Visión Artificial

Título: Practica 3

Nombre: Mario Gerardo Casas Miramontes

Registro: 22310165

Objetivo

Objetivo: Utilizar las funciones de umbrales para la recuperación de información.
Threshold1 binary, b_inv, Trunc, To Zero, Tz_inv, Mean, Gaus, Otsu.

Funcionamiento

El programa que se utilizo en base a la practica numero 2 en donde se realizaron las operaciones aritméticas a unas imágenes, sin embargo, para esta práctica es necesario que le realicemos un histograma y una ecualización tanto al histograma como a la imagen, a su vez que tienen que estar todas en una misma pestaña.

Para ello fue necesario que creáramos un objeto que tuviera 4 posiciones, ósea, dos filas y dos columnas y se realizo gracias a plt.figure la cual nos ayudo a realizar esta acción, a su vez que pasamos la imagen a RGB para que pueda ser interpretada por mathplot.

En la primera posición directamente le asignamos un espacio en la figura y no modificaos nada.

Para la segunda posición fue necesario identificar si la imagen tiene tres canales, ósea, colores y en caso de tener se realizaba un movimiento por la imagen para la detección de los pixeles y generamos un histograma gracias a la función cv2.calcHist. También fue necesario realizar un movimiento de las formas en las que se presenta la imagen para poderla representar correctamente dentro de la figura.

Para la tercera figura únicamente realizamos la impresión de la imagen ecualizada.

Finalmente, para la cuarta posición fue necesario realizar el histograma en base a la ecualización realizada, repitiendo así el proceso realizado en la generación del primer histograma, contando también con tener una parte por si la imagen es a color o escala de grises.

Código Fuente

```
#Mario Gerardo Casas Miramontes   6°G    22310165

#Practica 2


import cv2

import numpy as np

import matplotlib.pyplot as plt

# Cargar las imágenes

img1 = cv2.imread('3D-Matplotlib.png')  # Imagen de fondo

img2 = cv2.imread('mainlogo.png')  # Logotipo a superponer

img3 = cv2.imread('mainsvmimage.png')

(h, w) = img1.shape[:2] #Sacar valores del tamaño de la imagen
```

```
#Operaciones basicas
suma = img1+img3
resta = img1-img3
Mul = cv2.multiply(img1, img3)
Division = cv2.divide(img1, img3)

#Centro de la imagen 1
centro = (w // 2, h // 2)

# Crear la matriz de rotación
angulo = 90  # Cambia el ángulo según necesites
escala = 1.0  # Factor de escala (1.0 = mismo tamaño)
MatrizR = cv2.getRotationMatrix2D(centro, angulo, escala) #Matriz para la rotacion
Rotacion = cv2.warpAffine(img1, MatrizR, (w, h))

#Operacion: Traslacion
MovX = 100 #Desplazamiento X
MovY = 100 #Despalzamiento Y
MatrizT = np.float32([[1, 0, MovX], [0, 1, MovY]]) #matriz de traslacion
Traslacion = cv2.warpAffine(img1, MatrizT, (w, h))

#Operacion transpuesta
Transpuesta = cv2.transpose(img1)


rows, cols, channels = img2.shape
roi = img1[0:rows, 0:cols]
img2gray = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)
```

```
# Aplicar umbral para obtener la máscara del logo
ret, mask = cv2.threshold(img2gray, 220, 255, cv2.THRESH_BINARY_INV)

# Invertir la máscara
mask_inv = cv2.bitwise_not(mask)

img1_bg = cv2.bitwise_and(roi, roi, mask=mask_inv)

# Aplicar la máscara al logo para extraer solo la parte del logo sin fondo
img2_fg = cv2.bitwise_and(img2, img2, mask=mask)

# Combinar ambas imágenes (fondo con "hueco" + logo sin fondo)
dst = cv2.add(img1_bg, img2_fg)

# Colocar el resultado en la imagen principal
img1[0:rows, 0:cols] = dst

#La creacion de las imagenes y operaciones se mantienen igual en este momento realizamos los 4 apartados

def mostrar_imagen_y_histograma(nombre_ventana, imagen, figura_num=None):
    plt.figure(figura_num, figsize=(15, 10)) #Nos dedicamos a crear una figura

    # Convertir BGR a RGB
    if len(imagen.shape) == 3:
        imagen_mostrar = cv2.cvtColor(imagen, cv2.COLOR_BGR2RGB)
        es_color = True
    else:
        imagen_mostrar = imagen
        es_color = False
```

```
#Posicion 1: Imagen


    plt.subplot(2, 2, 1)    #Creamos una subfigura en la figura e insertamos en esa posicion
    if es_color:
        plt.imshow(imagen_mostrar)#Mostrar la imagen si es de color
    else:
        plt.imshow(imagen_mostrar, cmap='gray')#En caso de que la imagen sea gris se pone directo
    plt.title(f'{nombre_ventana} - Original')
    plt.axis('off')



#Posicion 2: Histograma


    plt.subplot(2, 2, 2) #Asignamos la posicion donde insertaremos la imagen del histograma
    if es_color: #Si detecta que es de color  se realiza lo sig.
        colores = ('b', 'g', 'r') #Tiene tres canales
        for i, col in enumerate(colores):
            hist = cv2.calcHist([imagen], [i], None, [256], [0, 256])#Generamos el histograma en base a
Imagen y los parametros solicitados
            plt.plot(hist, color=col, label=col)
        plt.legend()
    else:
        hist = cv2.calcHist([imagen], [0], None, [256], [0, 256]) #En caso de que la imagen sea en
escala de grises
        plt.plot(hist, color='k')
    plt.title('Histograma Original')
    plt.xlim([0, 256])



    if es_color:
```

