Trabajo Práctico de Implementación: Esteganografía

Informe del proyecto 1C 2024

Fecha de entrega: 25/06/2024

Alumnos:
Camila Sierra Pérez 60242
Germán Ariel Martinez 58574
Santiago Preusche 59233
Magdalena Flores Levalle 60077

Docentes:
Pablo Eduardo Abad
Ana Maria Arias Roig
Rodrigo Ramele

Grupo 5

Índice

1. Introducción

El objetivo del Trabajo Práctico es lograr la implementación de un código que nos introduzca al campo de la esteganografía y sus aplicaciones, experimentando con métodos de ocultamiento de información en archivos de tipo ".bmp" y analizando las ventajas y desventajas de cada uno. A lo largo del siguiente informe se detalla la descripción del proyecto realizado junto con el análisis de la solución correspondiente al estegoanálisis de los archivos proporcionados por la cátedra.

2. Cuestiones a analizar

2.1. Aspectos relativos al documento/paper

Para la realización del proyecto, la cátedra nos proporcionó el paper científico "An improved LSB image steganography technique using bit-inverse in 24 bit colour image"de Majeed y Sulaiman, con el cuál pudimos entender el algoritmo de Least-Significant Bit.

2.1.1. Organización formal

La organización formal del documento es adecuada para introducir el tema. Comienza mencionando la técnica estándar de Least-Significant Bit y estableciendo cómo abordarla. Luego, presenta el esquema de las imagenes a color de 24 bits y explica su funcionamiento. Además, presenta el algoritmo de extracción y embedding de información oculta en imágenes, mostrando ejemplos del código y explicaciones paso-a-paso del método. Por último, muestra ejemplos de resultados obtenidos mediante estos algoritmos y conclusiones de dichos datos. En resumen, el paper se encuentra bien estructurado para la lectura de una persona que esta introduciéndose al mundo de la esteganografía.

2.1.2. Descripción del algoritmo

El paper describe el algortimo de manera eficaz. Utiliza ejemplos con pseudocódigo que hacen que al lector le resulte más fácil comprender el funcionamiento del mismo. La descripción es detallada y estructura bien el contenido en sus secciones para hacer llevadera la explicación para el lector, contando ejemplos y resultados obtenidos, y tiene referencias para aportar contexto y respaldar los argumentos del mismo. Podría agregarse la explicación de otros algoritmos de LSB, como los que fueron implementados en el proyecto.

2.1.3. Notación utilizada

La notación utilizada, en su mayoría, es clara y correcta. Sin embargo, puede resultar un poco confusa la forma en la que está hecho el entintado de algunas líneas de código, lo que hace que resulte tediosa la lectura por momentos, ya que debe leerse nuevamente para entender la intencionalidad de los escritores. Una forma fácil de solucionar esto sería que el texto no se encuentre dividido en dos columnas, y así favorecer a la lectura.

2.2. Ventajas y desventajas de esteganografiar un archivo con cada algoritmo

Para realizar el análisis de ventajas y desventajas de cada tipo de esteganografiado, se utilizó la imagen "itba.png", encontrada en el archivo "lado.bmp"

Figura 1: itba.png

A continuación se presentan los resultados de esteganografiar este archivo un total de tres veces con un mismo método. Es importante mencionar que se utilizo como indice de comparacion el valor PSNR del archivo "lado.bmp" el cual representa el indice de cambio (ratio) entre la mejor resolucion posible de la imagen y el ruido que tiene la imagen en si que afecta la correcta visualizacion de la misma. Por ende, si un algoritmo genera mayor ruido este es peor en su tarea de ocultar la imagen en comparacion con el resto. De esta experiencia se observaron los siguientes valores:

- LSB1: obtuvo un PSNR de 54.9845dB.
- LSB4: obtuvo un PSNR de 38.7042dB.
- LSBI: obtuvo un PSNR de 55.4500dB.

A mayor PSNR se tiene una mejor calidad de la imagen, viendo estos resultados se observa que el algoritmo que genera la imagen con mayor calidad es el LSBI Improved. Razonando sobre estos datos podemos llegar a las siguientes conclusiones (ventajas y desventajas) de los algoritmos utilizados:

- LSB1: la implementación es más simple y genera una imágen de una calidad que logra competir bastante bien con el algoritmo LSBI. Sin embargo, se necesitan portadores de gran tamaño para ocultar el secreto.
- LSB4: en comparación con los otros algoritmos este no requiere de portadores tan grandes, esto se realiza a costo de generar una imagen con menor calidad.
- LSBI: mejor calidad de imagen. Se necesitan portadores de mayor tamaño.

