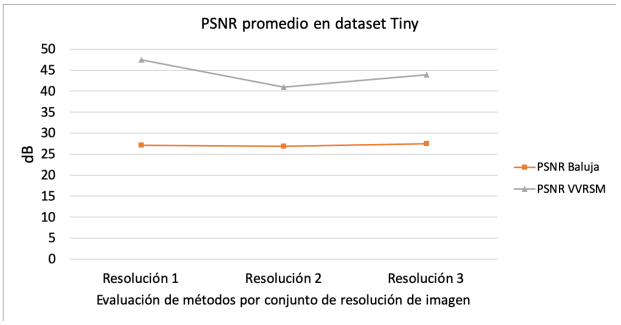
Es importante señalar que las imágenes de prueba del dataset Tiny han sido reescaladas de grupos de imágenes de baja resolución (72×72 píxeles), las cuales presentan una mayor distorsión al escalarlas a 128×128 o 256×256 píxeles, dado que la pérdida de datos es mayor. Sin embargo, las pruebas realizadas no indican variaciones abruptas entre la tendencia de los resultados que se han presentado en las evaluaciones, con la excepción de la métrica de MSE (debido a que se obtienen resultados superiores a 100 puntos, e indicaría que la imagen resultante estaría totalmente distorsionada, lo cual no es cierto del todo), en la cual es posible observar un cambio notorio en el promedio de los resultados del método Baluja. Una ventaja que presenta VVRSM referente al método de Baluja, es que, en VVRSM no existe perdida de datos, aunque por el proceso de la máscara DWT un porcentaje menor al 0.00001 % de todos los píxeles disponibles puede ser modificado de forma irreversible, mientras que, en el método de Baluja se observa una perdida al recuperar el mensaje incrustado. Esta pérdida de datos en el método de Baluja es debido a que el autor propone una representación de las imágenes que son incrustadas en las estego-imágenes mediante los 5 bits de mayor peso por cada píxel, esto se puede ver en las estego-imágenes bus, cebra y edificios con una tonalidad verde que se presenta a lo largo de las imágenes.

Una de las observaciones con respecto a la implementación de Deep Learning para esteganografía por parte del método de Baluja es que al evaluar un amplio conjunto de datos, el consumo de memoria RAM se dispara, sobre todo cuando las resoluciones son mayores a 64×64 píxeles, requiriendo una disponibilidad superior a 16 GB de memoria RAM, cuando se trabaja con CPU (para evitar el uso de memoria en disco), este método puede ser procesado por medio de GPU para acortar el tiempo de ejecución.

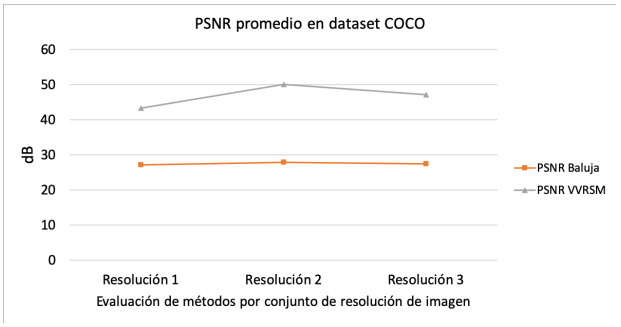
En la figura 29 se observan los resultados gráficos obtenidos al analizar las estego-imágenes con respecto a las imágenes de portada, tanto para el método de Baluja como para el método VVRSM. Es notorio que los resultados de PSNR y SNR para Baluja son inferiores a los obtenidos en VVRSM, tanto para el dataset Tiny como para el dataset COCO, estos resultados se presenta en las figuras con incisos ((a), (b), (c) y (d)). Es notable que las diferencias en dB para los conjuntos evaluados, así como en las resoluciones propuestas existe una diferencia que supera los 12 dB, esta diferencia es a favor de VVRSM, y como bien ha se descrito en Baluja está por debajo de los 30 dB, mientras que en VVRSM está por arriba de los 40 dB.

Los resultados obtenidos en MSE ((e), (f)) en Baluja indican que la imagen sufrió alteraciones notorias

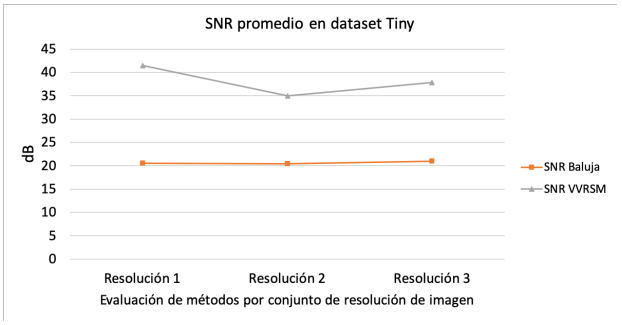
debido a que los puntajes para todos los casos son superiores a 140 puntos. Por otro lado, los resultados en SSIM ((g) y (h)) y CC ((i) y (j)) indican que, si bien la imagen ha resultado en alteraciones, la composición estructural de la estego-imagen se ha mantenido. Cuando las estego-imágenes se analizan mediante StegExpose denota otra tendencia, las estego-imágenes de Baluja tienen una tendencia de aproximadamente 50 % de ser detectado cuando se trabaja con estego-imágenes que pertenecen al subconjunto de la resolución 1, mientras se trabaja con el subconjunto según la resolución 1 y la resolución 2, tiene una probabilidad de evadir el estegoanálisis con una probabilidad del 90 %, donde nuestro método no tiene perdida de información, puesto que es posible recuperar el mensaje, sin alterar su estructura original. Los resultados anteriormente mencionados son respaldados por SSIM y CC, con esto podemos deducir que las métricas PSNR, SNR y MSE no son determinantes para determinar que un sistema de estenografía genera imágenes no detectables, aunque, cuando los valores son elevadamente bajos como en los mostrado en PSNR y SNR, o demasiado altos, como en MSE, es evidente que una imagen mostrará una composición lo suficientemente apreciable si se le compara contra la imagen de portada. En los gráficos se aprecia una superioridad notable en las métricas PSNR, SNR y MSE, a favor de VVRSM.



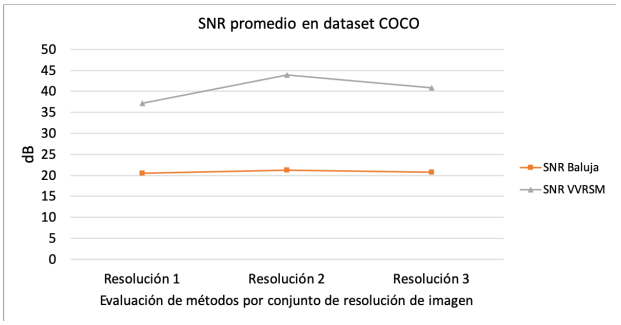
(a) PSNR en Tiny dataset



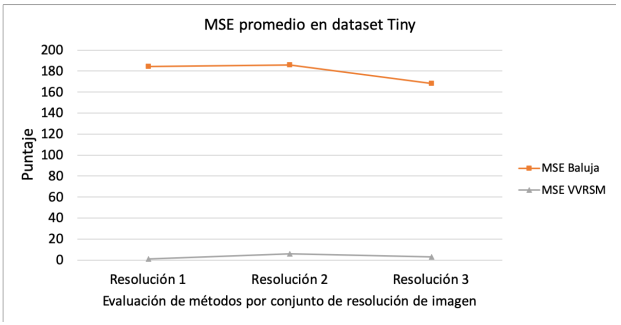
(b) PSNR en COCO dataset



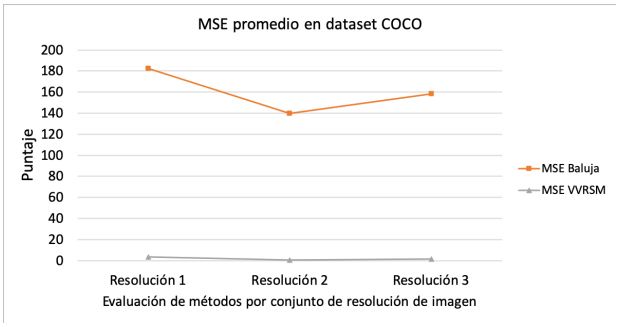
(c) SNR en Tiny dataset



(d) SNR en COCO dataset

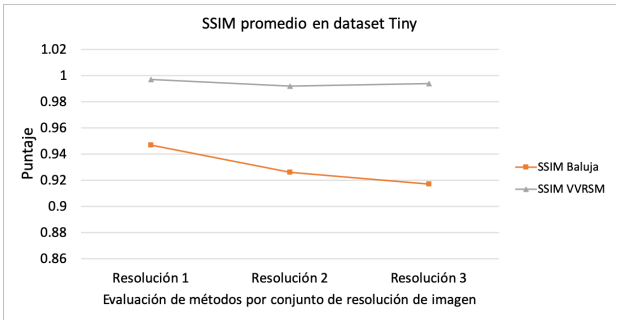


(e) MSE en Tiny dataset

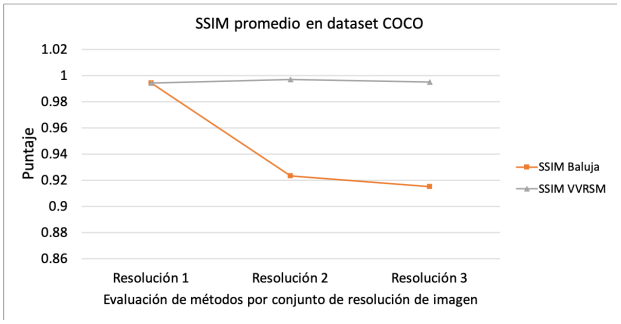


(f) MSE en COCO dataset

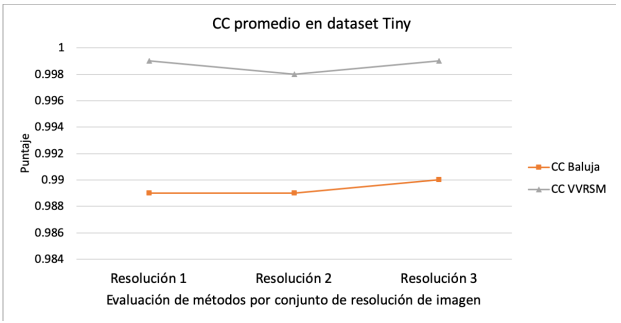
Figura 29: Resultados obtenidos de la evaluación entre método de Baluja y VVRSM al embeber imágenes en los subconjuntos de Resolución 1 (64×64), 2 (128×128) y 3 (256×256)



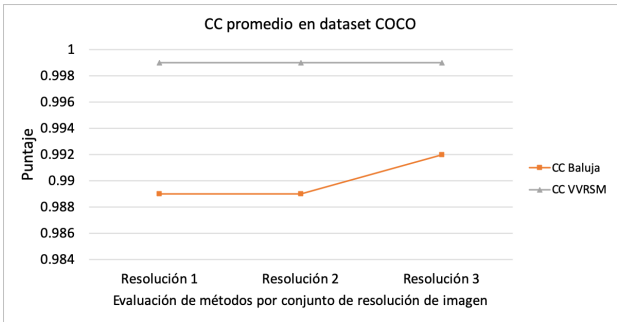
(g) SSIM en Tiny dataset



(h) SSIM en COCO dataset



(i) CC en Tiny dataset



(j) CC en COCO dataset

Figura 29: Continuación. Resultados obtenidos de la evaluación entre método de Baluja y VVRSM al embeber imágenes en los subconjuntos de Resolución 1 (64×64), 2 (128×128) y 3 (256×256)

4.2 Pruebas con DWT y SVD

En esta sección se pone a prueba la efectividad de SVD-DWT como mecanismo de compresión, se ha decidido emplear dos conjuntos de imágenes provenientes de los dataset COCO y UCID [127], con 1000 imágenes y 886 imágenes, respectivamente, el mensaje a incrustar es texto con una carga de 608,000 bytes para la versión sin SVD-DWT y 800,000 bytes para la versión con SVD-DWT, este último valor es un 31.5 % superior a la carga que representa el primer mensaje y está determinada de esta forma para observar la eficacia de SVD-DWT cuando las cargas a incrustar son superiores a 6 bpp.

En la tabla 21 se muestran los resultados de las estego-imágenes detectadas por StegExpose, así como los resultados medidos mediante las métricas de calidad PSNR, SNR, MSE, SSIM y CC, al emplear los dataset UCID y COCO para VVRSM en su versión original y con SVD-DWT.