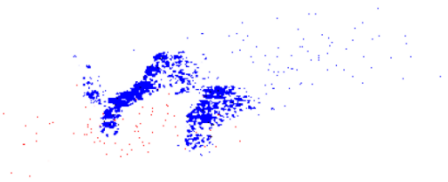
```
        # Para imágenes a color, ecualizamos en el espacio YUV
        img_yuv = cv2.cvtColor(imagen, cv2.COLOR_BGR2YUV)#Pasamos la imagen a YUV
        img_yuv[:,:,0] = cv2.equalizeHist(img_yuv[:,:,0])#Ecualizamos la imagen
        img_ecualizada = cv2.cvtColor(img_yuv, cv2.COLOR_YUV2BGR)#La volvemos a pasar al formato BGR
        img_ecualizada1= cv2.cvtColor(img_ecualizada, cv2.COLOR_BGR2RGB) #La volvemos a pasar a RGB
    else:
        # Para escala de grises, ecualización directa
        img_ecualizada = cv2.equalizeHist(imagen)
        img_ecualizada1 = img_ecualizada


#Posicion 3: Imagen ecualizada


    plt.subplot(2, 2, 3) #Establecemos la posicion de la imagen
    if es_color:
        plt.imshow(img_ecualizada1)#Mostramos la imagen si es de color
    else:
        plt.imshow(img_ecualizada1, cmap='gray')    #Mostramos si es negra
    plt.title(f'{nombre_ventana} - Ecualizada')
    plt.axis('off')


#Posicion 4: Histograma ecualizado


    plt.subplot(2, 2, 4)
    if es_color:
        colores = ('b', 'g', 'r')
        for i, col in enumerate(colores):
            hist_eq = cv2.calcHist([img_ecualizada], [i], None, [256], [0, 256])#Generamos el histograma cuzliazod
```

```
            plt.plot(hist_eq, color=col, label=col) #Se hace la grafica
        plt.legend()
    else:
        hist_eq = cv2.calcHist([img_ecualizada], [0], None, [256], [0, 256])#Si es de color blanco y negro se ecualiza directo
        plt.plot(hist_eq, color='k')
    plt.title('Histograma Ecualizado')
    plt.xlim([0, 256])


    plt.tight_layout()


# Mostrar todas las imágenes con sus histogramas
mostrar_imagen_y_histograma('Imagen Original', img1, 1)
mostrar_imagen_y_histograma('Logo en Escala de Grises', img2gray, 2)
mostrar_imagen_y_histograma('Máscara del Logo', mask, 3)
mostrar_imagen_y_histograma('Máscara Invertida', mask_inv, 4)
mostrar_imagen_y_histograma('Fondo sin Logo', img1_bg, 5)
mostrar_imagen_y_histograma('Solo el Logo', img2_fg, 6)
mostrar_imagen_y_histograma('Suma', suma, 7)
mostrar_imagen_y_histograma('Resta', resta, 8)
mostrar_imagen_y_histograma('Multiplicación', Mul, 9)
mostrar_imagen_y_histograma('División', Division, 10)
mostrar_imagen_y_histograma('Rotación', Rotacion, 11)
mostrar_imagen_y_histograma('Traslación', Traslacion, 12)
mostrar_imagen_y_histograma('Transpuesta', Transpuesta, 13)
mostrar_imagen_y_histograma('Resultado Final', img1, 14)


plt.show()
```