En resumen:

Algoritmo	**Resultados (PSNR)**	**Ventajas**	**Desventajas**
LSB1	54.98	Implementacion simple y calidad de imagen alta	Se necesitan portadores de gran tamaño
LSB4	38.7	No requiere portadores tan grandes como los otros metodos	La calidad de la imagen es menor
LSBI	55.45	Mayor calidad de imagen	Se necesitan portadores de gran tamaño

2.3. Descubrimiento en archivos esteganografiados

Para descubrir los mensajes ocultos, primero se debió implementar un algoritmo de desesteganografiado para poder extraer la información oculta en los diferentes archivos. Como pide el enunciado, se implementaron los algoritmos para LSB1, LSB4 y LSBI. Además, se implementó un algoritmo para desencriptar a partir de una clave, un algoritmo de encriptación y su modo.

A continuación se describe cómo fueron los pasos para descubrir todo lo que estaba oculto en cada archivo. Se contaba con 4 archivos *madrid.bmp*, *montevideo.bmp*, *medianoche1.bmp* y *titanic1.bmp*.

- En el archivo *montevideo.bmp*, si a este se lo leía como texto, podía descubrirse que el mensaje al final del archivo esconde la frase 'la password es camuflado'.
- En el caso de *medianoche1.bmp*, al extraerlo con el método LSBI obtuvimos un archivo de formato PDF que contiene la sigiuente frase: 'al .png cambiarle la extension por .zip y descomprimir'.
- Para el archivo de *titanic1.bmp*, al extraerlo utilizando LSB4, se obtuvo un .png de buscaminas (al cual llamaremos buscaminas.png).

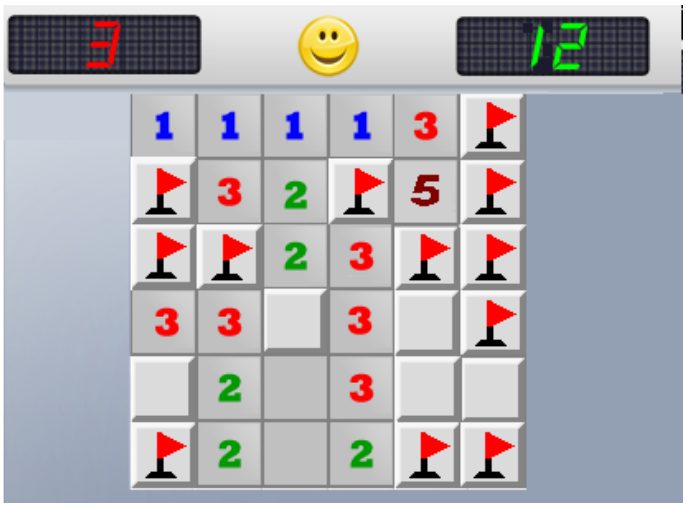

Figura 2: buscaminas.png

Siguiendo las instrucciones del pdf obtenido en el archivo *medianoche1.bmp*, hicimos unzip sobre *buscaminas.png* y así obtuvimos un archivo de texto *sol5.txt*. Este último contiene las instrucciones de cómo leer e interpretar los símbolos del buscaminas de forma tal que nos permite obtener el nombre y el modo del algoritmo de encripción con el que se encriptó un segmento de una película dentro de otro de los archivos proporcionado por la cátedra.

```
cada mina es un 1.
cada fila forma una letra.
Los ascii de las letras empiezan todos en 01.
Asi encontraras el algoritmo que tiene clave de 128 bits y el modo
La password esta en otro archivo
Con algoritmo, modo y password hay un .wmv encriptado y oculto.
```

Figura 3: sol5.txt

- Por último, teniendo en cuenta que el arhivo *madrid.bmp*, además de ser el restante para esteganografiar, tenía una tamaño mucho mayor al de los demás, supusimos que ahí estaba escondido el video. Al extraer con LSB1, aes128 cbc y haciendo uso de la password 'camuflado' obtenida gracias al archivo *montevideo.bmp*, pudimos obtener el .wmv que estaba oculto (sección 2.5).

2.4. Mensajes ocultos

Un mensaje oculto que se descubrió dentro de otro mensaje oculto, fue el caso del archivo *titanic1.bmp*. Como se mencionó en el punto anterior, este ocultaba una imagen de una partida del juego buscaminas. En una primera vista a la imagen, parecía no tener nada,

pero, al obtener las instrucciones de cómo interpretarlo del archivo *sol5.txt*, obtuvimos otro mensaje que estaba oculto en la imagen (un algoritmo y modo de encripción). Otro caso similar es lo que sucede con el video extraído del archivo *madrid.bmp* (sección 2.5).

2.5. Video oculto

Como se mencionó anteriormente, el video se encontraba escondido y encriptado en el archivo *madrid.bmp*. Desde un principio, las suposiciones de en dónde se encontraría este video, apuntaban a este archivo por ser notablemente más grande que el resto de los archivos a esteganografiar.