Tabla 21: Comparativa de resultados entre incrustación de datos sin SVD-DWT y con SVD-DWT, en función de las estego-imágenes detectadas por StegExpose

Dataset	Resolución	Estego-imágenes detectadas	PSNR	SNR	MSE	SSIM	CC
UCID	512×512	780	43.081	37.104	2.922	0.992	0.999
UCID con SVD-DWT	512×512	770	43.961	37.670	2.612	0.993	0.999
COCO	512×512	957	43.786	37.923	2.718	0.992	0.999
COCO con SVD-DWT	512×512	957	44.647	38.502	2.231	0.992	0.999

Al observar los resultados de la tabla 21 podemos verificar que las diferencias entre VVRSM sin SVD-DWT y VVRSM con SVD-DWT presentan resultados similares en la métricas de desempeño, siendo la versión con SVD-DWT con un comportamiento ligeramente superior a pesar una carga de datos suprior en un 31.5 %, donde la diferencia a favor de esta implementación con respecto al PSNR es 0.9 dB para ambos dataset, un promedio de 0.6 dB para SNR, y en el MSE las diferencias en UCID y COCO se presenta una diferencia de 0.3 y 0.46 puntos a favor de VVRSM con SVD-DWT. Por otra parte, la evasión de estego-imágenes favoreció a SVD-DWT en la evaluación del dataset UCID, alcanzando 10 estego-imágenes menos, que se han detectado con StegExpose.

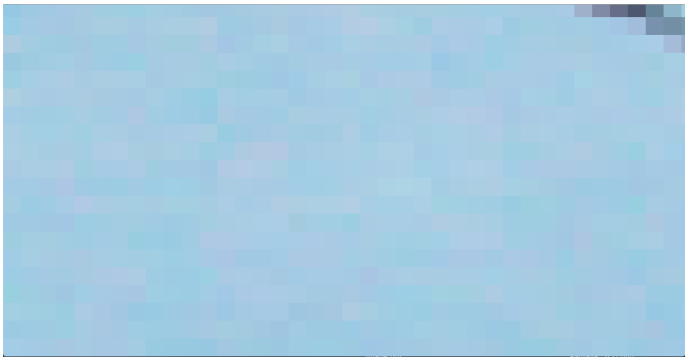
En la tabla 22 se muestran los resultados obtenidos de la evaluación los dataset UCID y COCO cuando las cargas de incrustación son de 800,000 para VVRSM con y sin SVD-DWT y pueden observar que los contrastes más notorios se pueden apreciar en los resultados de la métrica MSE, donde se aprecia una diferencia de un 310 % para el dataset UCID y 360 % para el dataset COCO donde no se empleó SVD-DWT, esto es debido a que se han empleado en casi su totalidad la modificación de los tres bits menos significativos para las estego-imágenes con respecto a la variante SVD-DWT (donde solo se llegó a modificar como máximo los dos bits menos significativos por cada píxel), y por lo tanto se aprecian grandes diferencias en esta métrica. Con respecto a las otras métricas la diferencia es menor, como referencia se tiene que la medición del PSNR es aproximadamente un 13 % mayor en la versión con SVD-DWT en ambos conjuntos, con respecto a VVRSM sin SVD-DWT, y el SNR expresa una variación aproximada 12 % para el conjunto UCID, y un 15 % para el conjunto COCO en favor a la variante de SVD-DWT, lo cual se puede apreciar en la figura 29, con respecto a las métricas SSIM y CC la diferencia es sensiblemente menor, debido a que las estego-imágenes para los 4 casos evaluados no están profundamente alteradas. Estos resultados indican que existen diferencias apreciables a nivel visual, sobre todo cuando se realiza una ampliación de las estego-imágenes, y es mucho más apreciable en aquellas donde el mensaje no se representó con SVD-DWT.

Tabla 22: Comparativa de resultados entre incrustación de datos sin SVD-DWT y con SVD-DWT

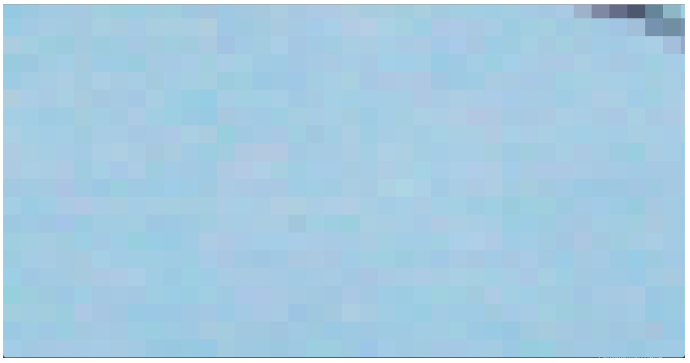
Dataset	Resolución	Variación presentada en % para MSE	PSNR	SNR	MSE	SSIM	CC
UCID sin SVD-DWT	512×512	310.987	39.161	33.155	8.123	0.976	0.998
UCID con SVD-DWT	512×512	0	43.961	37.670	2.612	0.993	0.999
COCO sin SVD-DWT	512×512	360.555	39.078	32.933	8.044	0.976	0.998
COCO con SVD-DWT	512×512	0	44.647	38.502	2.231	0.992	0.999

La figura 30 presenta tres extractos de imágenes que han sido redimensionados para corroborar el efecto de la incrustación de datos, la imagen (a) representa la imagen de portada, posteriormente la siguiente imagen (b) representa la estego-imagen con SVD-DWT y una carga total de 800,000 bytes, de esta forma podemos comprobar que los píxeles presentan variaciones en su tonalidad, de forma ligera, pero apreciable, mientras que la imagen (c), la cual contiene la incrustación por VVRSM, se puede observar que las diferencias con respecto a (a) es mucho mayor, mostrando numerosos píxeles donde es mayor el cambio de intensidades de los píxeles con cambios que tienden a un azul claro o con tonalidades variadas, lo cual se puede interpretar como los resultados elevados obtenidos en el MSE.

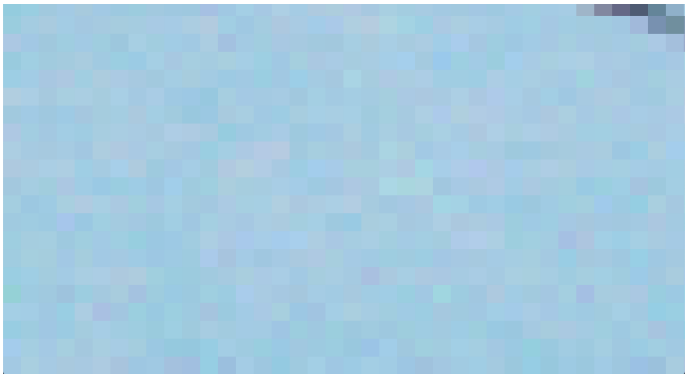
Esta adición al método es recomendable para cuando el objeto a incrustar modifica ampliamente más de 2 bits por píxel de la imagen de portada. Durante la ejecución de las pruebas se pudo observar que el consumo de recursos computacionales se eleva en aproximadamente un 50 % con respecto al proceso normal de incrustación de datos de VVRSM, por lo tanto, la condición de uso de la variante SVD-DWT está ligada con el tipo de objeto a incrustar y la necesidad de reducir la detección vía estegoanálisis.



(a) Imagen de portada



(b) Estego-imagen con SVD-DWT dataset



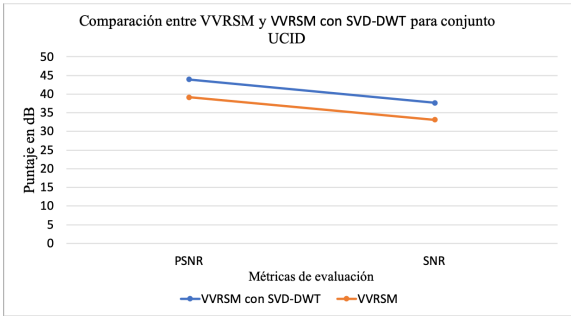
(c) Estego-imagen sin SVD-DWT dataset

Figura 30: Comparación de segmentos de estego-imágenes con imagen de portada para verificación de cambios con respecto a la aplicación SVD-DWT en la prueba de 800,000 bytes

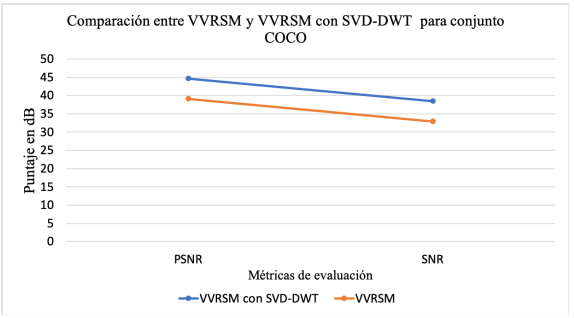
Es posible que el mensaje M_i, con la implementación de SVD-DWT se vea reducido hasta en un 33 %, esta compresión dependerá de las cualidades de D_r. Otro de los puntos que sean han considerado dentro de este análisis es, que si la imagen resultante de D_r mantienen sus características visuales con respecto a la original obtendremos mejores resultados de compresión debido a que la representación propuesta en el algoritmo de compresión de PNG o en JPEG logra reducir en una relación superior a 10 veces el tamaño original de la imagen D_r cuando existe una gran cantidad de píxeles, cuyo valor en áreas extensas de la imagen son uniformes, lo cual permite aprovechar la compresión de valores repetitivos. En este sentido se tiene que los datos contenidos en M_i son aleatorios, a pesar de que el alfabeto es limitado (dependiendo del sistema de codificación empleado incrementa o reduce este tamaño), y por lo tanto la entropía del mensaje varía, e indica que entre mejor sea el sistema de compresión aplicado a M_i obtendremos una menor cantidad

de variaciones en D_r.

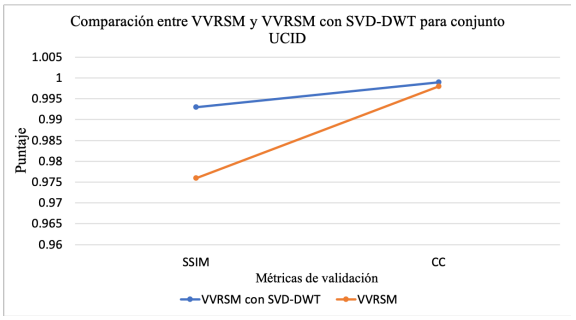
La figura 31 muestra los resultados graficados de la tabla 22, donde se puede apreciar las diferencias entre PSNR y SNR para los conjuntos UCID y COCO, como se puede observar, las diferencias obtenidas no son abruptas, pero si son significativas, teniendo en cuenta que en el PSNR cuando varia por un 1 dB esto se presenta por una escala logarítmica generando que la diferencia no sea lineal, por otra parte, en SSIM y CC en ambos conjuntos son mucho más reducidas las diferencias (diferencias menores al 2 %), como anteriormente se había señalado, así como también en la comparación MSE, la diferencias es mucho más notable, con una diferencia superior al 300 %.



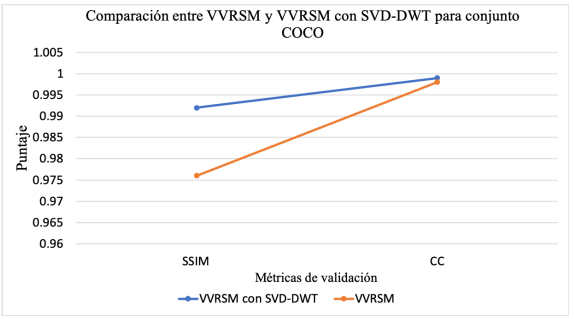
(a) Resultados obtenidos en el cálculo de PSNR y SNR para dataset UCID



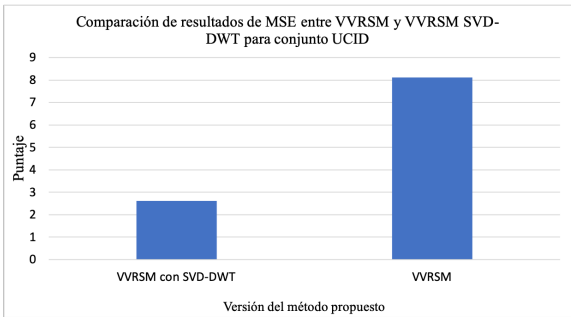
(b) Resultados obtenidos en el cálculo de PSNR y SNR para dataset COCO



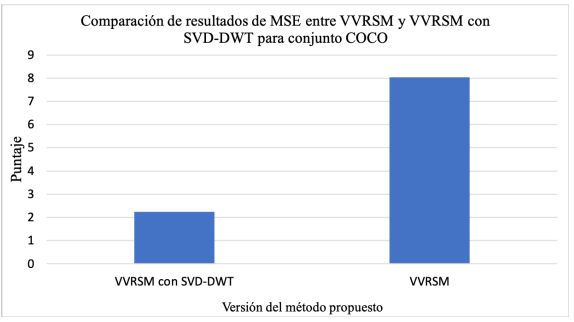
(c) Resultados obtenidos en el cálculo de SSIM y CC para dataset UCID



(d) Resultados obtenidos en el cálculo de SSIM y CC para dataset COCO



(e) Resultados obtenidos en el cálculo de MSE para dataset UCID



(f) Resultados obtenidos en el cálculo de MSE para dataset COCO

Figura 31: Comparación de resultados con métricas de calidad en la prueba de 800,000 bytes

4.3 Estegoanálisis profundo

Los métodos y herramientas de estegoanálisis son variados, debido a que tienen la finalidad de determinar sí una imagen contiene un mensaje oculto o no. Las herramientas más comunes que se utilizan son de tipo estadísticas para detectar las modificaciones que están presentes de forma secuencial.