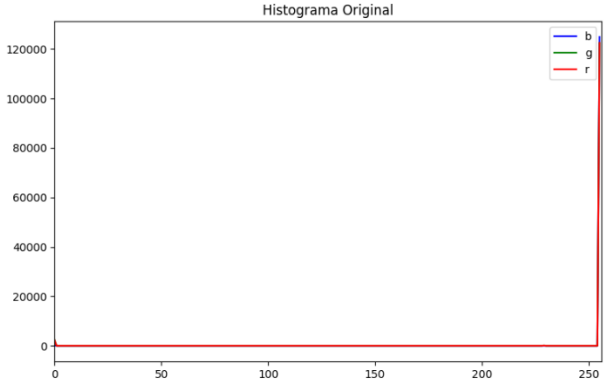
Resultado

Figure 9

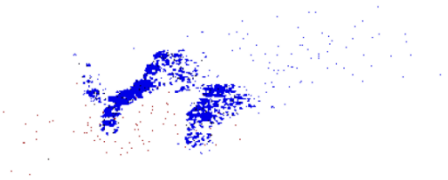
Multiplicación - Original



Histograma Original



Multiplicación - Ecualizada



Histograma Ecualizado

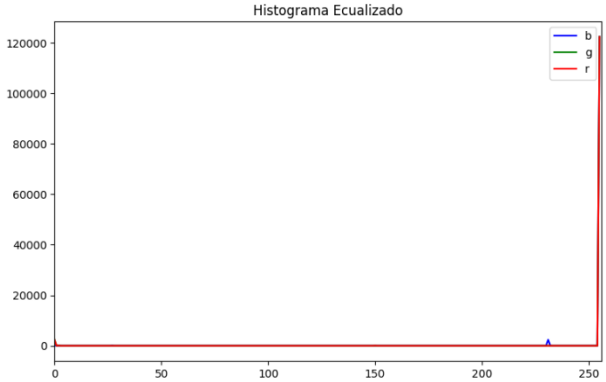
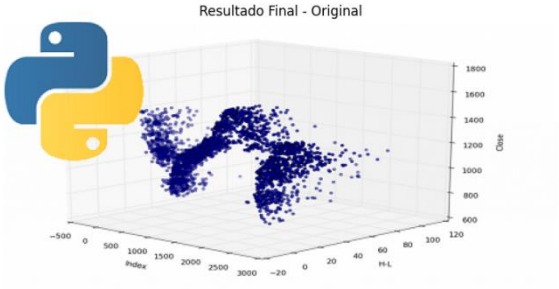
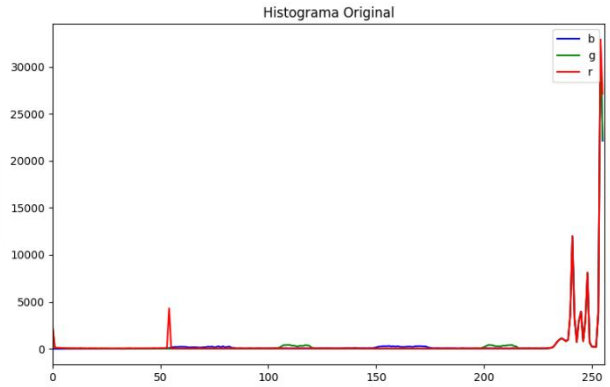


Figure 14

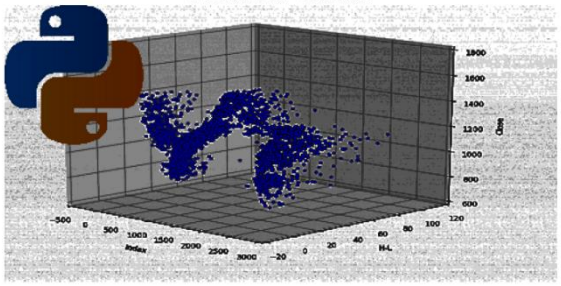
Resultado Final - Original



Histograma Original



Resultado Final - Ecualizada



Histograma Ecualizado

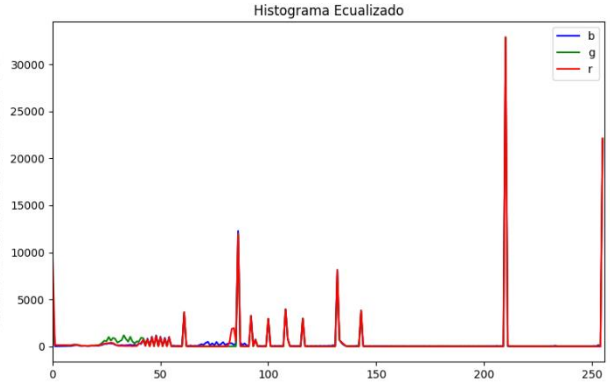
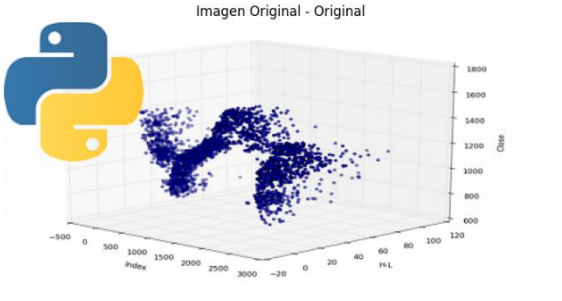


Figure 1

Imagen Original - Original



Histograma Original

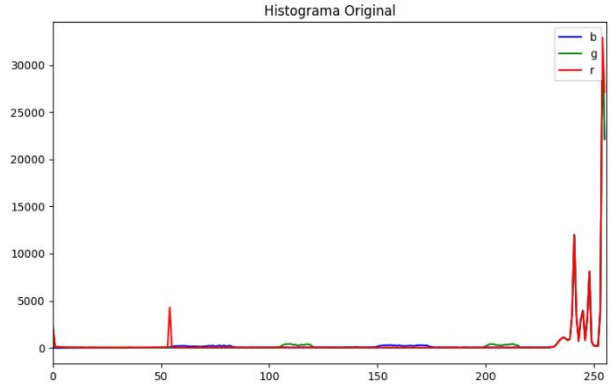
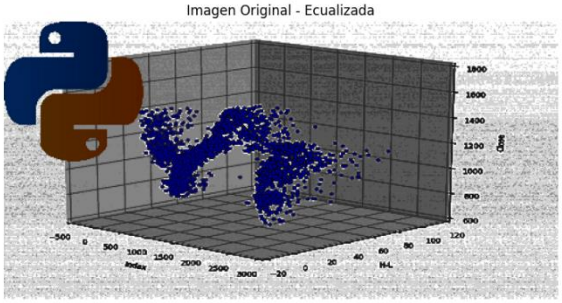


