INFORME DE ANALISIS DE LECTURA

KEWI YILIAN MOYA CATAÑO 1033487777

MARDEY SOLANYI ARIAS GAVIRIA 1039689519

La empresa informá2 nos pide la creación de un algoritmo que revierta una imagen distorsionada a la original. Nos proporcionan que la imagen se les pudo hacer echo unas transformaciones que son a nivel de bits y una máscara que se le sumará a un valor de la imagen. Primero nos centraremos en hacer pruebas con transformaciones y la máscara para entender cómo funcionan y afecta los datos.

Análisis de las transformaciones

Tenemos que las trasformaciones son:

* Un xOr con una imagen de ruido
* Desplazamientos de bits a la derecha o izquierda
* Rotaciones a la derecha o la izquierda

Análisis de los desplazamientos

En el caso del desplazamiento de bits a la derecha o a la izquierda sabemos que el máximo rango de movimiento puede ser de 8 bits. Sabemos que esta transformación se les hace a todos los bytes RGB de la imagen. Realizando algunos experimentos descubrimos que los desplazamientos tienen perdida de la información.

Les mostrare un ejemplo con dos desplazamientos una de 8 bits a la derecha y otro a la de 4 bits a la izquierda:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |

Bit original:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Primer desplazamiento:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Segundo desplazamiento:

Como pueden ver en este ejemplo los datos que se desplazan hacia un lado se pierden, y en caso de que se vuelva a desplazar, pero en su contraria, seguirá existiendo pérdida importante de información. En resumen, este operador daña la integridad de los datos y no sería útil para codificar ya que son operaciones irreversibles

Por otro lado, si esta información se pudiera almacenar en algún lugar, deberían ser específicos en un archivo aparte para posteriormente incorporarlos a la imagen original, sin embargo, este no es el caso.

Análisis de las rotaciones

En el caso de las rotaciones de bits a la derecha o a la izquierda sabemos que la máxima cantidad de movimientos es de 8 bits. Esta transformación se les aplica a todos los bytes RGB de la imagen.

Realizando algunos experimentos descubrimos que las rotaciones no tienen pérdida de la información.

Como la rotación de bit hace que el bit de un extremo pase a un extremo a otro no hay modificación de los datos ya que se hacen solo con la misma información, haciendo posible que vuelvan a su estado original.

Un ejemplo muy específico en este caso es que roten 4 bits a la izquierda y luego que se roten 4 bits a la derecha:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

Bit original:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |

Primera rotación:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |

Segunda rotación:

Cómo podemos visualizar en este caso muy específico, observamos que con las rotaciones que fueron la misma cantidad de bit, pero a direcciones contrarias, vemos que volvieron a los datos originales esto significa que las rotaciones pueden volver a su estado original y que no habrá pérdida de información. Como conclusión las rotaciones mantienen la integridad de los datos.

Análisis de xOr

En ese caso es un poco más complejo ya que la forma de hacer esa transformación tenemos una imagen que se va a utilizar para comparar cada a byte con la operación xOr de la imagen original. Con esto nos conlleva que toda la imagen va a ser transformada a partir de una imagen aleatoria del mismo tamaño.

Forma

El contenido generado por IA puede ser incorrecto.Primero vamos a ver cómo funciona xOr.

Teniendo en cuenta el caso en el que tomemos los primeros pixeles de cada imagen la originaria y las distorsionada. Y de eso tomamos el primer byte de cada uno y realizamos la operación y con el resultado realizamos la misma operación, pero se hace contra en byte de la imagen distorsionada.

M/O: byte de la imagen original

M/D: byte de la imagen distorsionada

C/R: resultado de xOr de M/O y M/D

C/Q: resultado de xOr de C/R y M/D

v C/Q

Tabla #2

Tabla #1

|  |  |  |
| --- | --- | --- |
| **M/O** | **M/D** | **C/R** |
| 0 | 1 | 1 |
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |

|  |  |  |
| --- | --- | --- |
| **C/R** | **M/D** | **C/Q** |
| 1 | 1 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| 1 | 0 | 1 |

Como podemos ver comparando los resultados de la operación xOr hecho en las tablas nos damos cuenta a qué podemos volver a la información original,

Y esto se comprueba ya que M/O y C/Q, Sus valores de byte son los mismo, llegando a la conclusión que si guardad la integridad de los datos.

Conclusión del análisis de las trasformaciones.

Después de analizar cada tipo de transformación nos percatamos que verdaderamente las únicas 2 transformaciones que se pueden aplicar a la imagen sin perder información serian las rotaciones y las operaciones xOr. Ya que según nuestros análisis y conclusiones de los diferentes tipos de transformaciones, consideramos que los desplazamientos pueden ser una trampa que nos aleja del camino correcto para hallar la imagen original.