Para validar las imágenes obtenidas a través del método VVRSM, se hace uso de la herramienta *StegExpose* debido a que combina el uso de diferentes técnicas como son: chi-cuadrada, RS y análisis de pares. La herramienta *StegExpose* se ha probado de forma exhaustiva mediante la evaluación de las estego-imágenes corroborando los resultados obtenidos en los dataset Tiny y COCO, con la generación de tres subconjuntos de 64×64, 128×128 y 256×256 píxeles.

Los resultados se comparan con el método de LSB++, con una carga de 1 bpp (máxima carga que puede incrustar este método), mientras que, para VVRSM se ha elegido de forma variable para los subconjuntos de la siguiente forma:

- 1.234 bpp para los subconjuntos 64×64 píxeles.
- 1.139 bpp para los subconjuntos de 128×128 píxeles.
- 1.059 bpp subconjuntos de 256×256 píxeles.

La tasa de incrustación es variable y oscila entre el 6 % y el 23 % superior a un 1 bpp, esto es con el propósito de demostrar la efectividad de VVRSM, por otra parte, es importante recordar que LSB++ soporta cargas máximas de 1 bpp, y su máxima efectividad de muestra en cargas de incrustación de 0.8 bpp. En la tabla 23 se presenta dicha comparación entre ambos métodos.

Tabla 23: Comparativa de detección de estego-imágenes entre LSB++ y VVRSM por StegExpose

Dataset	Resolución	Estego-imágenes detectadas en LSB++	Estego-imágenes detectadas en VVRSM
Tiny	64×64	986	906
COCO	64×64	980	412
Tiny	128×128	996	48
COCO	128×128	992	472
Tiny	256×256	996	399
COCO	256×256	994	588

En los resultados de la tabla 23 se observa que para el método LSB++ el 2 % de las estego-imágenes pueden aludir el análisis por StegExpose, para el conjunto de Tiny de 64×64 píxeles, mientras que para el subconjunto COCO para las resoluciones de 64×64 píxeles es de sólo el 1.4 % de las estego-imágenes que eluden la verificación por StegExpose. Los subconjuntos de estego-imágenes de 128×128 y 256×256 píxeles para el dataset Tiny y COCO se obtienen los porcentajes de evasión de: 0.4 %, 0.8 %, 0.4 % y 0.6 % respectivamente para LSB++. Por lo tanto, se concluye que la evasión de este método es baja cuando la carga promedio es superior a 0.8 bpp.

Por otro lado, al comparar los resultados del método VVRSM que se han obtenido de los subconjuntos procedentes de Tiny y COCO son los siguientes porcentajes de evasión 9.4 %, 58.8 %, 95.2 %, 52.8 %, 60.1 % y 41.2 % respectivamente. En la figura 32 se observa que en todas las pruebas realizadas a los subconjuntos de los dataset donde se aplicó LSB++ el promedio de evasión general es de un 1 %, contrario al caso VVRSM el cual tiene un promedio evasión de un 52.916 %, lo cual corrobora la efectividad del método propuesto.

Se realizaron pruebas en imágenes de portada procedentes de los dataset UCID y COCO con una resolución de 512×512 píxeles, en las cuales se incrustó información con VVRSM y SVD-DWT. Esta mejora ha permitido incrustar aproximadamente hasta un 33.33 % más de datos con respecto a las cargas máximas originalmente alcanzadas, logrando cargas de datos de 800,000 bytes por estego-imagen (texto incrustado),

en comparación a los 600,000 bytes anteriormente incrustados.

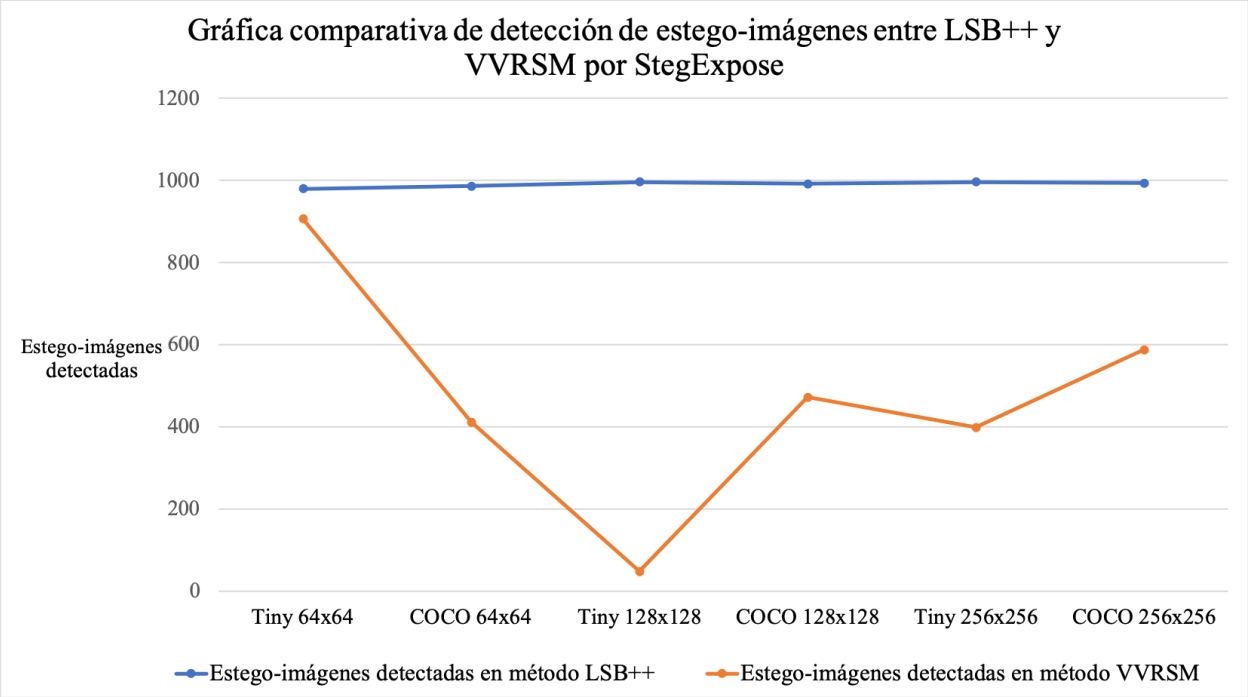


Figura 32: Gráfica comparativa de detección de estego-imágenes entre LSB++ y VVRSM por StegExpose

Como se ha mencionado con anterioridad, las métricas de calidad son importantes cuando se requiere ratificar la relación visual que existe entre la imagen de portada y la estego-imagen, y esto se refleja cuando obtenemos un PSNR superior a los 37 dB y un MSE que este por debajo de los 6 puntos, donde las diferencias no son notorias.

Tabla 24: Comparativa de resultados al aplicar StegExpose como mecanismo de detección de datos con VVRSM con y sin máscara DWT y aplicando SVD-DWT

Dataset	Resolución	Estego-imágenes detectadas	PSNR	SNR	MSE	SSIM	CC
UCID + DWT	512×512	780	43.081	37.104	2.922	0.992	0.999
UCID sin DWT	512×512	886	45.455	38.982	2.324	0.995	0.999
UCID + DWT y SVD-DWT	512×512	770	43.961	37.670	2.612	0.993	0.999
COCO + DWT	512×512	957	43.786	37.923	2.718	0.992	0.999
COCO sin DWT	512×512	999	45.156	39.026	1.984	0.993	0.999
COCO + DWT y SVD-DWT	512×512	957	44.647	38.502	2.231	0.992	0.999

En la tabla 24 se presenta un PSNR superior a 43 dB y un MSE con un puntaje menor a 3 puntos para VVRSM con SVD-DWT o sin SVD-DWT, sin embargo, esto no es garantía en superar pruebas de estegoanálisis, debido a que el resultado de evadir el estegoanálisis está ligado en evitar que un 3 % de bloques pares e impares modificados de cada imagen no superen dicha cifra cuando se emplea. En contra parte, un desempeño sobresaliente en SSIM y CC indican que existe un mejor balance en la distribución de datos. En la tabla 24 es posible apreciar que la aplicación de la máscara DWT permite modificar el número de

estego-imágenes detectadas, con un 15 % de evasión para UCID y un 4 % para COCO con respecto a las variantes sin DWT, aunque es más efectiva cuando se aplica con cargas promedio de 1.7 bpp (para mensajes con texto), en este caso, tenemos un promedio de 6.1 bpp (carga de 600,00 bytes) y 8.1 bpp (carga de 800,000 bytes), las cuales son tasas de incrustación bastante elevadas (cuando se superan los 4 bits por píxel), puesto que superan de forma representativa el 75 % de los bits disponibles de cada píxel para ser modificados (de acuerdo con las recomendaciones efectuadas en el capítulo 3). Es importante considerar que la aplicación de la máscara DWT demostró mayor efectividad para reducir la detección por estegoanálisis, pero decrementa la calidad visual, como se corrobora en los resultados obtenido por las métricas PSNR, SNR y MSE.

En el capítulo 5 se expresan las conclusiones obtenidas del presente trabajo de investigación, tomando en cuenta los objetivos planteados, y los resultados alcanzados, así como generalidades de esta investigación.

Capítulo 5

Conclusiones y Trabajo Futuro

En este capítulo se presentan las conclusiones obtenidas del presente trabajo de investigación, y el trabajo a futuro que indica la dirección a seguir con respecto al método VVRSM.

5.1 Conclusiones

- El objetivo general de este trabajo de tesis se ha cumplido, desde el punto de vista de incrustar tasas de datos superiores al 30 % con respecto a los píxeles disponibles en una imagen de portada, debido a que el mensaje a incrustar es representado mediante la aplicación de LZW, base 64 y UTF-6, permitiendo cargas de incrustación de datos superiores a los 7.0 bpp cuando el mensaje original es texto plano.
- La redistribución de píxeles es capaz de generar un reordenamiento temporal para romper con el esquema de incrustación original, ayudando a que el mensaje no sea accesible de forma directa por parte de un atacante. La forma que se propone para la redistribución de píxeles es variable, y no está limitada al uso de un solo conjunto de fórmulas, por lo que es un elemento versátil, que permite acomodar un conjunto de píxeles de acorde a las necesidades de transporte de la información a ocultar.
- La teoría de fractales es aplicada de una forma inédita, utilizando variaciones que existen en el cálculo de la dimensión fraccionaria para generar vectores, y de esta forma obtener modificaciones variables de las posiciones de símbolos procedentes de un alfabeto determinado de cada mensaje a incrustar, permitiendo usarla como un mecanismo de seguridad que impida la recuperación directa del mensaje en caso de ser extraído.
- Al aplicar las métricas de validación de calidad en las estego-imágenes, tales como, el PSNR y SNR se mantienen puntajes superiores a 40 dB y 37 dB respectivamente, lo cual indica que visualmente no manifiestan distorsiones que sugieran cambios por incrustación de mensajes secretos en ellas. En nuestros escenarios de pruebas nunca se obtuvieron estego-imágenes con calidad deficiente, cumpliendo con el objetivo de mantener el PSNR en 40 dB para un conjunto amplio de imágenes.
- De acuerdo con los escenarios de pruebas, podemos verificar que SSIM es una métrica que está más asociada a la apreciación visual humana, esto es posible corroborar con los resultados que se presentaron en los escenarios de prueba correspondientes al método de Baluja, donde un PSNR demasiado bajo y un MSE demasiado alto aparentemente sugeriría que las estego-imágenes eran técnicamente poco aptas para un proceso de esteganografía, sin embargo, se comprobó que más del 50 % de las estego-imágenes pasan correctamente las pruebas de estegoanálisis y, aunque visualmente existen variación de colores, no se deforman los objetos que están representados en las imágenes.
- En los resultados obtenidos del método VVRSM con SSIM, esta métrica obtuvo puntajes que oscilan entre 0.980 puntos y 0.990 puntos, lo cual reafirma que las estego-imágenes conservan su integridad visual, y a su vez indica que los resultados obtienen mayor validez, debido a que esta métrica corrige las falencias de PSNR y MSE, además de ser una métrica que no es común que se emplee en el análisis de imágenes, siendo que su incorporación en el análisis de calidad de imágenes es indispensable en la actualidad del entorno de la esteganografía.