Imagen Original - Ecualizada



Histograma Ecualizado

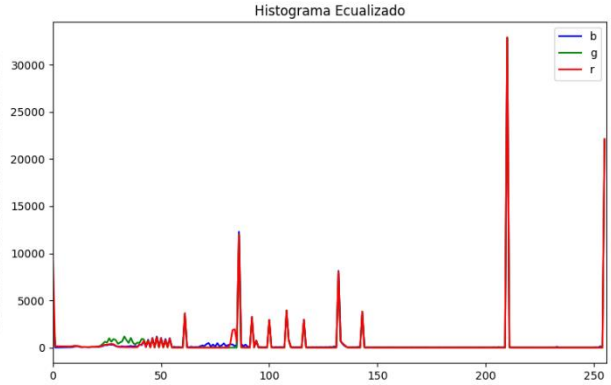
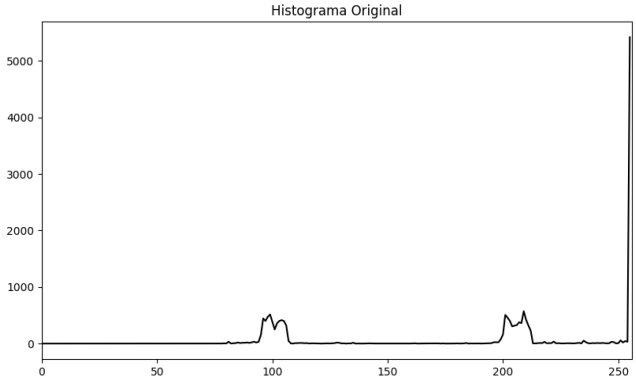


Figure 2

Logo en Escala de Grises - Original

Histograma Original



Logo en Escala de Grises - Ecualizada

Histograma Ecualizado

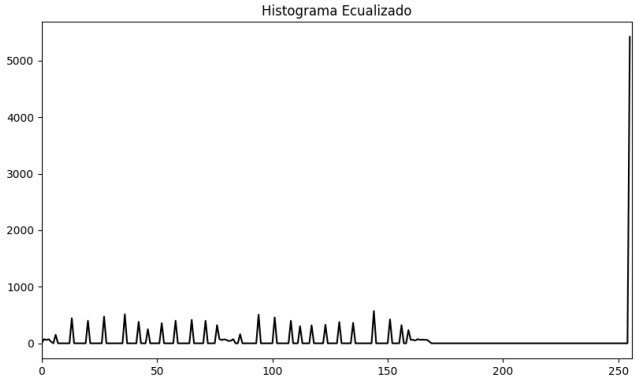
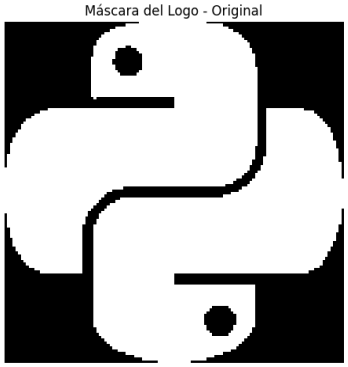
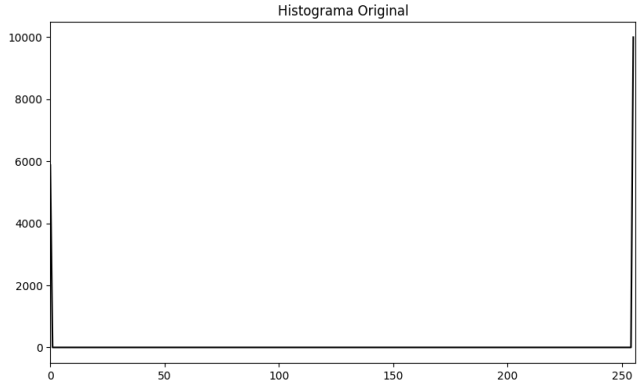


