Artículo de Investigación

Desarrollo de un algoritmo esteganográfico utilizando números aleatorios

Development of a steganographic algorithm using random numbers

Henry Villa[1], Pablo Méndez[1*], Vanessa Zabala[2], Verónica Villa[2]

[1] *Universidad Nacional de Chimborazo, Riobamba, Ecuador, 060108; hvilla@unach.edu.ec*

2 Universidad Internacional de la Rioja, Madrid, España; 26006; vaneza_9109@hotmail.es; veros_alexandra@hotmail.com

*Correspondencia: pmendez@unach.edu.ec

Citación: Villa, H., Méndez, P., Zabala, V., & Villa, V., (2023). Desarrollo de un algoritmo esteganográfico utilizando números aleatorios. *Novasinergia. 6*(1). 120-135.

https://doi.org/10.37135/ns.01.11.08

Recibido: 15 febrero 2022
Aceptado: 27 septiembre 2022
Publicación: 16 enero 2023

Novasinergia
ISSN: 2631-2654

Resumen: El objetivo de la presente investigación es desarrollar un algoritmo esteganográfico basándose en la técnica LSB (Least Significant Bit), el filtro Canny Edge para la detección de bordes y la generación de números aleatorios para mejorar la seguridad de la información; para lo cual se compararon los indicadores de calidad de imagen Peak Signal to Noise Ratio (PSNR) y Mean Square Error (MSE) y se aplicaron las pruebas esteganográficas: RS-Attack, Visual Attack , Byte Attack, pruebas benckmark para comprobar la hipótesis y obtener las conclusiones. Para desarrollar la aplicación se utilizó: Java (lenguaje de programación), Netbeans (IDE), Beyond Compare (comparar hexadecimales), Guiffy ImageDiff (comparar diferencias entre pixeles de las imágenes), StegSecret (realizar estegoanálisis) y Digital Invisible Ink Toolkit (para el cálculo métricas de calidad de imagen). Para tal fin se desarrolló dos prototipos (Prototipo I: LSB y Canny Edge; Prototipo II: LSB, Canny Edge y números aleatorios) y se comparó los resultados obtenidos, dando como resultado que se genera menos ruido al implementar el uso de número aleatorios en el proceso esteganográfico.

Palabras clave: Algoritmo esteganográfico, canny edge, least significant bit, números aleatorios, tratamiento de imágenes.

Abstract: *This research aims to reduce noise in images by using steganography based on the LSB (Least Significant Bit) technique, the Canny Edge filter for edge detection, and the generation of random numbers. For this purpose, several indicators were compared, and various statistical tests were applied to test the hypothesis and draw conclusions. To develop the application, we used: Java (programming language), Netbeans (IDE), Beyond Compare (compare hexadecimals), Guiffy ImageDiff (compare differences between pixels in images), StegSecret (perform steganalysis), and Digital Invisible Ink Toolkit (to the calculation of image quality metrics: PSNR and MSE). For this purpose, two prototypes were developed (Prototype I: LSB and Canny Edge; Prototype II: LSB, Canny Edge, and random numbers). The results obtained were compared, resulting in less noise generated when implementing random numbers in the steganographic process.*

Keywords*: Steganographic algorithm, canny edge, least significant bit, random numbers, image processing.*

1. Introducción

La esteganografía es el arte y ciencia de ocultar información dentro de una imagen sin que se altere de manera aparente la imagen, ni el mensaje a transmitir, así como evitando la visualización simple de la información (Saini & Harsh, 2013).

El término esteganografía proviene de dos vocablos griegos: "steganos" (cubierto) y "graphos" (escritura), por tanto, es la ciencia que estudia los métodos y técnicas para ocultar la información a través de medios multimedia (documento, imagen, audio, video) con el fin que el mensaje oculto pase inadvertido. La esteganografía es muy útil al transmitir mensajes a través de medios inseguros, ya que únicamente el receptor del mensaje será quien debe extraer el mensaje.

El propósito de la esteganografía es ocultar y engañar. Es una forma de comunicación encubierta y puede implicar el uso de cualquier medio para ocultar mensajes. No es una forma de criptografía, porque no implica codificar datos o usar una clave. En cambio, es una forma de ocultar datos y se puede ejecutar de manera inteligente (Ayudaley, 2021).

Al realizar el proceso esteganográfico, el medio digital resultante genera ruido que hace posible detectar fácilmente que dicho medio ha sido manipulado para ocultar información.

El objetivo de la presente investigación es desarrollar un algoritmo esteganográfico basándose en la técnica LSB (Least Significant Bit), el filtro Canny Edge para la detección de bordes y la generación de números aleatorios para mejorar la seguridad de la información, especialmente al momento de transferir información a través de medios inseguros, ya que, al realizar el proceso esteganográfico, usando el algoritmo propuesto, la imagen resultante genera menos ruido y por ende es indetectable. También según Kaspersky (2021), la esteganografía puede usarse para proteger un archivo digital de la copia ilegal, como por ejemplo a través del uso de marcas de agua.

1.1. Aspectos generales

En la presente sección se abordan aspectos relacionados con: Técnicas esteganográficas, Técnica LSB, Algoritmo Canny Edge, Número aleatorios y Revisión del estado del arte., mismos que son detallados a continuación:

Técnicas esteganográficas

Considerando que el presente trabajo se centra en el tratamiento de imágenes, se profundiza en las técnicas esteganográficas orientadas a este medio digital. Las técnicas más utilizadas para realizar esteganografía en imágenes son (López, 2021):

- Least Significant Bit (LSB): Consiste en usar el bit menos significativo de los pixeles de una imagen y alterarlo con la información a ocultar.
- Enmascaramiento y filtrado: La información se oculta de una imagen empleando marcas de agua que incluyen información.
- Algoritmos y transformaciones: Se basa en el uso de funciones matemáticas que se usan en la compresión de datos.
- Entre otras como, por ejemplo: codificación de patrón redundante, cifrar y dispersar, codificación y transformación del coseno.

La presente investigación se centra en la técnica esteganográfica LSB, ya que, no altera el tamaño del archivo portador, no utiliza muchos recursos del computador, es rápido, es sencilla su implementación y la alteración de la imagen, en referencia al color, es mínima. Además, se ha combinado con el algoritmo de filtrado Canny Edge y con la generación de número aleatorios para mejorar la seguridad de la información.