- De forma general, la métrica CC presenta resultados superiores a los 0.990 puntos para las estego-imágenes obtenidas del método VVRSM, lo cual indica que los datos entre las imágenes de portada y las estego-imágenes están altamente correlacionados, esto da la pauta para interpretar que visualmente las distorsiones no son apreciables visualmente.
- La combinación del dominio espacial y de la frecuencia fue posible mediante el reemplazo de los bits menos significativos para incrustar el mensaje en la imagen de portada, y el empleo de DWT como mecanismo de alteración de los patrones de incrustación, lo cual ha permitido elevar la evasión de estegoanálisis para cargas promedio hasta de 1.7 bpp con mensajes concebidos originalmente como texto. Esta combinación permite altas cargas de incrustación de datos y robustece el modelo de seguridad para la información oculta, cuando las estego-imágenes son analizadas con herramientas de estegoanálisis como StegExpose.
- El método VVRSM presenta un mejor rendimiento cuando el objeto a incrustar es texto, debido a que la mayoría de este tipo de mensaje cuentan con subcadenas que permiten asociarlas con patrones repetitivos, mientras que al transformar imágenes en texto y ser comprimidas, presenta una menor cantidad de patrones repetitivos. Por otra parte, los valores de los píxeles generalmente oscilan entre2 y 3 dígitos, lo que dificulta la compresión de datos, pero proporciona la ventaja de incrustar los objetos sin reducir la calidad del mensaje. Se ha observado que en los mensajes comprimidos con UTF-6 se logra reducir un aproximado de 25 % el tamaño del mensaje previamente comprimido con LZW y presentado en base 64.
- Al contrastar los resultados obtenidos con respecto a los competidores presentados en el estado del arte en cuyas evaluaciones emplearon métricas como SSIM y estegoanálisis, y no solo PSNR, SNR y MSE, los resultados obtenidos fueron superiores, con excepción en las pruebas con StegExpose, donde el método de Baluja obtiene mejores resultados, cercanos a una evasión de estegoanálisis del 50 % para conjuntos de imágenes de 256×256 píxeles, pero con la particularidad que las estego-imágenes de este método muestran distorsiones y el mensaje recuperado presenta perdidas para el caso de este método, además, de que necesita de un CPU (preferentemente GPU por su gran cantidad de núcleos) de alto desempeño y memoria volátil de gran capacidad (superior a 64 GB), caso contrario a nuestra propuesta donde el mensaje recuperado permanece integro y las estego-imágenes conservan una alta tasa de calidad visual.
- La aplicación de SVD-DWT con el objetivo de reducir el tamaño del mensaje, permite incrementar la carga de datos en las estego-imágenes sin que se comprometa la calidad de estas, desde el punto visual o empleando las métricas de calidad, además se reduce la cantidad de datos que son detectados por las herramientas de estegoanálisis, el rango de datos que permite incrustar es variable, pero se pueden obtener ganancias que oscilan desde el 15 % hasta superiores a un 30 % (tomando en cuenta la incrustación alcanzada en VVRSM) lo cual permite obtener un mejor desempeño en comparación con el método originalmente planteado. Por otra parte, el consumo de recursos computacionales se incrementa, específicamente en el aparado de consumo de tiempo en CPU, con un aumento del 50 %, y en el almacenamiento temporal de memoria no volátil.
- Con base al cumplimiento del objetivo general y los objetivos específicos de esta investigación, se llega a la conclusión que nuestra hipótesis planteada se ha podido validar como cierta, debido a que se logran altas cargas de incrustación, con estego-imágenes aprobadas mediante las métricas de medición de calidad, además de que es posible evadir técnicas de estegoanálisis como la chi-cuadrada, mediante la implementación de la representación de información con técnicas de compresión y codificación, las dispersión temporal de píxeles, el aprovechamiento de la dimensión fractal y la combinación de los enfoques de incrustación de datos vía espacial y en dominio de la frecuencia.

5.2 Trabajo Futuro

Tomando en cuenta las prueba realizadas y los resultados obtenidos, se propone generar un nuevo algoritmo con la capacidad de lograr una degradación en imágenes RGB, con la finalidad de obtener imágenes con reducido tamaño y la capacidad de albergar mensajes que superen los 7.8 bpp cuando se emplea la combinación de SVD-DWT en imágenes de N dimensiones con VVRSM, debido a que este fue el límite promedio

que se alcanzó en VVRSM con SVD-DWT.

Aplicar un algoritmo de redistribución en la dispersión de píxeles con el objeto de optimizar las distribuciones temporales y reducir la distorsión en las estego-imágenes resultantes.

Codificar los algoritmos propuestos para una arquitectura GPU, y de esta forma reducir el tiempo de generación de estego-imágenes, se propone el software Numba [9], debido a que este tipo de arquitecturas permiten sincronizar cientos de procesadores y de que esta forma sea posible reducir el tiempo de cálculo, sobre todo en los procesos de dispersión de píxeles, cálculo del vector basado en la dimensión fractal, los procesos de incrustación y recuperación del mensaje, esto permitirá trabajar con una mayor cantidad de imágenes provenientes de los dataset COCO, Tiny, UCID, entre otros.

Diseñar e implementar un algoritmo de búsqueda y eliminación profunda de los archivos temporales como imágenes intermedias y estructuras de datos, los cuales muestren información sobre el proceso generación de la estego-imagen para evitar recuperación de información sensible por herramientas de rastreo de archivos eliminados en unidades de almacenamiento.

Referencias

[1] Afrah A. and Nermin H. A Survey on Cloud Data Security using Image Steganography. *International Journal of Advanced Computer Science and Applications*, 11, 01 2020.

[2] T. Abdulkhaleq Abbas and K. H. Hamza. Steganography Using Fractal Images Technique. *Steganography Using Fractal Images Technique*, 4(2):52–61, 2014.

[3] H. Al-Bahadili. A Secure Block Permutation Image Steganography Algorithm. *International Journal on Cryptography and Information Security*, 3:11–22, 2013.

[4] O. I. I. Al-Farraji. New Technique of Steganography Based on Locations of LSB. *International Journal of Information Research and Review*, pages 3549–3353, 2017.

[5] A. A. Al-Shaaby. Cryptography and Steganography: New Approach. *Transactions on Networks and Communications*, 5(6):25–38, 2017.

[6] J.-P. Allouche and J. Shallit. *"The Sierpiński Carpet" in Automatic Sequences: Theory, Applications, Generalizations*. Cambridge University Press, 4 2003.

[7] D. G. Altman and J. M. Bland. Standard deviations and standard errors. *Education and debate*, 331:903, 10 2013.

[8] Ambika, R. L. Biradar, and V. Burkpalli. Efficient Approach for Steganography Using DWT and RSA Algorithm. *International Journal of Engineering and Advanced Technology (IJEAT)*, 8:1435–1443, 2019.

[9] Continuum Analytics. Numba. http://numba.pydata.org/numba-doc/latest/release-notes.html. Accessed: 2020-06-01.

[10] I. Aqeel and M.B Suleman. A Survey on Digital Image Steganography Approaches. *INTAP 2018. Communications in Computer and Information Science*, 932:769–778, 11 2019.

[11] S. Atawneh and P. Sumari. An Overview of Frequency based Digital Image Steganography. *International Journal of Cryptology Research*, 5:15–27, 2015.

[12] S. Baluja. Hiding Images in Plain Sight: Deep Steganography. *NIPS*, pages 2069–2079, 2017.

[13] Y. Bar-Shalom, F. Palmieri, A. Kumar, and H. M. Shertukde. "Analysis of wide-band cross-correlation for time delay estimation. *IEEE Transaction on Signal processing*, 41(1):385–387, 1993.

[14] M. Barnsley. *Fractals Everywhere*. Academic Press, 2 edition, 1993.

[15] J. Barrallo Calonge. *Geometría fractal: algorítmica y representación*. Anaya Multimedia, 1 edition, 1993.

[16] N. Bell and M. Garland. Implementing sparse Matrix-Vector Multiplication on Throughput-Oriented Processor. *Nvidia research*, pages 1–11, 2009.

[17] S. Bhattacharyya, A. Khan, A. Nandi, A. Dasmalakar, S. Roy, and G. Sanyal. Pixel Mapping Method (PMM) Based Bit Plane Complexity Segmentation (BPCS) Steganography. *World Congress on In Information and Communication Technologies*, pages 207–214, 2001.

[18] S. Bhattacharyya, L. Kumar, and Sanayi G. A Novel approach of Data Hiding Using PixelMapping Method (PMM). *International Journal of Computer Science and Information Security*, 8(4):207–214, 2010.

[19] H. Bilal and A. Koyun. A new imperceptible steganography method for grayscale images. *Journal of Engineering Sciences and Design*, 8(2):357–365, 2020.

[20] A. Bilich, P. Axelrad, and K. Larson. Scientific Utility of the Signal-to-Noise Ratio (SNR) Reported by Geodetic GPS Receivers. *20th International Technical Meeting of the Satellite Division of The Institute of Navigation 2007 ION GNSS 2007*, 2:10–20, 01 2007.

[21] R. Biswas and S. K. Bandyapadhay. Random selection based GA optimization in 2D-DCT domain color image steganography. *Multimedia Tools and Applications, Springer*, 79(11):7101–7120, 2020.

[22] B. Boehm. StegExpose - A Tool for Detecting LSB Steganography. *Cornell University*, 1(1):1–11, 2014.

[23] B. Boehm. Stegexpose - a tool for detecting LSB steganography. *Cornell University*, pages 1–11, 2014.

[24] D. Bohea. *Twenty Years of attacks on the RSA Cryptosystem*. AMS, 1999.

[25] M. H. Btoush and Z. E. Dawahdeh. A Complexity Analysis and Entropy for Different Data Compression Algorithms on Text Files. *Journal of Computer and Communications*, 6(1):301–315, 2018.

[26] M. H. Btoush and Z. E. Dawahdeh. A Complexity Analysis and Entropy for Different Data Compression

Algorithms on Text Files. *Journal of Computer and Communications*, 6(1):301–315, 2018.

[27] Chang C. C.and Chuang J. C. Using a simple and fast image compression algorithm to hide secret information. *Int. J. Computer Appl*, 4(28):329–333, 2006.

[28] Ch. Cachin. An Information-Theoretic Model for Steganography, Information Hiding. *Proceedings, Lect. Notes Comp. Sci, Springer*, 1525:306–318, 04 1998.

[29] R. Chandramouli and K. P. Subbalakshmi. Current trends in steganalysis: a critical survey. *ICARCV*, 1(1):962–968, 2004.

[30] C. C. Chang and H. W. Tseng. A steganographic method for digital images using side match. *Pattern Recognition Letters*, 10(25):1431–1437, 2004.

[31] K. C. Chang, H. Tu, and C. P. Chang. Adaptive Image Steganographic Scheme Based on Tri-way Pixel-Value Differencing. *IEEE International conference on Systems, Man and Cybernetics*, pages 1165–1170, 2007.

[32] S. Dahiya. Data Encryption Based Image Steganography: A Review. *IJEDR*, 5:18–20, 2017.

[33] K. A. Darabkh, A. K. Al-Dhamari, and I. F. Jafar. A New Steganographic Algorithm Based on Multi Directional PVD and Modified LSB. *Information Technology and Control*, pages 16–36, 2017.

[34] S. Das, S. Das, B. Bandyopadhyay, and S. Sanyal. Steganography and Steganalysis: Different Approaches. *Journal of Innovative Research in Engineering Sciences*, 2010.

[35] M. de Guzmán, M. A. Martín, M. Morán, and M. Reyes. *Estructuras fractales y sus aplicaciones.* Labor, 1 edition, 1993.

[36] H. V. Desai and A. V. Desai. Image steganography using Mandelbrot fractal. *Trans Stellar*, 4(2):71–79, 2014.

[37] V. Desai Hardikkumar and A. Desai Apurva. Steganography of Messages using Mandelbrot Fractal. *VNSGU Journal of Science and Technology*, 5(1):13, 2016.

[38] Sayantan Dhar, Sudipta Maity, Bhaskar Gupta, D.R. Poddar, and Rowdra Ghatak. A CPW fed slot loop Minkowski fractal antenna with enhanced channel selectivity. In *Communications, Devices and Intelligent Systems (CODIS), 2012 International Conference on*, pages 542–545, 12 2012.

[39] T. Dhruw and D. N. Tiwari. Different Method Used in Pixel Value Differencing Algorithm. *IOSR Journal of Computer Engineering*, pages 102–109, 2016.

[40] C. Di Laura, D. Pajuelo, and G. Kemper. A Novel Steganography Technique for SDTV H.264 AVC Encoded Video. *International Journal of Digital Multimedia Broadcasting*, 10:1–9, 4 2016.

[41] F. Djebba, B. Ayad, K. A. Meraim, and H. Hamam. Comparative Study of Digital Audio Steganography Techniques. *EURASIP Journal on Audio, Speech, and Music Processing*, pages 1–16, 1 2012.

[42] T. Dumas, A. Roumy, and C. Guillemot. Image compression with Stochastic Winner-Take-All Auto-Encoder. pages 1512–1516, 03 2017.

[43] Dragos Dumitrescu, Ioan-Mihail Stan, and Emil Simion. Steganography techniques. *IACR Cryptol. ePrint Arch.*, 2017:341, 2017.

[44] E. E. A. Elgabar and F. A. Mohammed. JPEG versus GIF Images in forms of LSB Steganography. *International Journal of Computer Science and Network*, 2(8):86–93, 2013.

[45] C. Estupiñan and F. Acosta. Una nueva técnica de transmisión segura de imágenes aplicando transformaciones de color reversibles en zonas ruidosas de la imagen. *RECI Revista Iberoamericana de las Ciencias Computacionales e Informática*, 7:80, 04 2018.

[46] G. S. Eswari, N. Leelavathy, and U. S. Rani. Fractal Image Steganography Using Non Linear Model. *International Journal of Innovative Research in Computer and Communication Engineering*, 2(1):2644–2649, 2014.

[47] C. Fajardo, O. M. Reyes, and A. Ramirez. Seismic Data Compression Using 2D Lifting-Wavelet Algorithms. *Ingenierría y Ciencia*, 21(11):221–238, 2015.