Figure 3

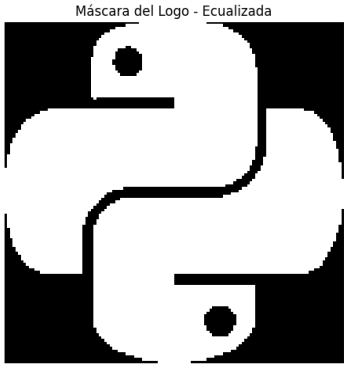
Máscara del Logo - Original



Histograma Original



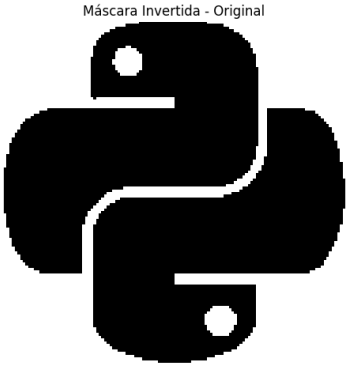
Máscara del Logo - Ecualizada



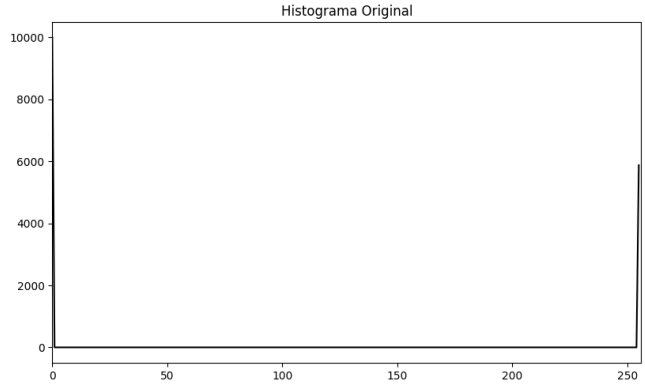
Histograma Ecualizado

Figure 4

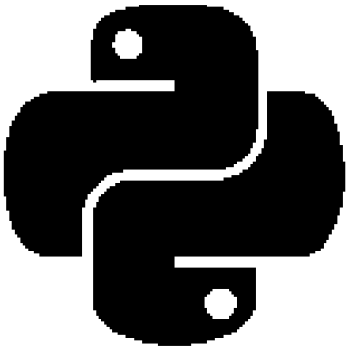
Máscara Invertida - Original



Histograma Original



Máscara Invertida - Ecualizada



Histograma Ecualizado

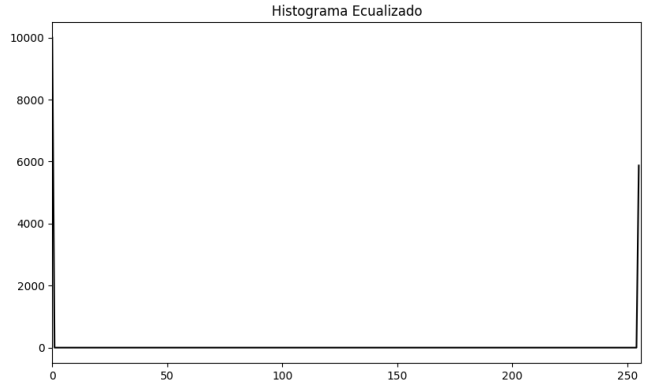
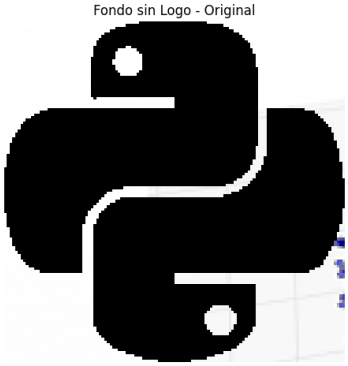
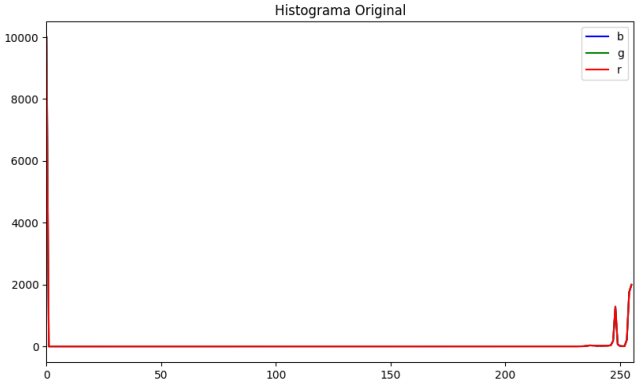


Figure 5

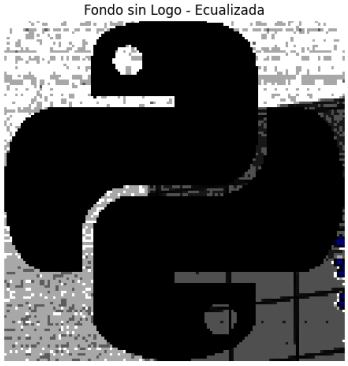
Fondo sin Logo - Original



Histograma Original



Fondo sin Logo - Ecualizada



Histograma Ecualizado

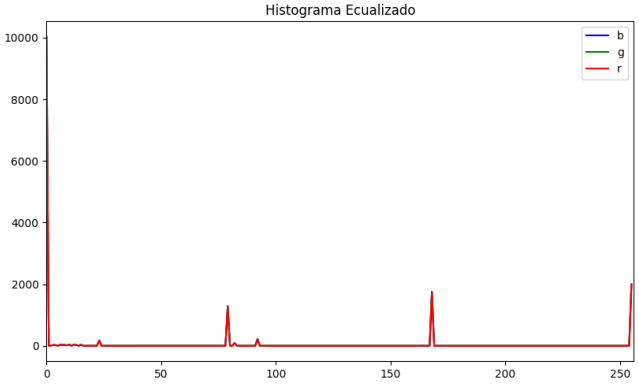
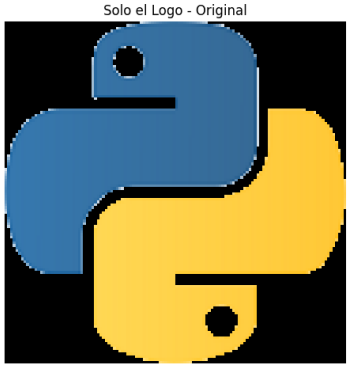
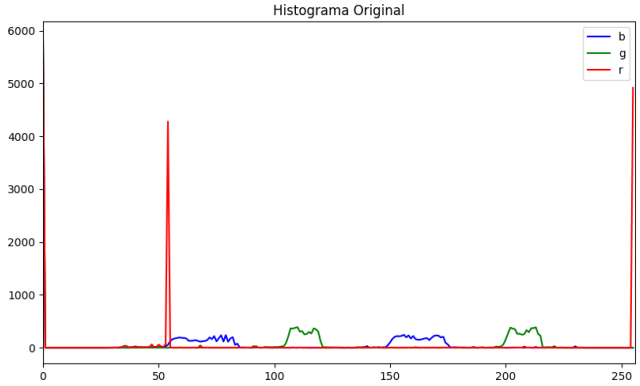