Técnica LSB

La técnica LSB, no genera dichos problemas, siendo ese el motivo por el cuál es una de las más utilizadas debido a que no altera el tamaño del archivo portador (imagen donde embebe la información).

Las principales características que representa el uso de la técnica LSB son la siguientes:

- No altera el tamaño del archivo portador.
- No utiliza muchos recursos del computador, por tanto, es rápido.
- Es sencilla su implementación.
- La alteración de la imagen, en referencia al color, es mínima.

El formato de imagen más compatible con la técnica LSB es el mapa de bits (BMP), debido a que generalmente tiene mayor resolución y no está comprimida como formatos comúnmente usados. La única desventaja que se puede considerar es que el tamaño y resolución de la imagen determina la longitud del mensaje que se puede embeber, es decir, mientras más grande (tamaño y resolución) es posible embeber un mensaje más largo.

Las siguientes ecuaciones verifican la relación mencionada para calcular el tamaño mínimo de la imagen considerando el mensaje y viceversa:

$$\begin{aligned} TamañoImg &= NumMsj * 8 * 3 \\ NumMsj &= \frac{LargoImg * AnchoImg}{3} \end{aligned} \quad (1)$$

Donde:

- $NumMsj$ = cantidad de caracteres del mensaje
- $TamañoImg$ = tamaño mínimo de la imagen (Resolución)
- $LargoImg$ = Largo de la imagen
- $AnchoImg$ = Ancho de la imagen

Además, se considera que 1 carácter tiene 8 bits (1 byte) y que se necesita 3 pixeles para realizar el proceso esteganográfico para los caracteres.

Algoritmo Canny Edge

De acuerdo con Suárez y Villavicencio (2017), "Los bordes se definen, en términos de procesamiento de imágenes digitales, como los lugares donde se produce un fuerte cambio de intensidad. Las técnicas de detección de bordes se requieren a menudo en diferentes tareas de procesamiento de imágenes y visión por ordenador, para la segmentación de imágenes, el reconocimiento de patrones, preservar importantes propiedades estructurales, entre otros. Estas tareas son aplicadas a áreas tales como la teledetección, la medicina, entre otras."

Canny Edge es un algoritmo muy popular, que se utiliza en la detección de bordes dentro de una imagen, se caracteriza porque detecta los bordes con gran precisión y en la simplicidad de su funcionamiento e implementación. Esencialmente en el algoritmo de Canny Edge se realizan cuatro pasos (OpenCV, 2021):

- Filtrado Gaussiano: al aplicar este filtrado a la imagen ayuda a reducir el ruido, debido a que al momento de realizar el proceso de detección de bordes es más susceptible a que se introduzca el ruido. Generalmente se usa un filtro Gaussiano de 5x5 o de 3x3 (Kohei, Yasuaki, & Koji, 2010).
- Filtrado de Sobel: posterior a obtener la imagen suavizada se aplica este filtro que ayuda a obtener la magnitud y dirección de la gradiente de intensidad de la imagen.
- Supresión de los no máximos: posterior a obtener la magnitud y dirección de la gradiente, se realiza un tipo escaneo de la imagen para eliminar los pixeles que no constituyen el borde. Con ese fin, en cada píxel se comprueba si es el máximo localmente dentro del grupo en dirección de la gradiente de la imagen.
- Umbralización de histéresis: en esta etapa se establece cuáles son los puntos o pixeles que realmente pertenecen al borde y los que no. Para tal fin, se establece 2 valores uno máximo y uno mínimo, y se procede a comprar cada punto, de tal manera que los puntos con la gradiente superior al valor máximo serán los bordes, mientras que los que se encuentran por debajo del valor mínimo con seguridad no son los bordes.

Números aleatorios

La generación de números aleatorios permite difuminar de mejor forma la información dentro de la imagen en el proceso esteganográfico.

Dentro de los diferentes tipos de generadores de números aleatorios existentes, el tipo que se usa mayormente por su fiabilidad y características son los de computación digital, debido a que implementan un algoritmo con base matemática y determinista. Por tanto, esto los convierte en generador de números pseudoaleatorios (Pérez, 2019).

Dentro de esta clasificación, los métodos de congruencia son lo más usados ya que, se determina la manera de obtener la serie de números en función del anterior, es decir, utiliza una forma de recursión para la generación de números aleatorios siendo:

$$x_{n+1} = (a\, x_n + b) \bmod m \tag{2}$$

Revisión del estado del arte

Hasta la fecha se han realizado diversas investigaciones en cada uno de los campos: esteganografía, números aleatorios, LSB, filtros de imágenes y algoritmo Canny Edge, algunas de manera individual y otras de manera combinatoria. Entre los trabajos más relevantes se tiene:

- Estudio basado en el algoritmo Canny Edge conjuntamente del operador Sobel para detectar los bordes, posteriormente utiliza la esteganografía específicamente el algoritmo LSB. Pero el trabajo se enfoca, en el reemplazo de los bordes más suaves o que generen menos ruido y así que el mensaje sea menos detectable. Pero esta detección de bordes lo hace una manera contínua de arriba abajo y de izquierda a derecha; no

utiliza un esquema para que sea aleatorio el reemplazo y la técnica esteganográfica sea indetectable, más bien se enfoca en el suavizado de bordes (Jumanto & Setiadi, 2018).