Figura 4: Video Oculto

En el video se muestra una tela que en un principio no parece ocultar nada. Pero al observarla más de cerca, se puede notar que en la forma en que pasan los hilos se representa un lenguaje (que puede ser representado por 1's y 0's). En este caso, al ver muy de cerca cada hilo, los protagonistas descubren que se oculta un nombre y un objetivo.

2.6. Otro método de esteganografiado

En el archivo montevideo.bmp, al mirar el archivo como texto (o realizar el comando strings sobre el mismo) vemos que al final del archivo se encuentra una frase que nos dice cual es la contraseña. Este metodo consiste en poner los bytes del texto al final del archivo. No altera la imagen pero aumenta el tamaño tanto como sea el tamaño del texto a ocultar.

2.7. Propuesta del documento de Majeed y Sulaiman

Para este informe se nos pidió trabajar con los algoritmos de estenografiado LSB1 y LSB4 junto con una variante conocida como LSBI o LSB-Improved. A la hora de trabajar con estos algoritmos, con cualquiera que seleccionemos, vamos a lograr nuestro objetivo de ocultar un archivo dentro de otro para enviar información oculta en forma de payload. Ahora, una característica que puede interesar es observar cual de estos es más imperceptible para las personas que simplemente estén observando el archivo de portada (el que se muestra de forma directa) y sobre este terreno es sobre el cual el algoritmo LSBI toma ventaja respecto a los otros dos algoritmos de esteganografía.

Esto sucede dado a que este algoritmo se potencia en dos ejes, el funcional y el natural. Sobre el eje funcional este algoritmo encuentra una manera distinta de ocultar información, utilizando inversión de bits para los bytes modificados que tengan mas impacto en la imagen. Esto, deja a la imagen modificada de la forma más similar a la original posible. Luego, sobre el eje natural, este aprovecha las desventajas inertes que posee el ser humano en la forma en la que este detecta y percibe colores.

Expandiendo sobre lo mencionado en el último párrafo, este algoritmo opta por no alterar la imagen en los bytes que representan el color rojo de los píxeles de la imagen de portada, esto es, dado a que las personas solo son capaces de detectar con un 65 % de eficacia variaciones en los colores rojos. Esto, en comparación con los colores azul (2 %) y verde (33 %) hace que el rojo sea el color sobre el cual más fácil es de detectar un cambio realizado por el algoritmo. Es por esto que el algoritmo de LSBI opta por saltearse los bytes rojos dejandolos como ruido para cuando un usuario no autorizado intente encontrar el secreto oculto en el archivo principal. Agregando así una barrera de seguridad mucho más sólida que la de los otros algoritmos.

Por lo mencionado hasta ahora en este apartado se concluye que el algoritmo LSBI es una clara mejora sobre los algoritmos LSB1 y LSB4 con los que se trabaja en este proyecto.

2.8. Otra forma de guardar el registro de los patrones invertidos

En la implementación del proyecto entregado se realiza una lectura de los patrones para guardarlos primero, en un array de booleanos para conocer qué patrones de bits fueron modificados y cuales no; y, en un segundo array, la cantidad de matcheos que tiene ese patrón. Otra manera de realizar el guardado del registro de los patrones invertidos podría ser duplicando el patrón al final de la imagen. Asimismo, podría guardarse en otra posición prefijada, siempre asegurando que la misma no se encuentre ocupada por el esteganografiado del payload con su metadata.

2.9. Dificultades encontradas

Las dificultades con las que nos encontramos a la forma de implementar el algoritmo del paper en nuestro proyecto fue el manejo de los bytes rojos en el método de esteganografiado LSBI (least significant bit improved). El mismo no estaba funcionando correctamente debido a que estos debían ser ignorados por el algoritmo. El paper habla de como el ojo humano pone más énfasis en el color rojo dentro de la tabla RGB, por lo que un mensaje encriptado en estos bytes sería más fácilmente identificable. Además, menciona que la ventaja que se obtiene de tratar a estos bytes con esta técnica, permite que actúen como ruido para dificultar la extracción del mensaje oculto. Una vez que realizamos esta implementación, el código funcionó correctamente.

2.10. Posibles mejoras

Luego de realizar este trabajo, se nos ocurrieron tres posibles mejoras que podrían hacerse. En primer lugar, podría incluirse soporte para otro tipo de archivos además de ".bmp". También se podría agregar un método que detecte el modo de esteganografiado usado para no tener que correr cada archivo con los distintos tipos de LSB. Por último, podrían realizarse mejoras en la eficiencia del programa, como por ejemplo, que el método de LSBI utilice un index para que la cantidad de iteraciones sea menor.