[48] M. Fernández-Martínez and M.A. Sánchez-Granero. Fractal dimension for fractal structures: A Hausdorff approach revisited. *Journal of Mathematical Analysis and Applications*, pages 321–330, 1 2014.

[49] L. Fitriya, T. Purboyo, and A. Prasasti. A Review of Data Compression Techniques. *International Journal of Applied Engineering Research*, 12(19):8956–8963, 2017.

[50] J. Fridrich. *Steganography in digital media: principles, algorithms, and applications.* Cambridge University Press, 2016.

[51] J. Fridrich and M. Goljan. Practical steganalysis of digital images - state of the art. In Proceedings of SPIE. *Springer-Verlag*, pages 1–13, 2002.

[52] Borko Furht, editor. *Huffman Coding*, pages 278–280. Springer US, Boston, MA, 2006.

[53] Johannes Fürnkranz, Philip Chan, Susan Craw, Claude Sammut, William Uther, Adwait Ratnaparkhi, Xin Jin, Jiawei Han, Ying Yang, Katharina Morik, Marco Dorigo, Mauro Birattari, Thomas Stützle, Pavel Brazdil, Ricardo Vilalta, Christophe Giraud-Carrier, Carlos Soares, Jorma Rissanen, Rohan Baxter, and Luc De Raedt. *Mean Squared Error*, pages 20–26. 01 2010.

[54] A. P. Gajendra-Jagnade and L. M. P. M. Ronak-Seth. Data Security Using Cryptosteganography in Web Application. *Computer Engineering and Intelligent Systems*, 3(74-80), 2012.

[55] G. Geethaa and K. Thamizhchelvyb. Application of Chaos and Fractals in Image Stegnography A Review. *International Journal of Control Theory and Applications*, 9(45):95–106, 2016.

[56] A. M. Gómez. Redes neuronales artificiales: The Self-Organizing Maps (SOM) para el reconocimiento de patrones. *Universidad los Libertadores*, 1(1):1–12, 2013.

[57] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Upper Saddle River: Pearson education, 2008.

[58] A. K. Gulve and M. S. Joshi. A High Capacity Secured Image Steganography Method with Five Pixel Pair Differencing and LSB Substitution. *I.J. Image, Graphics and Signal Processing*, 5:66–74, 2015.

[59] R. Gupta, M. Kumar, and R. Bathla. Data Compression-Lossless and Lossy Techniques. *International Journal of Application or Innovation in Engineering and Management (IJAIEM)*, 5(7):120–125, 2016.

[60] J. D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

[61] R. Hanel and S. Thurner. Generalized (c, d)-Entropy and Aging Random Walks. *Entropy*, 15, 10 2013.

[62] V. D. Hardikkumar and A. D. Apurva. VNSGU Journal of Science and Technology. *Steganography of messages using Mandelbrot fractal*, 5(1):13–20, 2016.

[63] Hausdorff. Anexo:Fractales por dimensión de Hausdorff. https://es.wikipedia.org/wiki/Anexo:Fractales_por_dimensión_de_Hausdorff.

[64] J. A. Hernández-Servín, J. R. Marcial-Romero, V. Muñoz-Jiménez, and H. A. Montes Venegas. A Modification of the TPVD Algorithm for Data Embedding. *Pattern Recognition*, pages 74–83, 6 2015.

[65] M. Hosseini. A Survey of Data Compression Algorithms and their Applications. *Research Gate, Conference Paper*, 1(1):1–14, 2012.

[66] C. Huang and C. Yuwen. Lossless Compression Algorithm for Multi-source Sensor Data Research. *Advances in Computer Science Research (ACSR)*, 76(1):1328–1335, 2018.

[67] M. Hussain. A Survey of Image Steganography Techniques. *International Journal of Advanced Science and Technology. (IJAST)*, 54:113–125, 05 2013.

[68] M. J. Hussain and K. F. Rafat. Secure Steganography for Digital Images. *International Journal of Advanced Computer Science and Applications*, page 15, 2016.

[69] S. Jayaraman, S. Esakkirajan, and T. Veerakumar. *Digital Image Processing*. Tata McGraw Hill, 3 edition, 2010.

[70] K. H. Jung and Y. Lee-Young. Three-Directional Data Hiding Method for Digital Images. *Research Gate*, 38(2):178–191, 2012.

[71] Aranzazu Jurio, Miguel Pagola, Mikel Galar, C. Lopez-Molina, and D. Paternain. A Comparison Study of Different Color Spaces in Clustering Based Image Segmentation. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Applications - 13th International Conference*, volume 81, pages 532–541, 06 2010.

[72] A. Jyoti, S. Banerjee, and G. Gupta. High Capacity Image Steganography Using Block Randomization. *International Journal of Computer Science and Network*, 3(4):559–562, 2014.

[73] Dhruval Kachhiya, Keyur Upadhayay, and Tejaskumar Bhatt. Privacy Preserving Image Transmission Using Random Pattern Mosaic Images Steganography - Survey. *International Journal of Computer Sciences and Engineering*, 6:880–883, 11 2018.

[74] R. Kakade, N. Kasar, S. Kulkarni, S. Kumbalpuri, and S. Patil. Image Steganography and Data hiding in QR Code. *International Research Journal of Engineering and Technology*, 4:2926–2928, 2017.

[75] S. Kaltzenbeisser and F. A. Petitcolas. *Hiding for Steganography and Digital Watermarking*. Artech House, 2000.

[76] S. Katzenbeisser and F. A. P. Peticolas. *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House, 2000.

[77] G. Kaur and E. G. Deep. HSI Color Space Conversion Steganography using Elliptic Curve. *International Journal of Innovations and Advancement in Computer Science*, 4:63–67, 2015.

[78] Harpreet Kaur and Jyoti Rani. A Survey on different techniques of steganography. *MATEC Web of Conferences*, 57:02003, 01 2016.

[79] P. Kavitha. A Survey on Lossless and Lossy Data Compression Methods. *International Journal of Computer Science and Engineering Technology (IJCSET)*, 7(3):110–114, 2016.

[80] S. S. Khaire and S. L. Nalbalwar. Review: Steganography Bit Plane Complexity Segmentation (BPCS) Technique. *International Journal of Engineering Science and Technology*, 2(9):4860–4868, 2010.

[81] Zakir Khan, Mohsin Shah, Muhammad Naeem, Toqeer Mahmood, Noor Amin, and Danish Shehzad. Threshold-based Steganography: A Novel Technique for Improved Payload and SNR. *International Arab Journal of Information Technology*, 13:381–386, 07 2016.

[82] G. Khandappalavar and G. Shrividya. Encryption of an Image Using Least Significant Bit Substitution Method and Arnold Transformation. *International Journal of Combined Research and Development*, 4:534–538, 2015.

[83] H. Kim. A study on the Cryptographic Algorithm for NFC. *Indian Journal of Science and Technology*, 9(1):1–5, 2016.

[84] T. Kohonen. Self-organized formation of topologically correct feature maps . *Biological Cybernetics*, 43:59–69, 1982.

[85] G. R. Kumar, M. M. P. Reddy, and T. L. Kumar. An Implementation of LSB Steganography Using DWT Technique. *International Journal of Engineering Research and General Science*, 2:398–403, 2014.

[86] M. K. Kumar and K. Kishore. Accelerating Sparse Matrix Vector Multiplication in Iterative Methods Using GPU. *Internacional Conference on Parallel Processing*, pages 612–621, 2011.

[87] Tzu-Chuen L. and Thanh V. *Reversible steganography techniques: A survey*, pages 189–213. Digital Media Steganography Principles, Algorithms, and Advances, 01 2020.

[88] N. La Serna Palomino, L. Pro Concepcion, Souvik, L. Kumar, and G. Sanayi. Watershed: un algoritmo eficiente y flexible para segmentación de imágenes de geles 2-DE. *Revista de Investigación de Sistemas e Informática*, 7(2):35–41, 2010.

[89] Y. P. Lee, J. C. Lee, W. K. Chen, K. C. Chang, and I. J. Su. High-payload image hiding with quality recovery using tri-way pixel-value differencing. *Information Sciences*, pages 214–225, 1 2012.

[90] T. Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollor. Microsoft COCO: Common Objects in Context. In arXiv.org, editor, *Advances in Neural Information Processing Systems 30*, pages 1–15. arXiv.org, 2014.

[91] Shaohui Liu, Lin Ma, Hongxun Yao, and Debin Zhao. Universal Steganalysis Based on Statistical Models Using Reorganization of Block-based DCT Coefficients. In *Proceedings of the Fifth International Conference on Information Assurance and Security, IAS 2009*, volume 1, pages 778–781, 01 2009.

[92] Z. Lu, Y. Wen, F. Yu, R. Shen, and W. Sun. High performance reversible data hiding for block truncation coding compressed image. *Springer-Verlag*, 1(1):1–10, 2013.

[93] J. Luo, R. G. Zhou, G. F. Luo, Y. C. Li, and G. Z. Liu. Traceable Quantum Steganography Scheme Based on Pixel Value Differencing. *Scientific Reports*, 9(1):15134, 2019.

[94] Arun Mahanta, Hemanta Sarmah, Ranu Paul, and Gautam Choudhury. Julia set and some of its properties. *International Journal of Applied Mathematics & Statistical Sciences (IJAMSS)*, 5:99–124, 02 2016.

[95] B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freeman, 2 edition, 1982.

[96] F.U. Mangla, S. Nokhaiz, M. Ramzan, and U. I. Lali. A novel steganography technique using grayscale image segmentation. *International Journal of Advanced and Applied Sciences*, 6(5):84–91, 2019.

[97] R. Meenakshi and K. Kuppusmay. The Use of Least Significant Bit Technique at Various Color Spaces for secure Steganography with Performance Evaluation. *IJARCSSE*, 1(5):1047–1051, 6 2014.

[98] V. Miszalok and V. Smolej. *Flower Images*. P. Henning: Taschenbuch Multimedia, 2006.

[99] K. Muhammad, J. Ahmad, H. Farman, and K. Jan. A New Image Steganographic Technique using Pattern based Bits Shuffling and Magic LSB for Grayscale Images. *Sindh University Research Journal*, 47:723–728, 2015.

[100] S. Muke, S. Shinde, C. Mistry, and P. Jawalkar. NFC Hardware Device Based Access Control System using Information Hiding. *International Journal of Innovative Research in electrical, electronics, Instrumentation and Control Engineering*, 5(1):69–72, 2017.

[101] A. Muñoz. *Privacidad y ocultación de información digital. Esteganografía*. Ra-Ma, 3 edition, 2017.

[102] R. Naoum, A. Shihab, and S. AlHamouz. Enhanced Image Steganography System based on Discrete Wavelet Transformation and Resilient Back-Propagation. *International Journal of Computer Science and Network*, 15(1):6–18, 2015.

[103] D. Nehete and A. Bhide. Skin Tone Based Secret Data Hiding in Images. *International Journal of*

Current Engineering and Technology, pages 18–24, 2014.

[104] M.R Nelson. *The Data Compression Book*. MT Books, 2 edition, 1995.

[105] A. Nissar and A. H. Mir. Classification of steganalysis techniques: A study. *Digital Signal Processing*, 20(6):1758–1770, 2010.

[106] A. S. Nori and A. M. Al-Qassab. Steganographic Technique Using Fractal Image. *International Journal of Information Technology and Business Management*, 52-59:8, 2012.

[107] L. Ouyang, J. H. Park, and H. Kau. Performance of Efficient Steganographic Methods for Image and Text. *Pendiente*, 7(1):29–33, 2016.

[108] A. Palmer. Metodología de las ciencias del comportamiento. *Universitat de les Illes Belears*, 43, 2002.

[109] D. D. Patil. Text Information Hiding in Image by BBPVD Steganography Techniques. *International Journal of Advanced Information Science and Technology*, 14:56–61, 6 2013.

[110] H. O. Peitgen, J. Hartmut, and S. Dietmar. *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag, 2 edition, 2004.

[111] F. Peng, X. Li, and B. Yang. Adaptive reversible data hiding scheme based on integer transform. Signal Process. *Pendiente*, 2(92):54–62, 2012.

[112] R. L. Plackett. Karl Pearson and the Chi-Squared Test. *International Statistical Review (International Statistical Institute (ISI))*, 51(1):59–72, 1983.

[113] G. Prabakaran, R. Bhavani, and S. Sankaran. Dual Transform Color Image Steganography Method. *International Journal of Innovative Research in Science, Engineering and Technology*, pages 129–135, 2014.

[114] N. Prokhozhev, O. Mikhailichenko, A. Sivachev, D. Bashmakov, and A. Korobeynikov. Passive Steganalysis Evaluation: Reliabilities of Modern Quantitative Steganalysis Algorithms. *Advances in Intelligent Systems and Computing, Springer*, 451(1):89–94, 2016.

[115] Y. Puri and G. Deep. Image Seeded Steganography: Steganography Using Seed Values. *International Journal Of Scientific Research And Education*, 3:4004–4012, 2015.

[116] Jiaohua Q., Yuanjing L., Xuyu X., Yun T., and Huajun H. Coverless Image Steganography: A Survey. *IEEE Access*, 7:171372–171394, 11 2019.