- Investigación que implementa el algoritmo de Canny Edge, pero usando el operador Sobel (cálculo de borde a través una aproximación al gradiente de la función de intensidad de una imagen) que permite hacer el análisis de gradientes para detectar los bordes, posterior a encontrado el borde utiliza el algoritmo LSB para embeber el mensaje en la imagen, pero para reemplazar el último bit significativo lo hace en orden; en el primer pixel reemplaza el último bit, luego el segundo, luego el tercero y así sucesivamente. Por tanto, permitiría obtener el mensaje más rápido (Barnali & Samir, 2015).
- Estudio que combina la esteganografía y la criptografía y está orientado al intercambio de mensajes a través de vías inseguras. En cuando a la esteganografía se observa la modificación del funcionamiento del algoritmo LSB juntamente con el algoritmo Canny Edge, pero únicamente detecta el borde, calcula un salto a priori y en los pixeles detectados utiliza la esteganografía (LSB). En cuanto a la criptografía luego de obtener los resultados se cifra con el algoritmo simétrico Advanced Encryption Standard (AES) para mejorar la seguridad de la información (Méndez, Villa, & Cisneros, 2017).
- Trabajo que combina la esteganografía y criptografía; referente a la esteganografía incorpora el uso del algoritmo LSB y usa el filtro Canny Edge, pero únicamente detecta el borde que se va a utilizar para el proceso de esteganografía el algoritmo LSB sin considerar ningún tipo de salto y referente a la criptografía se utiliza el algoritmo simétrico de cifrado Data Encryption Standard (DES), que por naturaleza es inseguro. Y brinda muchas herramientas y técnicas libres que se pueden usar en la esteganografía y criptografía (Kusuma, Indriani, Sari, Rachmawanto, & Setiadi, 2017).
- Estudio que detecta los bordes utilizando el algoritmo Canny Edge, pero al momento de utilizar el algoritmo LSB se basa en la lógica de reemplazo del bit menos significativo realizando una operación de XOR entre los bits del mensaje con los bits del borde detectados con el algoritmo LSB, pero al hacer ese tipo de operaciones sigue una lógica muy deducible y, por tanto, es fácil detectar que tiene embebido un mensaje (Gaurav & Ghanekar, 2018).

Por defecto las investigaciones que utilizan el algoritmo de filtrado Canny Edge insertan la información de forma secuencial en la imagen, sin embargo, el objetivo de la presente investigación es cambiar la forma en la que se realiza el proceso esteganográfico utilizando números aleatorios para determinar las posiciones en las cuales se inserte la información de forma difusa con la técnica de LSB.

2. Metodología

Con base a los antecedentes de investigación mencionados en la sección anterior, a continuación, detallamos los pasos seguidos de la metodología para la presente investigación:

- Determinar e implementar la técnica esteganográfica: Seleccionar la técnica esteganográfica más adecuada para la investigación.
- Desarrollo de la propuesta del algoritmo esteganográfico: Establecer los lineamientos para la propuesta de algoritmo esteganográfico.

- Establecer el escenario de pruebas y prototipos: Desarrollar los prototipos de experimentación y ejecutarlos en el ambiente de pruebas para la recolección de datos.

Las herramientas que permitieron desarrollar y recopilar la información de las pruebas de los 2 prototipos son las siguientes:

- Netbeans: se usa como entorno de desarrollo integrado (IDE), mismo que permite el desarrollo en algunos lenguajes de programación como Java, PHP, C++, entre otros. La ventaja de este IDE es que es de código abierto ya que tiene la licencia CDDL, GPL2 (Netbeans, 2022).
- Java: es un lenguaje de programación muy popular en el mundo según el índice TIOBE, mismo que permite crear aplicaciones de diferente tipo, como, por ejemplo: sitios web, aplicaciones de desktop, aplicaciones móviles, entre otras. Además, este lenguaje permite crear aplicaciones multiplataforma (TIOBE, 2022).
- Beyond Compare: específicamente es un editor hexadecimal y esta herramienta permite insertar, modificar, eliminar y ver los datos hexadecimales de los archivos (Scooter Software, 2022).
- Guiffy ImageDiff: herramienta de código abierto que permite realizar la comparación entre imágenes píxel a píxel para poder detectar diferencias (Software Guiffy, 2022).
- StegSecret: es una herramienta de código abierto para realizar estegoanálisis de las imágenes resultantes del proceso esteganográfico (Muñoz, 2007).
- Digital Invisible Ink Toolkit: permite realizar pruebas de benchmark, mismos que permiten determinar el PSNR y MSE de las imágenes después del proceso esteganográfico (Hempstalk, 2022).

Se ha establecido como base fundamental para la investigación las siguientes técnicas/algoritmos:

- Técnica esteganográfica LSB: debido a su versatilidad, rapidez, fácil implementación, distorsión mínima, y no altera el tamaño. Consiste en sustituir parte de la información en un determinado píxel (bit menos significativo) con la información de los datos de la imagen, de hecho, se debe sumar o restar un 1 al valor de cada componente del píxel (RGB = Red Green Blue), al realizar este proceso la información es casi indetectable por cualquier persona a simple vista (Mouse, 2011).
- Algoritmo Canny Edge: es un algoritmo muy popular, que se utiliza en la detección de bordes dentro de una imagen, se caracteriza porque detecta los bordes con gran precisión y en la simplicidad de su funcionamiento e implementación. Esencialmente en el algoritmo de Canny Edge se realizan cuatro pasos que son: Filtrado Gaussiano, Filtrado de Sobel, Supresión de los no máximos, Umbralización de histéresis (Kohei, Yasuaki, & Koji, 2010).
- Generación de números aleatorios: Los generadores de números aleatorios se usan para crear secuencias de números sin un orden específico en base a un algoritmo matemático y determinista (Pérez, 2019).

El nuevo algoritmo propuesto se basa en la combinación del algoritmo Canny Edge para encontrar los píxeles de los bordes de la imagen portadora, a partir de ahí se los selecciona mediante la aplicación de un generador de números aleatorios y a los píxeles seleccionados

se aplica la técnica LSB para ocultar la información en la imagen, dicho proceso se implementa en el Prototipo II, tal como se puede observar en la Figura 2. De esa manera la información oculta en la imagen esteganografiada es imperceptible al ojo humano.

Es importante mencionar que al utilizar los píxeles de los bordes para hacer el LSB no genera un impacto visual significativo el cambio, ya que son los menos significativos en la imagen.

Además, de manera complementaria se genera un archivo XML que almacena los píxeles que han sido reemplazados con el LSB y mismo que permite recuperar la información oculta en la imagen esteganografiada.

Se define un ambiente de pruebas común, en los que se ejecutará los prototipos con el fin de tomar los datos necesario y así demostrar la hipótesis.