Figure 6

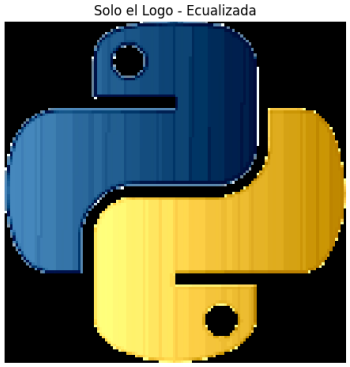
Solo el Logo - Original



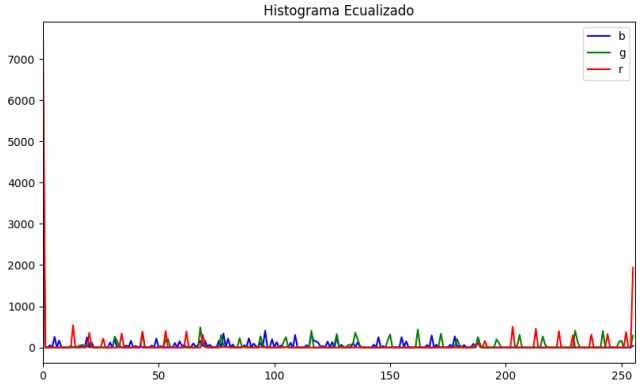
Histograma Original



Solo el Logo - Ecualizada



Histograma Ecualizado



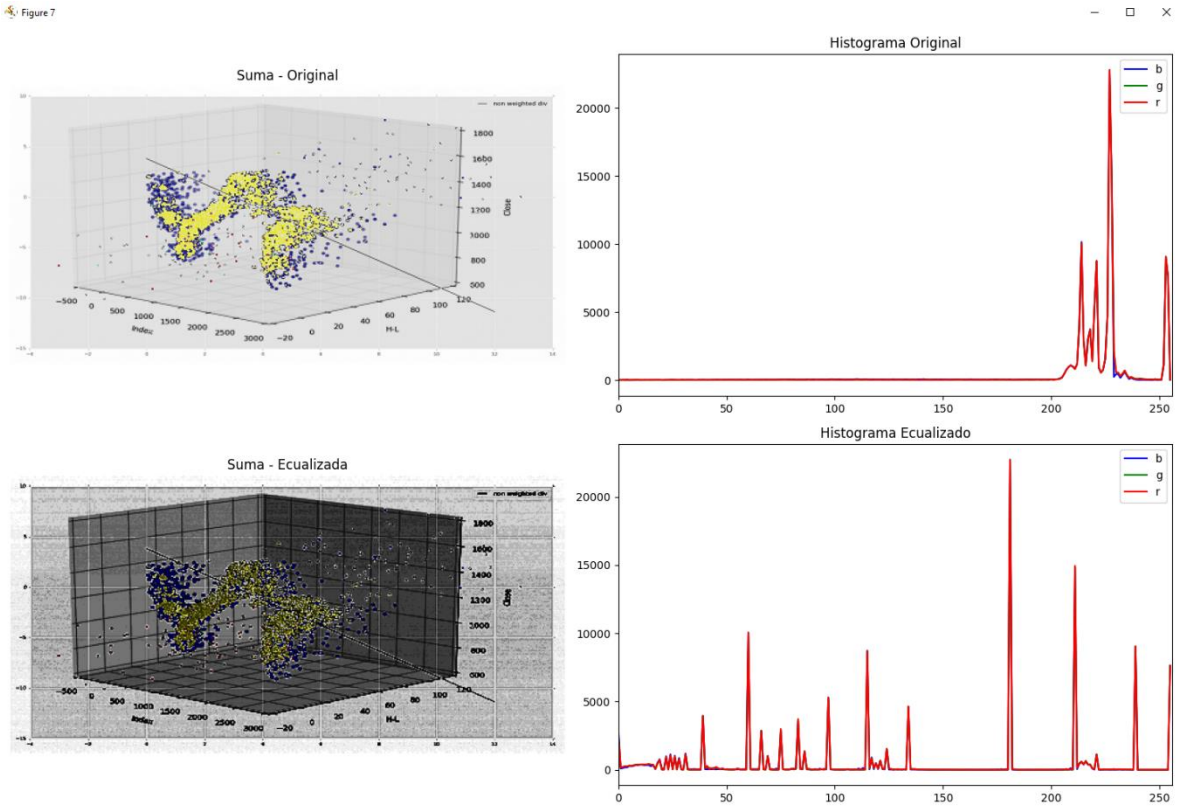
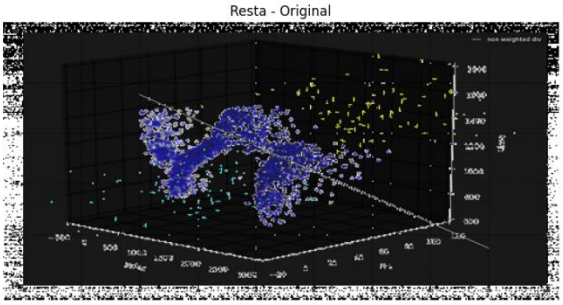


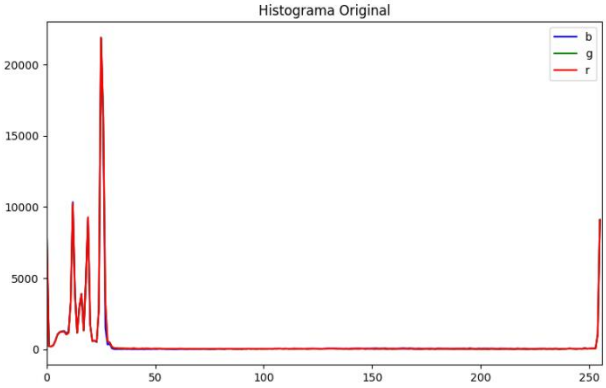
Figure 7
Suma - Original
Histograma Original
b
g
r
Suma - Ecualizada
Histograma Ecualizado
b
g
r