[117] K. Qazanfari and R. Safabakhsh. A new steganography method which preserves histogram: generalization of LSB++. *Information Sciences, Elsevier*, 2(1):90–101, 2014.

[118] R. Queirós, P.M. Girão, and A. Serra. Cross-Correlation and Sine-Fitting Techniques for High Resolution Ultrasonic Ranging. In *IEEE Instrumentation and Measurement Technology Conf.*, pages 552–556, April 2006.

[119] T. Radu, G. Shuhang, W. Jiqing, L. Van Gool, L. Zhang, M.H. Yang, M. Haris, et al. NTIRE 2018 Challenge on Single Image Super-Resolution: Methods and Results. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1–10, June 2018.

[120] A. S. Ragab, A. S. Mohmed, and M.S Hamid. Efficiency of Analytical Transforms for Image Compression. *5th National Radio Science Conference)*, 1, 1998.

[121] Mamta Rani and Vinod Kumar. Superior Mandelbrot Set. *Research in Mathematical Education*, 8:279–291, 01 2004.

[122] R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 1(1):120–126, 1978.

[123] R. Roy and S. Changder. Steganography with Projection aided Payload Dimension Reduction and Reconstruction for Military Covert Communication. *International Journal of Computer Applications*, 139(3):32–37, 2016.

[124] O. Russakovsky, J. Deng, H. Su, Krause J., S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[125] D. Salomon and G. Motta. *Handbook of Data Compression*. Springer, 2010.

[126] L.J. Sankpal, S. Mundhe, M. Kotwal, P. Machale, and S. Malchikare. NFC Based Access Control System Using Image Hiding. *International Journal of Innovative Research in Computer and Communication Engineering*, 5(1):5352–5355, 2017.

[127] G. Schaefer and S. Michal. UCID, "An uncompressed color image database". *Storage and Retrieval Methods and Applications for Multimedia*, 5307(1):472–480, 2004.

[128] P. Schober, C. Boer, and L. Schwarte. Correlation Coefficients: Appropriate Use and Interpretation. *Anesthesia and Analgesia*, 126:1763–1768, 5 2018.

[129] B. S. Shashikiran, K. Shaila, and K. R. Venigopal. Hybrid Domain Steganography for Multiple Images using DWT-LSB Method. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, 9(2):1326–1334, 2019.
[130] M. Shobana. An Efficient Image Steganographic Algorithm Using CMYK Color Model. *International Journal of Research and Innovations in Science and Technology*, pages 25–31, 2015.
[131] G. Simmons. *The Prisoners Problem and the Subliminal Channel in Proceedings of Crypto*. Plenum Press, 1984.
[132] A. Singh and E. Meenakshi. Research Paper on Text Data Compression Algorithm using Hybrid Approach. *International Journal of Computer Science and Mobile Computing*, 3(12):1–10, 2014.
[133] R. Singh-Brar and B. Singh. A Survey on Different Compression Techniques and Bit Reduction Algorithm for Compression of Text/Lossless Data. *IJARCSSE*, 3(3):1579–1582, 2013.
[134] S. Singhal and R. S. Rathore. Detailed Review of Image Based Steganographic Techniques. *IJCST*, 6:93–95, 2015.
[135] A. G. Sofloo and M. Aghayi. Steganography in Least Significant Bit. *Journal of Innovative Research in Engineering Sciences*, 3:8–14, 2017.
[136] H. Soleymania and A. H. Taherinia. Steganography. *Elsevier)*, 1(1):1–12, 2018.
[137] B. Stanislaus. Sierpinski triangle. https://upload.wikimedia.org/wikipedia/commons/4/45/Sierpinski_triangle.svg. Accessed: 2020-09-18.
[138] V. Stoyanova and Z. Tasheva. Research Of The Characteristics Of A Steganography Algorithm Based On LSB Method Of Embedding Information In Images. *Technics Technologies Education Safety*, pages 1–4, 2015.
[139] I. M. A. D. Suarjaya. A New Algorithm for Data Compression Optimization. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 3(8):90–101, 2012.
[140] G. Swain. A steganographic method combining LSB substitution and PVD in a block. *Procedia Computer Science, Elsevier*, pages 39–44, 2016.
[141] P. Symes. *Video Compression Demystified*. McGraw-Hill, 2001.
[142] K. Thamizhchelvy and G. Geetha. Data Hiding Technique with Fractal Image Generation Method using Chaos Theory and Watermarking. *Indian Journal of Science and Technology*, 7(9):1271–1278, 2014.
[143] P. Thomas. Literature Survey On Modern Image Steganographic Techniques. *International Journal of Engineering Research and Technology*, 5:107–111, 2013.
[144] V. Tyagi. *Understanding Digital Image Processing*. CRC Press, 09 2018.
[145] A. Umbarkar, P. R. Kamble, and A. V. Thakre. Comparative Study of Edge Based LSB Matching Steganography for Color Images. *ICTACT Journal On Image And Video Processing*, 6(3):1185–1191, 2016.
[146] Clinton V. S. Conductivity properties of the Sierpinski triangle. *arXiv:1710.06346*, 10 2017.
[147] A. Vaishali and A. Kajal. Increasing data hiding capacity of BPCS steganography using LZW compression technique. *International Journal of Advanced Computational Engineering and Networking*, 3(7):55–60, 2015.
[148] R. Varalakshmi. Digital Steganography for Preventing Cybercrime using Artificial Intelligence Technology. *Journal of Critical Reviews*, 7(6):749–753, 2020.
[149] C. L. Velasco-Bautista, J. C. López-Hernández, M. Nakano Miyatake, and H. M. Pérez-Meana. Esteganografía en una imagen digital en el dominio DCT. *Científica*, 11(4):169–176, 2007.
[150] M. A. Vorontsova, V. V. Dudorovb, M. O. Zyryanovab, V. V. Kolosovb, and G. A. Filimonovb. Bit Error Rate in FreeSpace Optical Communication Systems with a Partially Coherent Transmitting Beam. *Optical Waves Propagation*, 26(3):185–189, 2015.
[151] P. Wang, Z. Pan, M. Zhag, R. W.H. Lau, and H. Song. The alpha parallelogram predictor: A lossless compression method for motion capture data. *Information Sciences, Elseivier*, 232(20):1–10, 2013.
[152] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):1–13, 2004.
[153] Z. Wang, E. P. Simoncelli, and A. Bovik. Multi scale Structural Similarity for Image Quality Assessment. *Conference Record of the Thirty-Seventh Asilomar Conference on Signal, Systems and Computers*, 2(7):1–5, 11 2003.
[154] P. Wayner. *Disappearing Cryptography, Second Edition: Information Hiding: Steganography and Watermarking*. Morgan Kaufmann, 2 edition, 2002.

[155] E. W. Weisstein. Sierpiński Carpet. https://mathworld.wolfram.com/SierpinskiCarpet.html. Accessed: 2020-08-31.

[156] A. Westfeld and A. Pfitzmann. Attacks on steganographic systems. *Proceedings of the Third International Workshop on Information Hiding*, 1(1):60–70, 2000.

[157] D. Wu and W. Tsai. A steganographic method for images by Pixel Value Differencing. *Pattern Recognition Letters*, 9:24, 2003.

[158] H.C. Wu, N.I. Wu, C.S. Tsai, and M.S. Hwang. Image Steganographic Scheme Based on Pixel-Value Differencing and LSB Replacement Methods. *IEEE Proceedings on Vision, Image and Signal Processing*, 151(5):611–615, 2005.

[159] N. I. Wu, K. C. Wu, and C. M. Wang. Exploring Pixel Value Differencing and base decomposition for low distortion data embedding. *Applied Soft Computing*, pages 942–960, 10 2011.

[160] L. Xin, Y. W. Qiao, and J. Zhang. A Steganographic Method for Digital Images with Four-Pixel Differencing and Modified LSB Substitution. *Journal of Visual Communication and Image Representation*, 22(1):54–62, 2011.

[161] A. Yahya, N. Hamid, R. B. Ahmad, and J. M. Chuma. *Steganography Techniques for Digital Images*. Springer, 2019.

[162] E. S. Yakovleva, A. A. Makarov, V. N. Gorbachev, and E. M. Kainarova. Wavelet Methods in Steganography. *KnE Engineering*, 3(5):318–330, 2018.

[163] E. S. Yakovleva, A. A. Makarov, V. N. Gorbachev, and E. M. Kainarova. Wavelet Methods in Steganography. *KnE Engineering*, 3(5):318–330, 2018.

[164] X. Yang, S. Parthasarathy, and P. Sadayappan. Fast sparse Matrix Vector Multiplication on GPUs: Implications for graph mining. *Proceedings of the VLDB Endowment*, 4(4):231–242, 8 2011.

[165] A. Zenati, W. Ouarda, and A. M. Alimi. SSDIS-BEM: A New Signature Steganography Document Image System based on Beta Elliptic Modeling. *Enginnering Science and Technology Intertational Journal*, 23:470–482, 2019.

[166] K. A. Zhang, A. Cuesta-Infante, and K. Veeramachaneni. SteganoGAN: High Capacity Image Steganography with GANs. *arXiv preprint arXiv:1901.03892*, 2019.

[167] X. Zhang and S. Wang. Efficient Steganographic Embedding by Exploiting Modification Direction. *IEEE Communications Letters*, 10(11):1431–1437, 2006.

[168] D. Zou, Y. Q. Shi, W. Su, and G. Xuan. Steganalysis based on Markov Model of Thresholded Prediction-Error Image. *IEEE International Conference on Multimedia and Expo*, 52(9):1365 – 1368, 2006.

Anexo A

Técnicas adicionales de compresión de datos

En el presente anexo se muestra una técnica de compresión de datos mediante la generación de bloques de 7 y 6 bit que representen caracteres del mensaje a ocultar para el proceso de esteganografía.

La figura A1 representa dos bloques lógicos de 7 bits, los cuales representan dos símbolos distintos.

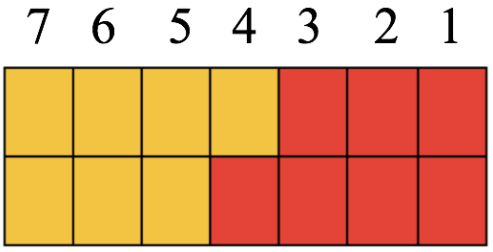


Figura A1: Ejemplo de dos bloques de 7 bits

La figura A2 muestra dos bloques de 6 bits para representar dos símbolos distintos. Aplicando los bloques de 7 bits se puede obtener hasta un 12.5 % de reducción en comparación cuando se utilizan bloques de 8 bits, por otra parte, si se utilizan bloques de 6 bits para representar caracteres se puede obtener hasta 25 % en la cantidad total de bits que se empleados para representar un mensaje.

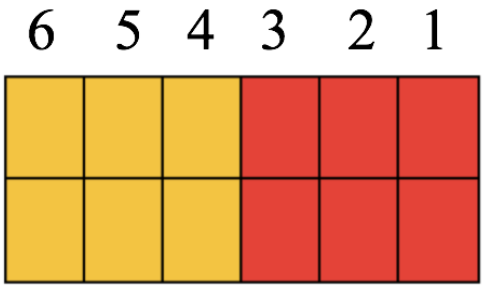


Figura A2: Ejemplo de dos bloques de 6 bits

Los bloques de 7 y 6 bits se forman a través de la selección de los bits menos significativos de los píxeles de una imagen, esto se puede observar en la figura A3, donde 6 píxeles forman un bloque de 6 bits.

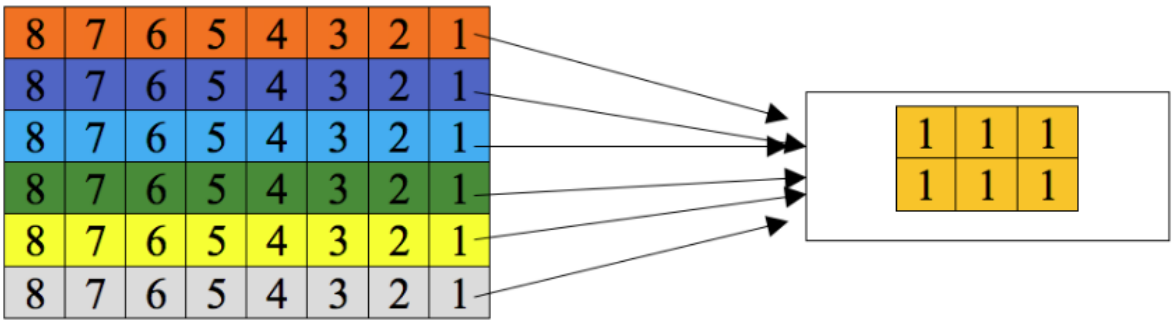


Figura A3: Conformación de bloques de 6 bits

La figura A3 muestra cómo se forma un bloque de 6 bits (casillas marcada por 1), si los datos a incrustar en una estego-imagen no pueden ser representados por el bit de menor peso, se usarán los siguientes 2 bits (2do bit y 3er bit de menor peso).

La codificación base 64 bits cuenta con 64 caracteres, estos necesitan de 6 bits para ser representados más el caracter '=' empleado para verificación del mensaje, por lo tanto, se tendría que proceder con una reasignación de los bits como se muestra en la tabla A1.