Las condiciones del ambiente de pruebas son las siguientes:

- Mensaje: “Hola mundo”
- Tipo de imagen: Tipo mapa de bits (BMP) de dimensiones
- Dimensiones de la imagen: 640 x 426 píxeles.
- Generador de números aleatorios: Se considera un generador computacional que utiliza el método de congruencia mixto específicamente el caso de $m = 2^k$ (m potencia de 2) (Pérez, 2019).
- Generador de número aleatorios: $x_{n+1} = (5 * x_n + 5) \bmod 16384,\ x_0 = 1$ (Pérez, 2019).

Se considera el desarrollo de dos prototipos, mismos que permitirán realizar una comparación entre los resultados de la imagen producto de la esteganografía y determinar cuál genera menos ruido. Cada prototipo se determina considerando:

- Prototipo I: el diagrama del Prototipo I utilizando: LSB y Canny Edge se muestra en la Figura 1.

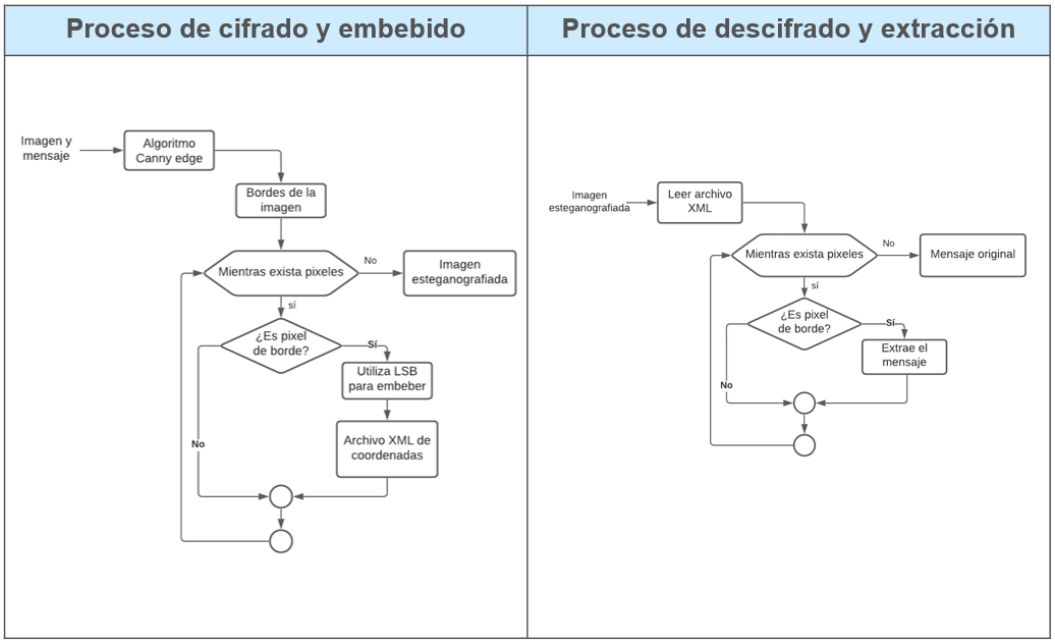

Figura 1: Algoritmo del Prototipo I.

- Prototipo II: el diagrama del prototipo II utilizando: LSB, Canny Edge y números aleatorios, se muestra en la Figura 2.

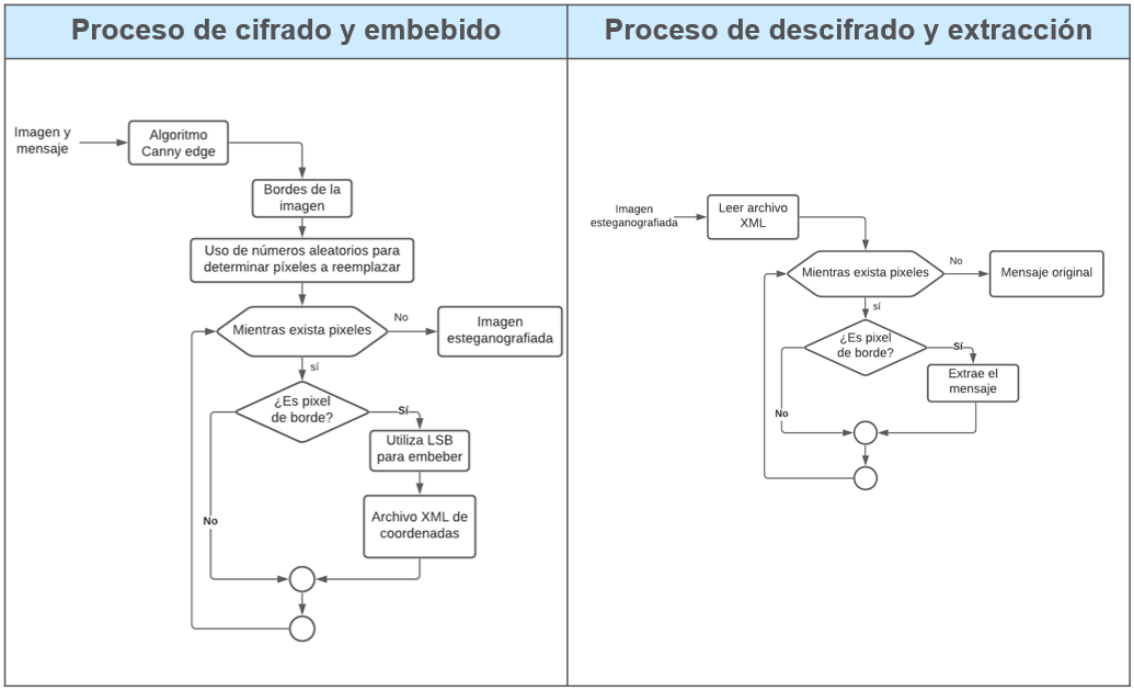

Figura 2: Algoritmo del prototipo II

El proceso para realizar la propuesta de algoritmo esteganográfico en los prototipos se describe a continuación:

Prototipo I

- Ingreso de la imagen: se lee y carga la imagen en el programa
- Ingreso del texto: lee el texto y transforma en binario
- Ejecución del algoritmo Canny Edge: detecta los bordes de la imagen
- LSB en pixeles: Aplica la técnica LSB para embeber la información en los pixeles
- Archivo de coordenadas XML: almacena las posiciones en las que se distribuye el mensaje para el embebido o extracción.