Figure 8

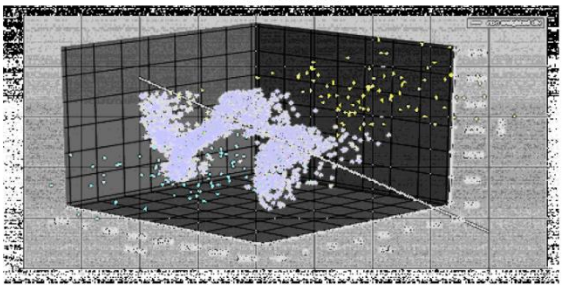
Resta - Original



Histograma Original



Resta - Ecualizada



Histograma Ecualizado

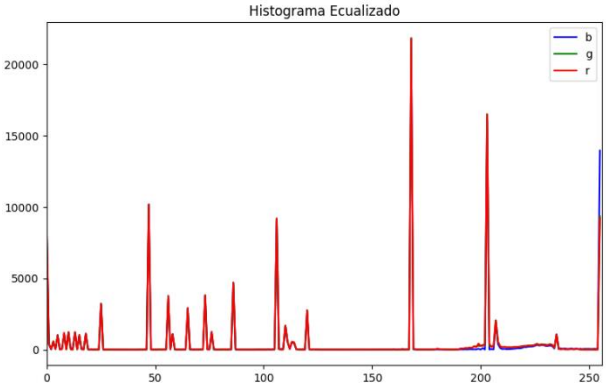
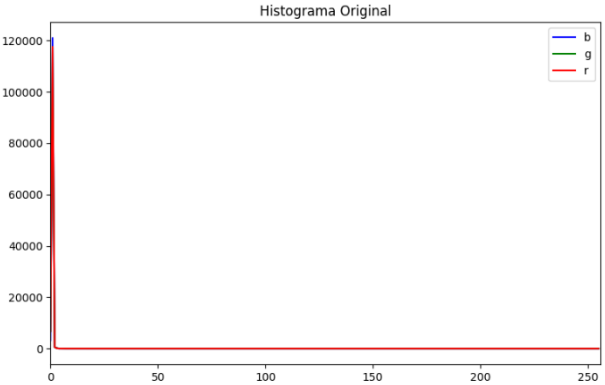


Figure 10

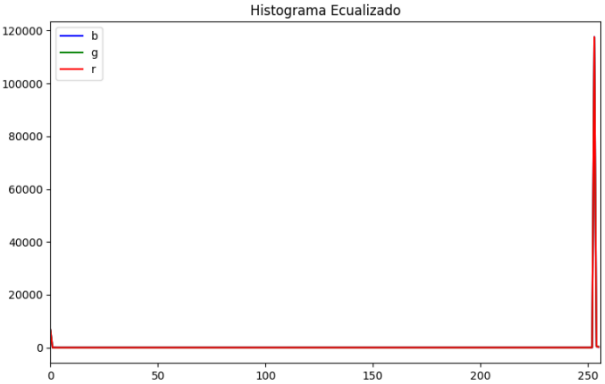
División - Original

Histograma Original



División - Ecualizada

Histograma Ecualizado



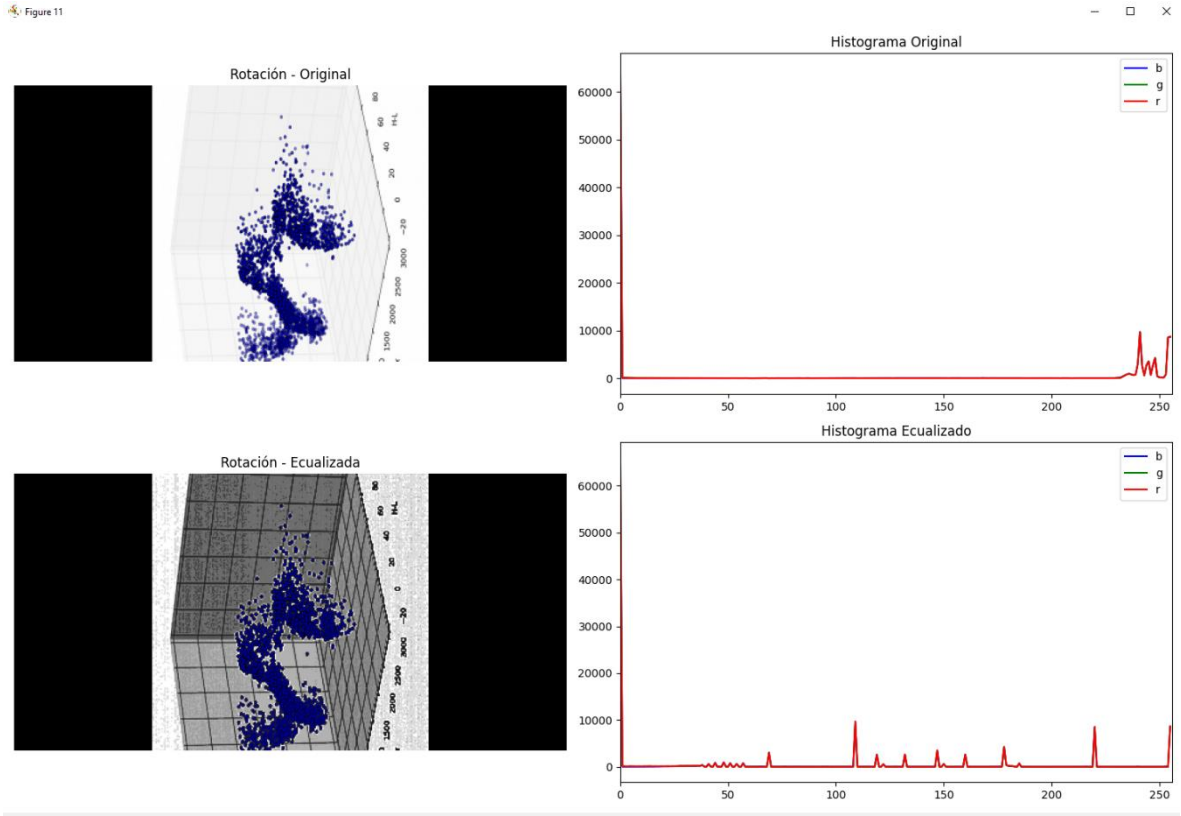


Figure 11
Rotación - Original
Histograma Original
Rotación - Ecualizada
Histograma Ecualizado
b
g
r