Tabla A1: Ejemplo de representación de datos con codificación

Representación binaria	Símbolo representado	Representación binaria	Símbolo representado
000000	A	010110	W
000001	B	010111	X
000010	C	011000	Y
000011	D	011001	Z
000100	E	011010	a
000101	F	011011	b
000110	G	011100	c
000111	H	011101	d
001000	I	011110	e
001001	J	011111	f
001010	K	100000	g
001011	L	100001	h
001100	M	100010	i
001101	N	100011	j
001110	O	100100	k
001111	P	100101	l
010000	Q	100110	m
010001	R	100111	n
010010	S	101000	o
010011	T	101001	p
010100	U	101010	q
010101	V	101011	r

Tabla A1: continuación, Ejemplo de representación de datos con codificación

Representación binaria	Símbolo representado	Representación binaria	Símbolo representado
101100	s	110110	2
101101	t	110111	3
101110	u	111000	4
101111	v	111001	5
110000	w	111010	6
110001	x	111011	7
110010	y	111100	8
110011	z	111101	9
110100	0	111110	+
110101	1	111111	/

Las columnas anteriores representan un ejemplo, por lo cual la organización puede cambiar una vez implementada en código fuente y depende de las necesidades específicas del proceso de esteganografía que se esté llevando a cabo. Si la mayor parte de los símbolos codificados están en base 64, se puede reducir en aproximadamente 25 % el tamaño de la incrustación con respecto a la incrustación clásica de 8 bits por símbolo.

Análisis de subcadenas redundantes en mensajes

Cuando un mensaje es codificado en base 16 es posible aplicar dos tipos de compresiones adicionales, las cuales están presentas en la figura A4, esto consiste en que sí un valor numérico representado en forma hexadecimal tiene de vecino un elemento del mismo valor, este deberá ser sustituido por un símbolo diferente al que existe dentro del alfabeto de la base 16, un ejemplo sería, si en una sección existen los valores “AE AE AE BC”, podemos observar que el número “AE” (base 16) se repite 3 veces y el último es diferente, por lo tanto solo se tomará AE para realizar la compresión, de esta forma si elegimos el símbolo z para reemplazar “AE”, la subcadena quedaría de la siguiente forma “AEzzBC”, con esto se consigue la eliminación de 2 caracteres. Esta técnica es aplicable debido a que en imágenes y en texto existe una gran cantidad de zonas donde se repiten valores y que su posición es consecutiva, es posible generar compresión sobre la ellos. La segunda opción de compresión se presenta cuando la sección analizada cuenta con una cantidad de valores repetidos superiores a dos elementos iguales y consecutivos, lo conveniente es expresar el número de caracteres iguales consecutivos y evitar escribir varias veces los símbolos, un ejemplo sería “AE AE AE AE AE AE AE BC”, y quedaría expresado como “AE6zBC”, donde 6 es el número de veces que “AE” se ha repetido y z representa “AE”.

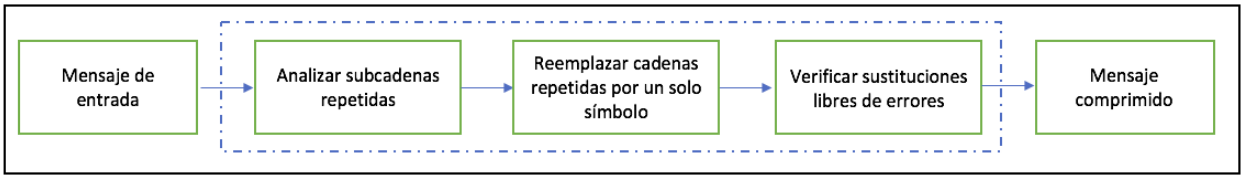


Figura A4: Compresión de datos por valores vecinos redundantes

Anexo B

Ecuaciones de fractales deterministas

En el presente anexo se muestran los nombres y las respectivas ecuaciones para el cálculo de la dimensión exacta de los fractales deterministas más comunes según la dimensión de Hausdorff [63]. En la tabla B1 están presentes los fractales deterministas más comunes.

Tabla B1: Fractales deterministas según la dimensión de Hausdorff

Nombre	Valor exacto	Valor
Bifurcación de la curva logística	$Log\,(2)\,/log(d)$	4.498
Conjunto de Cantor	$Log\,(2)\,/log\,(3)$	0.6309
Espectro del hamiltoniano de Fibonacci	$Log(1+\sqrt{2})$	0.88137
Conjunto de Smith-Volterra-Cantor	1	1
Contorno de la isla de Gosper	$Log\,(8)\,/log\,(7)$	1.0686
Conjunto de Julia Dentrita	Medida	1,2
Fractal de Fibonacci(60°)	$3((log(\phi))/log((3+\sqrt{13})/2)$	1.2083
Atractor de Hénon	Calculado	1,26
Curva de Koch	$Log\,(4)\,/log\,(3)$	1,2619
Frontera de la curva del terdragón	$Log\,(4)\,/log\,(3)$	1,2619
Polvo de Cantor bidimensional	$Log\,(4)\,/log\,(3)$	1,2619

Tabla B1: Continuación. Fractales deterministas según la dimensión de Hausdorff

Nombre	Valor exacto	Valor
Conjunto de Julia (z^2)-1	Calculado	1,2683
Circunferencia de Apolonio	Calculado	1,3057
Conejo de Douady	Calculado	1,3934
Fractal de Vicsek	$Log\,(5)\,/(Log\,(3)$	1,4649
Curva cuadrática de Kock(tipo 1)	$Log\,(5)\,/(Log\,(3)$	1,4649
Curva de la punta de fleche de Sierpinski	$Log\,(3)\,/log\,(2)$	1.585
Triángulo de Pascal módulo 3	$1 + log\,(2)\,/log\,(3)$	1.6309
Fractal de Fibonacci	$3(log(\phi)/log(1+\sqrt{2}))$	1.6379
Triángulo de Pascal	$1 + log\,(3)\,/log\,(5)$	1.6826
Fractal de molinete	$Log\,(4)\,/log(\sqrt{5})$	1.7227
Hexácopo	$Log\,(7)\,/log\,(3)$	1.7712
Curva de von Koch a 85º	$Log\,(4)\,/Log\,(2\,(1+cos\,(85)))$	1.7848
Pentácopo	$Log\,(6)\,/log(1+\phi)$	1.8617
Alfombra de Sierpinski	$Log\,(8)\,/log\,(3)$	1.8928
Polvo de Cantor tridimensional	$Log\,(8)\,/log\,(3)$	1.8928
Frontera de la curva de Lévy	Estimado	1,9340
Teselación de Penrose		1,974
Frontera del conjunto de Mandelbrot	2	1,9340
Conjunto de Julia	2	2
Curva de Sierpinski	2	2
Curva de Hilbert	2	2

Tabla B1: Continuación. Fractales deterministas según la dimensión de Hausdorff

Nombre	Valor exacto	Valor
Curva de Peano	2	2
Curva de Moore	2	2
Curva de Lebesgue o de orden z	2	2
Curva del dragón	$Log\,(2)\,/log(\sqrt{2})$	2
Curva del terdragón		2
T-cuadrado	$Log\,(4)\,/log\,(2)$	2
Curva de Gosper	$Log\,(4)\,/log\,(2)$	2
Tetraedro de Sierpinski	$Log\,(4)\,/log\,(2)$	2
Fractal H	$Log\,(4)\,/log\,(2)$	2
Árbol de Pitágoras	$Log\,(2)\,/log(2/\sqrt{2})$	
Cruz griega fractal en 2D	$Log\,(4)\,/log\,(2)$	2
Atractor de Lorenz	Calculado	2,06
Dodecaedro fractal	$Log\,(20)\,/log(2+\phi)$	2.3296
Superficie cuadrática tridimensional de Koch (tipo 1)	$Log\,(13)\,/log\,(3)$	2.3347
Intersticios entre las esferas de Apolonio		2.4739
Superficie cuadrática tridimensional de Koch (tipo 2)	$Log\,(32)\,/log\,(4)$	2.5
Teseracto de Cantor	$Log\,(16)\,/log\,(3)$	2.5237
Icosaedro fractal	Calculado	2,5819
Cruz griega fractal en 3D	$Log\,(12)\,/log(1+\phi)$	2.5849
Cruz griega fractal en 3D	$Log\,(6)\,/log\,(2)\,2$	2.5849
Octaedro fractal	$Log\,(6)\,/log\,(2)$	2.5849

Tabla B1: Continuación. Fractales deterministas según la dimensión de Hausdorff

Nombre	Valor exacto	Valor
Superficie de Koch	$Log\,(20)\,/log\,(3)$	2.5849
Esponja de Menger	$Log\,(20)\,/log\,(3)$	2.7268
Curva de Hilbert en 3D	$Log\,(8)\,(log\,(2)$	3
Curva de Lebegue en 3D	$Log\,(8)\,(log\,(2)$	3
Curva de Moore en 3D	$Log\,(8)\,(log\,(2)$	3
Curva cuadrática de Koch (tipo 2)	$Log\,(3)\,/Log\,(4)$	1.5
Frontera de la curva del dragón	$Log((1+\sqrt[3]{73}-6\sqrt{87}+\sqrt[3]{73}+6\sqrt{87})/3)/log\,(2)$	1.5236
Árbol de tres ramas	$Log\,(3)\,/log\,(2)$	1.5805
Triángulo de Sierpinski	$Log\,(3)\,/log\,(2)$	1.585

Anexo C

Pseudocódigos de reglas de operación en *VVRSM*

En el presente anexo se muestra los pseudocódigos que presentan los aspectos generales sobre el método VVRSM y el análisis de reglas de recuperación de datos, mediante la generación de un analizador léxico y sintáctico de cadenas.

Es necesario determinar las características del mensaje a ser incrustado y el tratamiento al que debe de ser sometido, esto se muestra en el pseudocódigo C1.

Pseudocódigo C1 Determinación del tipo de objeto a incrustar

```
1: Inicio
2: Tipo, mensaje, Tamano, Ratio, M64, Mr
3: Tamano = Size(mensaje) // Determinar tamaño del mensaje
4: Leer (mensaje) // Carga mensaje
5: Tipo = Objeto(mensaje) // Determina el tipo de objeto que es
6: if Tipo == "texto" then
7:     M64 = B64(LZW(mensaje)) // Comprimir y codificar en base 64
8:     Ratio = Comp (mensaje, Mb64) Calcular el ratio de compresión final
9: end if
10: if Tipo == "escala de gris" then
11:     M64 = B64(LZW(mensaje)) // Comprimir y codificar en base 64
12:     Ratio = Comp (mensaje, Mb64) Calcular el ratio de compresión final
13: end if
14: if Tipo == "RGB" then
15:     Mb64 = B64(LZW(mensaje)) // Comprimir y codificar en base 64
16:     Ratio = Comp (mensaje, Mb64) Calcular el ratio de compresión final
17: end if
18: Mr= Mr + Tipo // Reglas de control de compresión
19: "Enviar datos" ->(Tamano, Ratio) // Se envía datos para los procesos de incrustación
20: Fin
```

En el pseudocódigo C1 se extraen las características principales del objeto a incrustar, como ratio de compresión de *mensaje*, el objeto que se está incrustando, y el ratio de compresión, estos parámetros son necesarios para realizar proceso de incrustación en la imagen portada.

Cuando se han incrustado los datos tanto en el dominio del espacio y de la frecuencia se debe de proceder a generar el análisis de calidad de la imagen, para ello se debe de ejecutar el pseudocódigo C2.

Pseudocódigo C2 Análisis de calidad del estego-objeto

Inicio
limites[x] //límites deseables de las métricas de calidad
E'_m // Estego-imagen
a = MSE(E'_m) //MSE
s = SSIM(E'_m) // SSIM
p = PSNR(E'_m) // PSNR
n = SNR(E'_m) // SNR
c = PSNR(E'_m) // C
if a >4 **then**
    "Probable alteración visual en imagen"
else
    Correcto
end if
if s <0.9 **then**
    "Probable alteración visual en imagen"
else
    Correcto
end if
if p >38 dB **then**
    Correcto
else
    "Probable alteración visual en imagen"
end if
if n >30 dB **then**
    Correcto
else
    "Probable alteración visual en imagen"
end if
if c >0.9 **then**
    Correcto
else
    "Probable alteración visual en imagen"
end if
if *limites*[a] >8 o *limites*[s] <0.9 o *limites*[p] <35 dB o *limites*[n] <27 dB o *limites*[c] <0.9 **then**
    Imprimir ("La estego-imagen no supera el mínimo aceptable")
end if
Fin

Si E'_m no cumple con los parámetros mínimos deberá de ser rechazada, debido a que es probable que las alteraciones sean demasiado notables.

Analizador léxico y sintáctico propuesto

La construcción de un analizador léxico y sintáctico es necesario para obtener las reglas de producción de una gramática, para posteriormente generar la validación de cadenas de bits que representen símbolos o instrucciones. La figura C1 presenta un ejemplo de gramática. Analizando la figura C1, todos los elementos anteriores al símbolo "=>" son los símbolos no terminales, posteriormente, toda gramática se presenta por un símbolo inicial, en este caso es "S", los símbolos no terminales son aquellos que se encuentran a la izquierda "=>" (parte izquierda de la gramática), los símbolos terminales son aquellos localizados en el lado derecho de la gramática y que no tienen referencia en el lado izquierdo.