Prototipo II

- Ingreso de la imagen: se lee y carga la imagen en el programa
- Ingreso del texto: lee el texto y transforma en binario
- Ejecución del algoritmo Canny Edge: detecta los bordes de la imagen
- Generación de números aleatorios: determina de forma aleatoria las posiciones en las que se colocan los elementos del mensaje
- LSB en pixeles: Aplica la técnica LSB para embeber la información en los pixeles
- Archivo de coordenadas XML: almacena las posiciones en las que se distribuye el mensaje para el embebido o extracción

Para cada prototipo se desarrolla una aplicación de escritorio utilizando el IDE NetBeans y el lenguaje de programación de Java, en la cual se ha creado una misma estructura, considerando los escenarios para el Prototipo I y II.

3. Resultados

En la Figura 3 se muestra la imagen original “paisaje.bmp” usada para embeber el mensaje, mientras que en la Figura 4 y 5 se muestran las imágenes con el mensaje embebido usando los Prototipos I y II respectivamente.

Figura 3: Imagen original “paisaje.bmp”.

Figura 4: Imagen generada con el Prototipo I.

Figura 5: Imagen generada con el Prototipo II.

Se pudo apreciar la imagen original en comparación con las generadas con los Prototipos I y II no presentan diferencias a simple vista.

Posteriormente, se comparó el código hexadecimal con en el programa Beyond Compare de las imágenes embebidas con los Prototipos I y II mostrando las diferencias entre ellas en las Figuras 6 y 7, respectivamente.

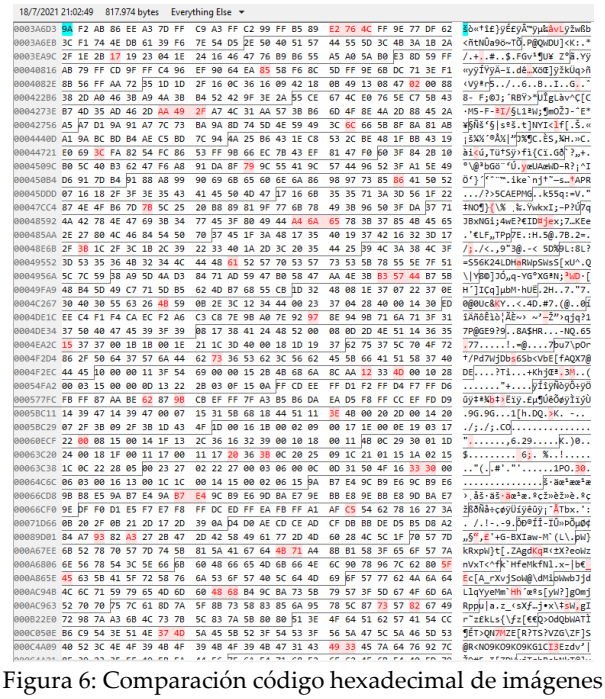

Figura 6: Comparación código hexadecimal de imágenes esteganografiadas con el Prototipo I.

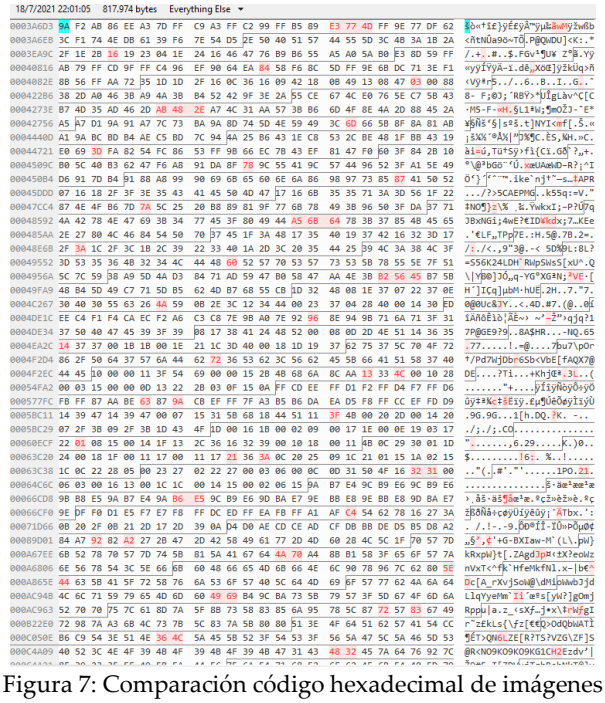

Figura 7: Comparación código hexadecimal de imágenes esteganografiadas con el Prototipo II.

Se aprecia que existe diferencias entre las dos imágenes embebidas en su código hexadecimal de las imágenes embebidas por el Prototipo I y II.

Se realiza la comparación entre la imagen esteganografiada por cada prototipo con el fin de establecer los píxeles diferentes que contienen la información embebida en la imagen. En las Figuras 8 y 9 se observan los pixeles modificados en relación con la imagen original.