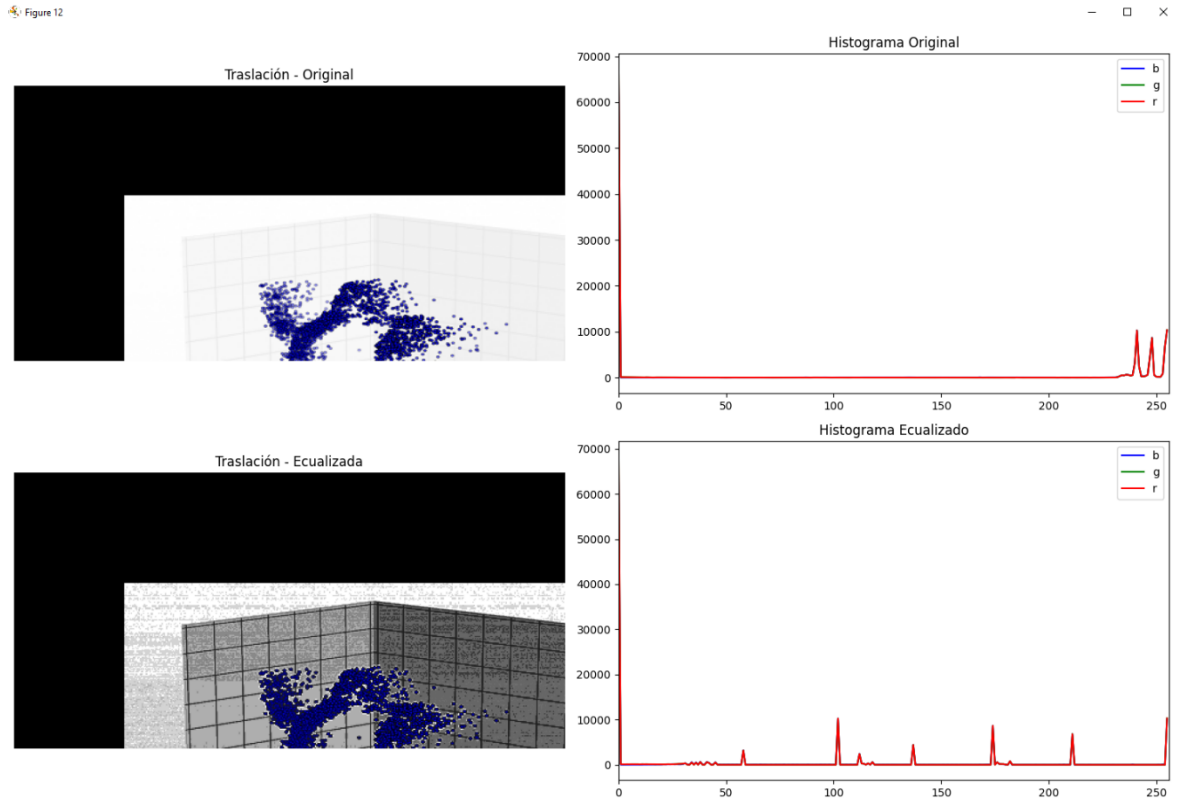
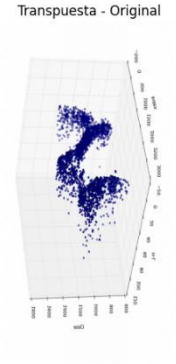


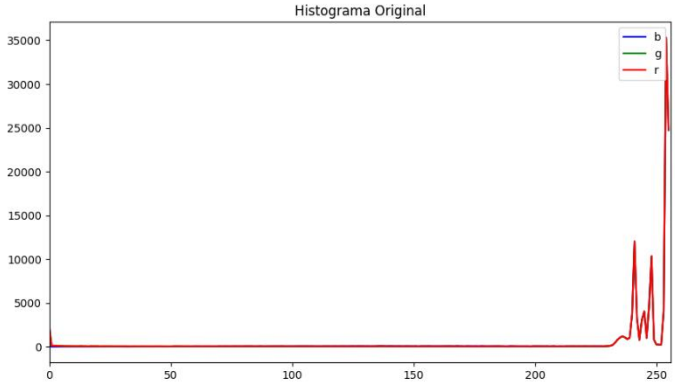
Figure 12
Traslación - Original
Histograma Original
Traslación - Ecualizada
Histograma Ecualizado
b
g
r

Figure 13

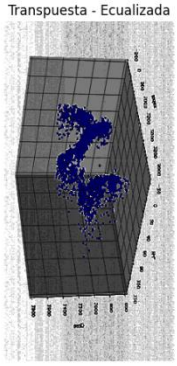
Transpuesta - Original



Histograma Original



Transpuesta - Ecualizada



Histograma Ecualizado