Códigos y Criptografía - Curso 2018-2019

Práctica 8: Cifrando una imagen: Arnold.

Usaremos la función de prácticas anteriores *inv_modulo(A,n).*

Todas las funciones que se realicen a lo largo de la práctica deben ser válidas tanto para fotografías en escala de grises como en RGB.

1.- Función desorden_pixel (foto, A)

Función que desordena los píxeles de las matrices asociadas a una imagen, *foto*, de acuerdo a la transformación asociada a la matriz *A*. No queremos que nos muestre nada, ni imágenes ni matrices, sólo que guarde las matrices obtenidas para usarlas en otras funciones.

Para ello se puede usar *setappdata (gcf, 'matriz', matriz)*, y cuando más adelante necesitemos usar esa matriz, *matriz = getappdata (gcf, 'matriz')*.

Entradas:

foto: la foto de la que queremos desordenar sus píxeles. Debe ser cuadrada.

A: matriz que nos determina la transformación. Debe ser 2x2 y tener inversa módulo el número de filas de *foto*.

Salidas: Ninguna. Debe guardar las nuevas matrices obtenidas para un posible uso posterior.

2.- Función arnold (foto, A)

Función que va a ordenar o a desordenar una foto. Para elegir una de estas dos opciones se debe usar un switch con dos casos: **caso 1** para desordenar y **caso 2** para ordenar.

Entradas:

foto: fotografía (que debe ser cuadrada), a la que queremos aplicarle una transformación de Arnold.

Debe ser la fotografía original en el *caso 1*, y la fotografía desordenada según la matriz *A* en el *caso 2*.

A: matriz que se va a usar para desordenar en el caso 1, o que ya se ha usado para desordenar en el caso 2. Debe ser una matriz 2x2 con inversa módulo el número de filas de *foto*.

Salida:

En el caso 1 la imagen desordenada según la transformación indicada por la matriz *A.*

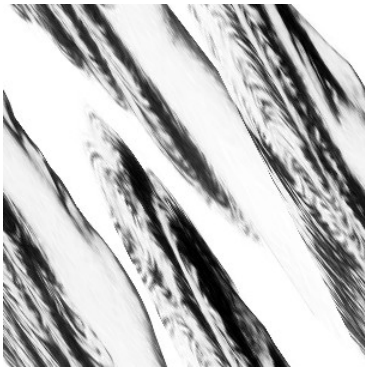
En el caso 2 la matriz ordenada teniendo en cuenta que se desordenó usando la matriz *A*.

Ejemplo

>> arnold('hypatia.bmp',[1 2;1 1])

Introduce un 1 si quieres desordenar, o un 2 si quieres ordenar: 1

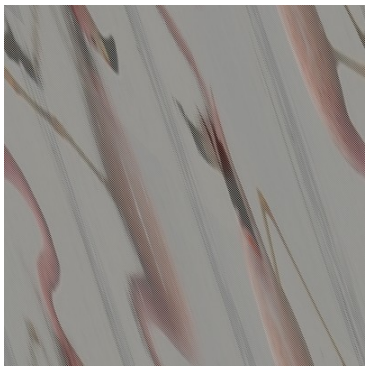
hypatia.bmp



hypatiadesordenada.bmp

>> arnold('desorden.bmp',[1 3;1 1])

Introduce un 1 si quieres desordenar, o un 2 si quieres ordenar: 2



desorden.bmp

orden.bmp

3.- Función n = pote (A, m)

Función que debe calcular el mínimo valor del exponente de la potencia de *A* que módulo *m* es igual a la matriz identidad.

Entradas:

A: matriz (cuadrada de orden 2 y con inversa módulo *m*).

m: módulo de trabajo.

Salida: El valor del exponente que hace que la correspondiente potencia de *A* sea la identidad.

Ejemplo

```
>> n=pote([54 118;260 14],193)

n =    96
```

4.- Función potencia = arnold_02 (foto, A)

Función que debe desordenar los píxeles de la imagen *foto* según la matriz *A*, de manera sucesiva. Vamos a usar un switch con dos casos: **caso 1** para desordenar hasta recuperar la imagen original y **caso 2** para desordenar el número de veces que se indique.

Entradas:

foto: foto que queremos desordenar.

A: matriz que determina la transformación.

Salida: *potencia*: el número de veces que hemos realizado la transformación de forma sucesiva. Además, en ambos casos debe mostrar la imagen original y todas las imagenes transformadas que se hayan ido realizando.

Ejemplo

```
>> potencia=arnold_02('hypatia_peq.bmp',[2 1;1 0])

introduce 1 si quieres desordenar la imagen hasta volver a la original, o 2 si quieres
desordenarla hasta una determinada potencia: 1

hay que hacer 48 transformaciones

potencia =    48


>> potencia=arnold_02('hypatia_peq.bmp',[2 1;1 0])

introduce 1 si quieres desordenar la imagen hasta volver a la original, o 2 si quieres
desordenarla hasta una determinada potencia: 2
```

¿cuántas transformaciones quieres hacer? 10

potencia = 10

hypatia_peq.bmp

hypatia_peq_desorden.bmp

(10 potencias)