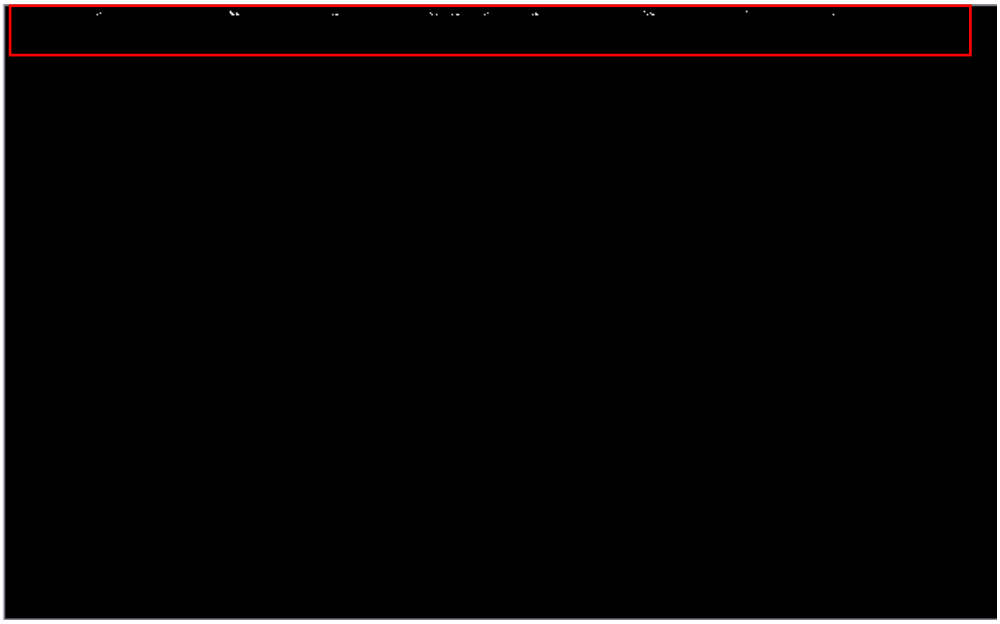

Figura 8: Comparación píxel a píxel de la imagen sin fondo Prototipo I.

Figura 9: Comparación píxel a píxel de la imagen sin fondo Prototipo II.

Se pudo observar que los píxeles del Prototipo I, están reemplazados de manera ordenada en la parte superior de la imagen en las primeras filas, pero en el Prototipo II por el uso de los números aleatorios, los píxeles están distribuidos. Esto dificulta detectar el mensaje embebido, ya que como se visualiza éste está más disperso en toda la imagen.

A continuación, se realizan pruebas de estegoanálisis que permiten aplicar estrategias y procedimientos para la detección, identificación y análisis de información oculta en imágenes, sonidos y canales encubiertos (Isaza, Espinosa, & Ocampo, 2006).

Mediante la herramienta StegSecret se realiza el estegoanálisis de las imágenes resultantes del proceso esteganográfico de los prototipos comparada con la original se obtiene los resultados que se muestra en la Tabla 1 utilizando los algoritmos que detectan información oculta con técnicas LSB en pixeles elegidos de forma secuencial y/o pseudoaleatoria RS-Attack y Visual Attack analizando la alteración de las propiedades estadísticas de los pixeles en los componentes RGB (Red-Green-Blue) (Aziz, Najaran, & Afsar, 2015).

Tabla 1: Estegoanálisis de RS-Attack y de Visual Attack

Prototipo I	Prototipo II
RS ANALYSIS	RS ANALYSIS
RS Analysis (Non-overlapping groups)	RS Analysis (Non-overlapping groups)
Percentage in red: 6.21293	Percentage in red: 6.20431
Approximate length (in bytes) from red: 6352.09582	Approximate length (in bytes) from red: 6343.28904
Percentage in green: 6.23382	Percentage in green: 6.27018
Approximate length (in bytes) from green: 6373.4601	Approximate length (in bytes) from green: 6410.63022
Percentage in blue: 39.36015	Percentage in blue: 39.34596
Approximate length (in bytes) from blue: 40241.8187	Approximate length (in bytes) from blue: 40227.31065
RS Analysis (Overlapping groups)	RS Analysis (Overlapping groups)
Percentage in red: 7.58482	Percentage in red: 7.5698
Approximate length (in bytes) from red: 7754.72473	Approximate length (in bytes) from red: 7739.36762
Percentage in green: 4.90749	Percentage in green: 4.90059
Approximate length (in bytes) from green: 5017.41889	Approximate length (in bytes) from green: 5010.3591
Percentage in blue: 39.62645	Percentage in blue: 39.60559
Approximate length (in bytes) from blue: 40514.08278	Approximate length (in bytes) from blue: 40492.7553
Average across all groups/colours: 17.32094	Average across all groups/colours: 17.31607
Average approximate length across all groups/colours: 17708.93351	Average approximate length across all groups/colours: 17703.95199

Se puede notar que al realizar el análisis usando la correlación por rangos de Spearman para detectar si hay información oculta, de manera general y por cada componente del color, se nota que usando el Prototipo II presenta una mejora en relación con el Prototipo I.

Se realiza un estegoanálisis usando la técnica del byte attack para establecer diferencias que el ojo humano aprecia y a través de esta técnica deben ser más evidentes, como se muestra en la Figura 10 y 11 (Aziz, Najaran, & Afsar, 2015).

Figura 10: Estegoanálisis de Visual Attack / Byte Attack (Byte 7) con el Prototipo I.

Figura 11: Estegoanálisis de Visual Attack / Byte Attack (Byte 7) con el Prototipo II.

Para calcular las métricas que demuestran la calidad de la imagen, se utiliza los indicadores:

- Mean Square Error (MSE): ayuda a determinar el error cuadrático medio entre la imagen original y la portadora del mensaje (esteganografiada).
  Un valor bajo de MSE significa un error menor entre las dos imágenes, cuanto menor sea el valor de MSE, menor será el error (Kamaldeep, Rajkumar, & Sachin, 2016).
  La fórmula matemática para su cálculo es:

$$MSE = \frac{1}{M*N}\sum_{i=1}^{M*N}(o_i - r_i)^2. \quad (3)$$

  Donde:
  $M * N$ =número de filas y columnas de la imagen
  o_i = imagen original
  r_i = imagen esteganografiada

- Peak Signal to Noise Ratio (PSNR): ayuda a medir la proporción máxima de señal a ruido de los datos embebidos en la imagen. Un valor al alto del PSNR indica que la calidad de la imagen es buena, además que la codificación que se usa es la mejor. Si se obtiene un método de reducción de ruido en imágenes con un valor de MSE bajo y un valor de PSNR alto, se reconoce como el de mejor desempeño (Rosas, 2006).

  Matemáticamente se calcula basándose en la medida del Mean Square Error (MSE), mediante la siguiente fórmula:

$$PSNR = 10 * \log_{10}\frac{MAX_I^2}{MSE} \quad (4)$$

  Donde:
  MAX =máxima cantidad de filas y columnas en la imagen.
  MSE = Mean Square Error.

Estas pruebas se las realiza usando la herramienta Digital Invisible Ink Toolkit, entre la imagen original y la imagen esteganografiada de los 2 prototipos, los resultados obtenidos se pueden observar en la Tabla 2.