Por tanto:

* Transformaciones que si se utilizan: xOr y rotaciones.
* Transformaciones que no se utilizan: desplazamientos.

Análisis de la mascara

la información que nos proporcionan de la máscara es que se le suma como valor entero a entero, y es más pequeña que la imagen original. La máscara utiliza un valor S aleatorio, que desplazará S pixeles desde el píxel cero de la imagen.

La máscara se añade a la imagen en cada paso de la transformación, esto nos lleva a pensar que la imagen tiene pocos pasos y vemos que la máscara solo distorsiona una pequeña parte de la imagen original. Pero si son muchos pasos la máscara se distorsionará mucho cuando intente volver a la original.

Un ejemplo grafico:

Imagen original:

Mascara:

Imagen A Imagen B

Como vemos en la imagen A se hicieron pocos pasos teniendo pocas máscaras eso nos da como resultado que si queremos pudiéramos buscar la forma de encontrar la imagen original a prueba de error solamente a transformación. pero siendo muy ineficientes por qué tendríamos que hacer una imagen por cada caso posible de transformación hasta encontrar una imagen muy parecida a la original, pero con fallas por las máscaras.

Pero en la imagen número B hay una cantidad de pasos mayor haciendo que la integridad de los datos de la imagen se vea afectada.

Conclusión de la mascara

Los valores de la máscara que están en la imagen se deben eliminar (desenmascarar) haciendo una resta que es el proceso inverso a la suma; de esta manera ya casi aseguramos la integridad de los datos y tenemos la capacidad de volver a la imagen original sin corrupción.

Creación de la estructura del algoritmo

Para empezar, veamos cómo se hace el proceso de encriptación:

**TA:** trasformaciones

**M:** mascara

**TX:** archivo dejado en el proceso (pista)

**M\_O:** imagen original

**M\_D:** imagen distorsionada

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Paso 1*** | | | | | ***Paso 2*** | | | | | ***Paso 3*** | | | |
| **M\_O** | TA | M | TX.1 | M\_D1 | **M\_D1** | TA | M | TX.2 | M\_D2 | **M\_D2** | TA | M | M\_D3 |

Como vemos en el proceso de encriptación se divide en pasos y los pasos se dividen

En entrada de la imagen, transformaciones, mascara, archivo de texto, imagen encriptada. Este proceso se puede repetir varias veces. Pero nosotros no tendremos esas imágenes internadas ya que nuestras entradas serán:

* Imagen distorsionada final. (en el caso de arriba seria M\_D3)
* Archivos de texto. (en el caso de arriba serian TX.1, TX.2)
* Máscaras
* Y una imagen de ruido gaussiano que se puede utilizar en una transformación.

Si quieres regresar a la imagen original tendríamos que hacer el proceso inverso. Y este proceso tenemos que encontrar las trasformaciones que se le hacen a la imagen original y de esta manera aplicarlo hasta llegar a la imagen original.

Funciones necesarias.

Los profesores nos proporcionaron algunas funciones que nos ayudan en el proceso de codificación.

Loadpixeles: carga una imagen y la convierte en formato de 24 bits, extrae los datos de esta en un puntero que contiene un arreglo lineal y retorna este puntero. (en resumen, esta función podrá ser utilizada para sacar la información de la imagen distorsionada)

ExportImage: crea imagen de QT vacía, y copia los valores de RGB del puntero anterior (se pasa como argumento de la función) y guarda esta imagen.

LoadSeedMasking: Lee los archivos de texto dados para la práctica, toma la primera línea como semilla y devuelve un arreglo con datos RGB y cantidad de pixeles.

Teniendo en cuenta esta gran ayuda, el proceso de codificación se llevaría a cabo de la siguiente manera:

1. Utilizamos la función de loadpixeles para que nos brinde la información de la imagen distorsionada, a continuación, con esa información, debemos crear una función que se encargue de realizar la transformación que queremos aplicar y que retorne la información de la transformación en un arreglo.
2. Según nuestro análisis anterior, queremos desenmascarar, por tanto, debemos utilizar la función loadpixeles nuevamente, pero esta vez con la imagen de la máscara para extraer la información a un arreglo y junto con la función loadseedmasking para obtener la información de los archivos de texto.

Después de tener la imagen distorsionada y la imagen de la máscara en arreglos, hay que crear una función que se encargue de realizar la resta de la máscara en la porción de la imagen que queremos encontrar. El valor de retorno debe ser un arreglo que contenga toda la información de la imagen ya procesada.

1. Después debemos utilizar el valor retornado del paso 3 y pasarlo como argumento a la función importImage.
2. Por último, hacer la comparación y analizar si nuestro procedimiento fue correcto o si debemos aplicar otra transformación.