Figura C1: Ejemplo de gramática

La primera parte del analizador léxico se muestra en el pseudocódigo C3, cuya función es la identificación de símbolos de un lenguaje, en este caso se ha tomado como cadena que contiene las reglas de un lenguaje a M_r. El pseudocódigo C3, se especifican cuatro pasos esenciales para generación de la gramática, el primer paso del pseudocódigo genera las reglas de producción, cada regla de producción es separada por un salto de línea, se representa por el símbolo "/n". Al localizarse un salto de línea se terminan de unir los caracteres que conforman a las reglas de producción, el proceso termina cuando se encuentre un renglón con 0 caracteres escritos. Una vez ejecutada la segunda parte del pseudocódigo C3 se procede a localizar los símbolos no terminales.

Pseudocódigo C3 Analizador léxico

```
1: Inicio
2: M_r, reglas, produccion // cadena a analizar, arreglo de reglas, producciones encontradas
3: Verificar si M_r está vacía
4: if Size(M_r) == "vacia" then
5:     exit // No hay datos que analizar
6: else
7:     while longitud(M_r) or buscar(M_r,/n) do
8:         etq:
9:         if vacia(M_r) >0 then
10:            Continuar
11:            concatenar(reglas,M_r[posicion])
12:            Leer el arreglo produccion
13:        end if
14:        if simbolo == "=>" in produccion then
15:            reglas.avanza_indice() // se termina de anexar los caracteres anteriores y guarda
    la cadena concatenada en arreglo
16:        end if
```

Pseudocódigo C3 Analizador léxico

```
        if [produccion count] +1 == [ produccion count ]  then
            terminar
        else
            Volver a etq
        end if
    end while
end if
Fin
```

Los símbolos no terminales generalmente se encuentran antes del símbolo "=>" este símbolo se lee como produce, cualquier carácter hallado antes de un símbolo produce, se reconoce como no terminal, y es de igual forma una regla de producción. El pseudocódigo C3 indica que al instante de ejecutar el análisis de las reglas de producción previamente almacenadas en la primera mitad del pseudocódigo C3, se procede a concatenar todos los símbolos posteriores a "=>", estos símbolos son almacenados en un arreglo *reglas*. El ciclo de procesamiento termina hasta hallar el final del arreglo de cadenas. Establecido el pseudocódigo C3 se obtiene el conjunto de símbolos no terminales, posteriormente se procede a obtener el alfabeto de la gramática, para esto se ejecuta el pseudocódigo C4.

Pseudocódigo C4 Reglas de producción

```
Inicio
reglas, produccion2
while reglas do
    Eliminar todos los "=>" de reglas
    if  buscar(reglas, "=>")  then
        produccion2 = posicion(reglas) // Agregar regla de producción
    end if
    if numelem < [arreglocount] then
        exit //Romper el ciclo cuando se obtengan todas las reglas
    end if
end while
Fin
```

Para obtener los símbolos terminales, son implementados dos pasos, el primero está contenido en el pseudocódigo C4, en el cual se ejecuta la lectura del arreglo de cadenas, donde se obtienen todos los caracteres posteriores a "=>", de tal forma que el nuevo arreglo de cadenas tiene el mismo número de localidades que el arreglo reglas. Obtenido el nuevo arreglo que contiene las reglas de producción sin "=>" y los no terminales anteriores a él, se ejecuta con el pseudocódigo C5.

Pseudocódigo C5 Sustitución de símbolos en el arreglo

```
Inicio
arreglo, produccion, noterminal, alfabeto
if arreglo==produccion then // Verificar reglas de producción
    noterminal=arreglo //Sustituir posición actual
    Crear tokens en tokens(noterminal)
end if
while tokens.reglas do
    concatenar_tokens(alfabeto) // Concatenar símbolos de alfabeto
end while
Fin
```

Creado el arreglo alfabeto (“#”) de la gramática, se implementa el arreglo de cadenas denominado *comodin*, en el cual se reescribe la gramática en forma de una estructura de pila, siguiendo los pasos del pseudocódigo C6, este pseudocódigo implica la implementación de un analizador sintáctico, el cual es el encargado de analizar si las cadenas que recibe la gramática presentan la estructura descrita por las reglas de producción.

Pseudocódigo C6 Análisis sintáctico

```
1: Inicio
2: terminal, grammars
3: while reglas.fin do // Leer reglas de producción en ciclo
4:     if  es un terminal then
5:         grammars = concatenar(terminal) // concatenar
6:     end if
7:     if es noterminal then
8:         grammars=concatenar(“/+*/”) //Es regla de producción
9:     end if
10: end while
11: Fin
```

El nuevo arreglo generado por el pseudocódigo C6, presenta una sustitución de los elementos no terminales por el símbolo “/+*/”, los elementos que contienen la regla de producción son almacenados en una nueva localidad del arreglo de objetos, en la figura C1 se ejemplifica el pseudocódigo C6 utilizando parte de la gramática expuesta en la figura C1. En este proceso el analizador sintáctico comprueba las oraciones obtenidas que son obtenidas del archivo designado.

En la figura C1 se presentan algunos elementos de la figura C2, los cuales son introducidos en el lado izquierdo, los símbolos no terminales que especifican la regla de producción se convierten en “/+*/”, los demás símbolos pasan a ocupar una posición igual a la del arreglo anterior y sin cambios en su estructura.

S	“=>” No terminal se convierte en	/+*/
Numero	Se mantiene el símbolo	Numero
Numero	“=>” No terminal se convierte en	/+*/
D	Se mantiene el símbolo	D

Figura C2: Ejemplo de gramática para ser almacenada

Al generarse el arreglo denominado *comodin*, se emplea el método de evaluación de pertenencia de una cadena “C”, para ello se procede a ejecutar el pseudocódigo C7.

Pseudocódigo C7 Manejo de reglas

```
1: Inicio
2: Edoacep, reglas, reglas, comodin // Variables de inicio
3: Ubicar reglas en posicionactual( comodin, referencia(reglas)
4: pertenencia(Reglas_de_produccion)
5: Desplazar función carril()
6: while carril() == “|” or carril() == “/+*/” do
7:     posicion(reglas)
```

Pseudocódigo C7 Manejo de reglas

```
8: if terminal(reglas) then
9:     Concatenar
10: else
11:    Retornar posición del nuevo símbolo terminal a evaluar
12: end if
13: if la cadena a evaluar no termina aún then
14:    Regresar e incrementar una posición en comodin
15: else
16:    Continua
17: end if
18: if carrril()==Edoacep then // Al encontrar el final en función carril()
19:    estado_aceptacion
20: end if
21:
22: Fin
```

Si el método *carril*() ha retornado una cadena de terminales, se verifica en el método *evaluar*() la cadena obtenida por la gramática, comparándola con la cadena introducida previamente por el pseudocódigo C7, de esta forma se obtiene la respuesta si la cadena escrita pertenece o no pertenece al lenguaje, como se muestra en la figura C3.

S->Numero
Numero->DNumero
D-> 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

Figura C3: Ejemplo de gramática con 3 reglas de producción

En la figura C4 se ejemplifica el proceso de transformación de la gramática escrita en la figura C3.

Gramática original		Gramática equivalente
S	=>	/+*/
Numero	----	Numero
D	----	D
D	=>	/+*/
0	----	0
|	----	|
1	----	1
|	----	|
2	----	2
|	----	|
…	----	….

Figura C4: Proceso de transformación de gramática

La figura C5 contiene la cadena “efajk” para evaluación, del lado derecho se presenta la regla en curso que se está empleando para efectuar la operación de pertenencia al lenguaje, para este ejemplo se expone que la regla de producción S contiene los símbolos no terminales “efa”. El principio básico del método pila es sustituir los caracteres que correspondan a los de la pila de la cadena, lado izquierdo, con los de la pila de la gramática, lado derecho. Al final se sustituyen aquellos caracteres por el símbolo “#”, que ocupen la misma posición y que presente una igualdad en su morfología en ambas pilas, lo anterior representa a los caracteres de la cadena y se sustituye por un símbolo vacío denominado como “ø” a los caracteres que pertenezcan a la pila de la gramática, el número total de símbolos “#” y símbolos “ø” deben de coincidir entre cada pila.

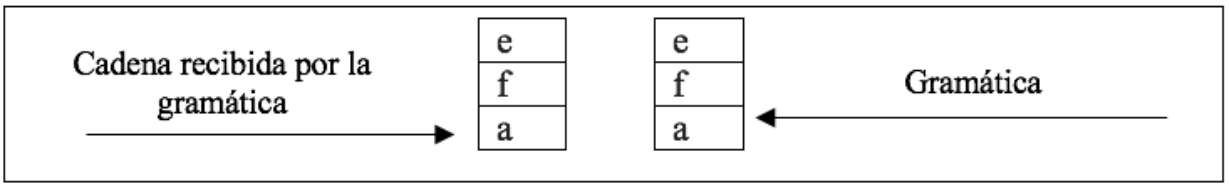


Figura C5: Implementación de una pila para evaluación de cadenas

Si la cadena introducida no corresponde con la gramática como se muestra en la figura C6, lado derecho, la cadena es rechazada y se dice que no pertenece al lenguaje, por otra parte, si la cadena llega al lado izquierdo se dice que la cadena ha sido aceptada.

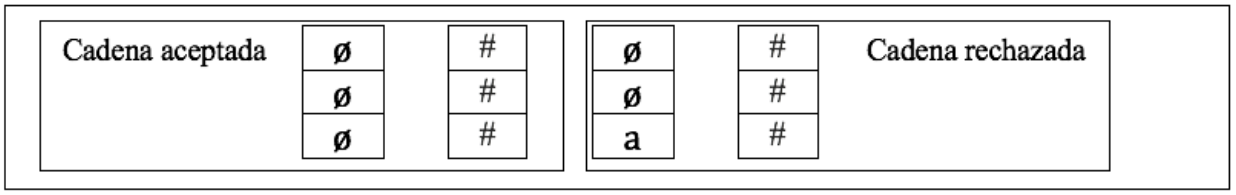


Figura C6: Estado de aceptación valido de la pila, lado izquierdo. Estado de aceptación invalido de la pila, lado derecho

En la figura C6, se visualiza como las dos pilas del lado derecho no contienen el mismo número de elementos “#” y “ø” por lo tanto la cadena introducida no pertenece al lenguaje, este resultado se obtiene inicializando un contador se verifica el número de símbolos “#” existen en una pila y el número de símbolos “ø” en la otra, si el número total de símbolos de cada pila es igual al tamaño de estas, se llega a un estado de aceptación favorable, como en el ejemplo de la figura C-8. Debido a que otros métodos de resolución de gramáticas, sugieren la expansión de la gramática a través del árbol sintáctico, en este apartado, se expone un proceso similar, pero sin utilizar un árbol sintáctico, se usa un arreglo variable (el número de elementos puede incrementar o disminuir en tiempo real) para conseguir la expansión cuando se encuentra un no terminal que contiene un terminal o varios de estos.

Al analizar la figura C7 se expone el proceso de expansión de una gramática para dar lugar a la generación de una respuesta a la cadena de entrada. Primero se apila el carácter “c”, posteriormente “b” y al final a (columna I), en la columna II, la regla S se sustituye el contenido a evaluar por aBC, posteriormente la columna III no varía en comparación con la columna I, mientras el no terminal B se sustituye por su terminal “b” (columna 4). Continuando con la expansión de la gramática, el no terminal C, se sustituye por “c”, y queda como se ejemplifica en la columna VI. El proceso final consiste al tener apilados los no terminales se comparan con la cadena de inicio, se empieza a generar una relación de igualdad entre el contenido de los índices de los arreglos, si son iguales cambian por los símbolos vacío. Al existir 3 vacíos y 3 “#” símbolos, se llega al estado de aceptación valido, como se visualiza en la figura C-8.

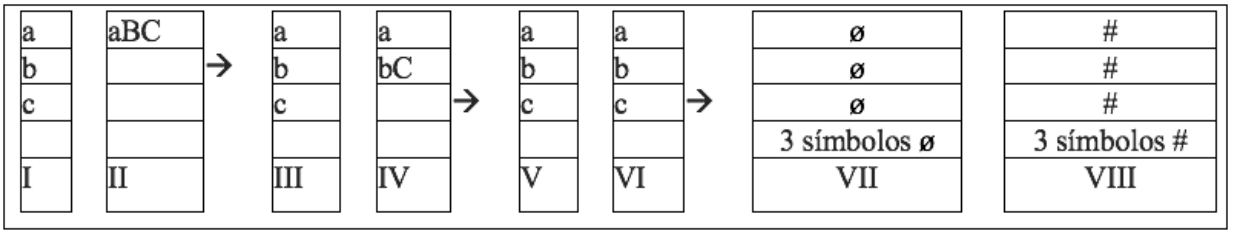


Figura C7: Ejemplo de expansión de una gramática utilizando el método pila