Tabla 2: *Estegoanálisis Pruebas Benchmark.*

Prototipo I	Prototipo II
Results of benchmark tests ========== Average Absolute Difference: 1.943955399061033E-4 Mean Squared Error: 3.924589201877934E-4 LpNorm: 1.962294600938967E-4 Laplacian Mean Squared Error: 1.0705866962985329E-7 Signal to Noise Ratio: 3.517512540467289 7E8 Peak Signal to Noise Ratio: 1.483388375327103E9 Normalised Cross-Correlation: 0.9999999890480363 Correlation Quality: 153.00653869076416	Results of benchmark tests ========== Average Absolute Difference: 2.01731220657277E-4 Mean Squared Error: 3.851232394366197E-4 LpNorm: 1.9256161971830985E-4 Laplacian Mean Squared Error: 1.0089133120213648E-7 Signal to Noise Ratio: 3.5845127793333334E8 Peak Signal to Noise Ratio: 1.511643392E9 Normalised Cross-Correlation: 0.9999999218151486 Correlation Quality: 153.0065284036926

Para almacenar las posiciones generadas con los números aleatorios, se crea un archivo XML mismo que permitirá extraer la información esteganografiada. En las Figuras 12 y 13 se

muestran los archivos XML que contienen las coordenadas de los pixeles que fueron modificados por el LSB en los 2 prototipos.

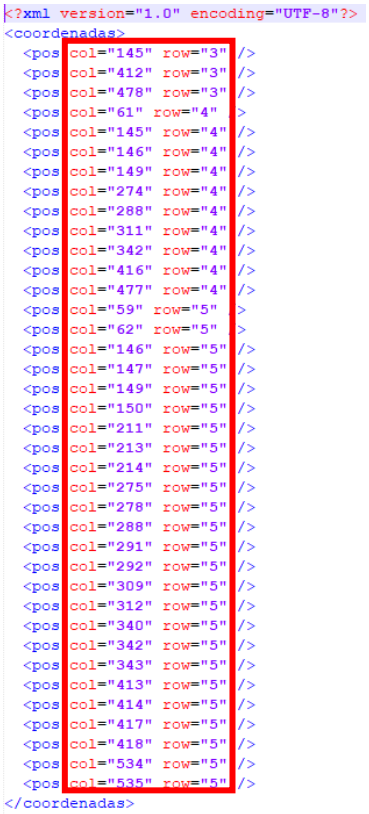

Figura 12: Comparación de archivo de coordenadas de píxeles con el Prototipo I.

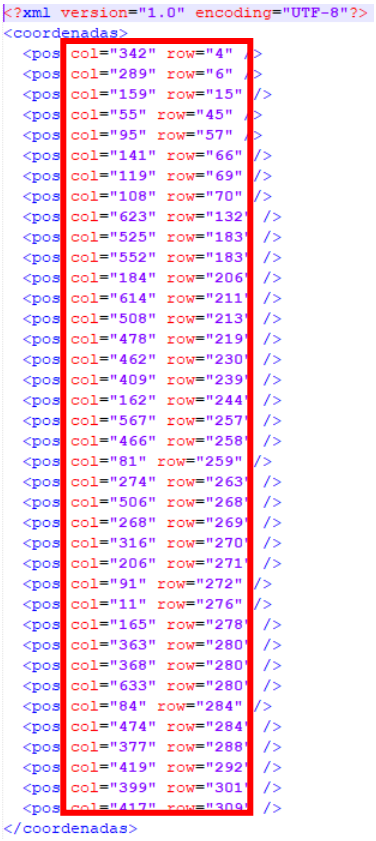

Figura 13: Comparación de archivo de coordenadas de píxeles con el Prototipo II.

4. Discusión

En la presente investigación se evaluó la técnica esteganográfica LSB y el filtro Canny Edge para la detección de borden y la generación de números aleatorios para mejorar la seguridad de la información. Los resultados obtenidos en base a las pruebas de estegoanálisis realizadas demuestran que en el Prototipo II propuesto existe una mejor calidad de imagen (PSNR alto) y menor ruido (MSE bajo) demuestran que la propuesta realizada dispersa de mejor forma la información en toda la imagen sin generar alteraciones que sean fácilmente visualizadas.

A diferencia de otras investigaciones realizadas anteriormente que embeben la información de forma secuencial, la propuesta del algoritmo del Prototipo II modifica este patrón para ocultar la información de forma aleatoria en cada ocasión, dificultando posibles vulneraciones para determinar el mensaje original por personas no autorizadas.

Los archivos XML generados contienen las coordenadas de los pixeles estenografiados, en el Prototipo I se reemplaza cada píxel ordenadamente de izquierda a derecha y de arriba abajo, pero en el Prototipo II, por lo que, al utilizar la generación de números aleatorios cada píxel tiene distinta posición, y eso permite que la información sea más difícil de detectar y extraer por parte de personas no autorizadas.

Los resultados de esta investigación constituyen una base para futuras investigaciones relacionadas con la esteganografía en imágenes. Como líneas de trabajo futuro se propone

combinarlo con algoritmos criptográficos que cifren la información antes de ocultarla. Propuestas de nuevos métodos para determinar las posiciones en las que se embebe la información en la imagen. Propuestas de nuevas herramientas para estegoanálisis y determinación de las métricas para la calidad de la imagen.

5. Conclusiones

Al usar el algoritmo propuesto, usando números aleatorios la imagen esteganografiada contiene la información más dispersa, manteniendo el tamaño, calidad y proporción de la imagen original; ayudando a que los cambios sean imperceptibles a los sentidos humanos.

Si se desea ocultar mensajes grandes se debe utilizar imágenes de mayor tamaño y calidad, ya que, el tamaño del mensaje es directamente proporcional al tamaño de la imagen.

La medida del Mean Square Error (MSE) de la imagen esteganografiada en el Prototipo II, posee un menor error cuadrático medio con un valor de 3.85 frente al valor de 3.92 obtenido en el Prototipo I, por lo tanto, se puede concluir que el algoritmo propuesto tiene un mejor desempeño en reducción de ruido al realizar la esteganografía de las imágenes.

La medida del Peak Signal to Noise Ratio (PSNR) de la imagen esteganografiada en el Prototipo II, posee un menor mayor valor con 1.51 con relación a 1.28 obtenido por el Prototipo I, por lo tanto, la calidad de la imagen en buena.

En base a los resultados de las métricas de calidad de la imagen, al obtener un valor bajo en MSE y alto en PSNR, se puede verificar que la propuesta del algoritmo del Prototipo II posee un mejor rendimiento en comparación con el Prototipo I desde el punto de vista esteganográfico.

Las coordenadas en el archivo XML, oculta la información en la imagen con el Prototipo I que se generan de forma secuencial, sin embargo, con el Prototipo II, al utilizar números aleatorios los pixeles tienen distintas posiciones, por lo que la información se oculta de forma más dispersa.

Contribuciones de los autores

	Villa, H.	Méndez, P.	Zabala, V.	Villa, V.
Conceptualización		■		
Análisis formal	■			
Investigación	■	■	■	■
Metodología	■			
Recursos	■	■	■	■
Validación			■	■
Redacción – revisión y edición	■	■	■	■

Conflicto de Interés

Los autores declaran que no existen conflictos de interés de naturaleza alguna con la presente investigación.

Referencias

Ayudaley. (2021). Ayudaley. Obtenido de https://ayudaleyprotecciondatos.es/2021/03/17/esteganografia/#Como_funciona_la_esteganografia_digital

Aziz, M., Najaran, M., & Afsar, M. (2015). A cycling chaos-based cryptic-free algorithm for image steganography.

Barnali, B., & Samir, B. (2015). An image steganography method on edge detection using multiple LSB modification technique. Journal of Basic and Applied Research International (págs. 75-80). Calcutta, India: International Knowledge Press. Obtenido de https://www.researchgate.net/publication/328611029_AN_IMAGE_STEGANOGRAPHY_METHOD_ON_EDGE_DETECTION_USING_MULTIPLE_LSB_MODIFICATION_TECHNIQUE/stats

Gaurav, K., & Ghanekar, U. (2018). Image steganography based on Canny edge detection, dilation operator and hybrid coding. Journal of Information Security and Applications (págs. 41-51). Kurukshetra, India: ScienceDirect. doi:https://doi.org/10.1016/j.jisa.2018.05.001

Hempstalk, K. (2022). Sorceforge. Obtenido de http://diit.sourceforge.net/

Isaza, G., Espinosa, C., & Ocampo, S. (2006). Análisis de técnicas esécnicas esécnicas esécnicas esécnicas esttttegegegegeganoganoganoganoganogrrrrráfáfáfáfáficas y esicas y esicas y esicas y esicas y esttttegoanálisis en canalesegoanálisis en canalesegoanálisis en canalesegoanálisis en canalesegoanálisis en canalesegoa. Vector, 29-38.

Jumanto, I., & Setiadi, M. (2018). An Enhanced LSB-Image Steganography Using the Hybrid Canny-Sobel Edge Detection. Cybernetics and Information Technologies (págs. 74-88). Sofía, Bulgaria: Institute of Information and Communication Technologies of Bulgarian Academy of Sciences. doi:https://doi.org/10.2478/cait-2018-0029

Kamaldeep, J., Rajkumar, Y., & Sachin, A. (2016). PSNR and MSE based investigation of LSB. 2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT) (págs. 505-509). India: IEEE. doi:10.1109/ICCTICT.2016.7514593

Kaspersky. (2021). Kaspersky. Obtenido de https://www.kaspersky.es/blog/digital-steganography/18791/

Kohei, O., Yasuaki, I., & Koji, N. (2010). Efficient Canny Edge Detection using a GPU. First International Conference on Networking and Computing (págs. 279-280). Hiroshima: Hiroshima University. doi:10.1109/IC-NC.2010.13

Kusuma, E., Indriani, O., Sari, C., Rachmawanto, E., & Setiadi, D. (2017). An imperceptible LSB image hiding on edge region using DES encryption. International Conference on Innovative and Creative Information Technology (ICITech) (págs. 1-6). Salatiga, Indonesia: IEEE. doi:10.1109/INNOCIT.2017.8319132

López, M. (2021). UnoCero. Obtenido de https://www.unocero.com/noticias/esteganografia-para-cifrar-mensajes-en-imagenes/

Méndez, P., Villa, H., & Cisneros, S. (2017). Nuevo algoritmo para la detección de bordes en imágenes para esteganografía. Maskana, 17-29. Obtenido de https://publicaciones.ucuenca.edu.ec/ojs/index.php/maskana/article/view/1449

Mouse, J. (2011). Obtenido de https://www.jc-mouse.net/ingenieria-de-sistemas/esteganografia-lsb-en-java-proyecto-completo

Muñoz, A. (2007). Obtenido de http://stegsecret.sourceforge.net/indexS.html

Netbeans. (2022). Obtenido de https://netbeans.apache.org/

OpenCV. (2021). Open Source Computer Vision. Obtenido de https://docs.opencv.org/3.4/da/d22/tutorial_py_canny.html

Pérez, D. (2019). Generación de números aleatorios. En D. Pérez Palau, Modelado y simulación numérica (págs. 150-190). Rioja: UNIR.

Rosas, M. (2006). Método para la reducción de ruido en imágenes utilizando la Transformada Wavelet Compleja con un algoritmo de Umbral Óptimo. Puebla, México: Bibliotecas UDLAP. Obtenido de http://catarina.udlap.mx/u_dl_a/tales/documentos/meie/rosas_o_mc/

Saini, J. K., & Harsh, K. V. (2013). A hybrid approach for image security by combining encryption and steganography. En IEEE (Ed.), 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013) (págs. 607-611). Shimla, India: IEEE. doi:10.1109/ICIIP.2013.6707665

Scooter Software. (2022). Obtenido de https://www.scootersoftware.com/

Software Guiffy. (2022). Obtenido de https://www.guiffy.com/Image-Diff-Tool.html

Suárez, P., & Villavicencio, M. (2017). Canny Edge Detection in Cross-Spectral Fused Images. Enfoque UTE, 8, 16-30. doi:10.29019/enfoqueute.v8n1.127

TIOBE. (2022). Obtenido de https://www.tiobe.com/tiobe-index/