

## Práctica 5.1 Tratamiento Digital de Imagen

David Casillas Pérez. Modificada para Julia por Francisco J. Valverde Albacete 05 de diciembre de 2023

### Resumen

El objetivo de esta práctica es familiarizarse con las funciones y paquetes del entorno de procesado de imagen de Julia [1].

### Instrucciones

En esta práctica se proporciona un fichero "pract\_1\_1.jl" incompleto. El objetivo es que rellenes el código necesario para completar cada una de las tareas que se indican. Podéis añadir tantas líneas de código y variables como deseis, pero la solución de cada uno de los ejercicios debe tener el nombre especifico que se indica, no podéis cambiarlos. Los conocimientos para realizar dicha práctica se pueden comenzar aprendiendo en [2].

A continuación se muestra el fragmento de código que se deberá rellenar para esta primera práctica:

```
# 1. Lee las siguientes imágenes
img1 = # Leer camarama en tonos de gris
img2 = # Leer peppers en colores

## Calcula su resolución Espacial
rest_im1g1 =
println("La resolución de la imagen 1 es", rest_im1g1)
rest_im2g1 =
println("La resolución de la imagen 2 es", rest_im2g1)

# 2. Pasa la imagen 2 a escala de grises
img2_g = 

# 3. Obtén los canales R, G y B por separado de la imagen img2
R, G, B = 

# 4. Equalizar linealmente la imagen img1 e img2_g: imv = (imx - minimo(imx)) / (maximo(imx) - minimo(imx))
img1_n =
img2_g_n = 

# 5. Representa img1 y su correspondiente imagen equalizada en ix2. Pon titulos
using Plots

# 6. Representa img2 en escala de grises y su correspondiente imagen equalizada en ix2. Ponga titulos por subplot
# 7. Representa la imagen img2 a color

# 8. Representa en una figura ix3 cada canal de la imagen img2

# 9. Reduce y muestra la imagen un factor de 10 con los métodos: el más cercano, bilineal, bicubica
img1_red1 =
img1_red2 =
img1_red3 = 

# 10. Muestra las imágenes reducidas en una figura ix3 con los métodos: el más cercano, bilineal, bicubica
img1_aug1 =
img1_aug2 =
img1_aug3 = 

In [ ]: # Realiza la práctica

# añadimos paquetes
# using Pkg
# Pkg.add("Interpolations")
# Pkg.add("Images")
# Pkg.add("Imageview")
# Pkg.add("TestImages", "TestImages", "Plots", "Interpolations")

In [ ]: # Leemos las siguientes imágenes
img1 = testimage("cameraman"), # Leer camarama en tonos de gris
img2 = testimage("peppers"), # Leer peppers en colores

println("El tamaño de la imagen 1 es $(size(img1)) bytes")
println("La resolución de la imagen 1 es $(rest_im1g1) pixels")
rest_im1g1 =
println("El tamaño de la imagen 2 es $(size(img2)) bytes")
println("La resolución de la imagen 2 es $(rest_im2g1) pixels")

# 1. Lee los siguientes imágenes
img1 = testimage("cameraman"), # Leer camarama en tonos de gris
img2 = testimage("peppers"), # Leer peppers en colores

println("El tamaño de la imagen 1 es $(size(img1)) bytes")
println("La resolución de la imagen 1 es $(rest_im1g1) pixels")
rest_im1g1 =
println("El tamaño de la imagen 2 es $(size(img2)) bytes")
println("La resolución de la imagen 2 es $(rest_im2g1) pixels")

# 2. Pasa la imagen 2 a escala de grises
img2_g = Gray.(img2)
display(img2_g)



In [ ]: # 3. Obtén los canales R, G y B por separado de la imagen img2
channels = channelview(img2)
# return matrix 3x52x52x3. the last Z dimensions correspond
# to color channels. image, filter each channel by the first param of matrix

G_im1g1 = channels[1,:,:]
G_im2g1 = channels[2,:,:]
B_im2g1 = channels[3,:,:]

# Verifica las dimensiones de los canales si es necesario
println("Dimensiones del canal R: $(size(R_im1g1))") 
println("Dimensiones del canal G: $(size(G_im1g1))") 
println("Dimensiones del canal B: $(size(B_im1g1))") 

Dimensiones del canal R: (512, 512)
Dimensiones del canal G: (512, 512)
Dimensiones del canal B: (512, 512)

In [ ]: # 4. Equalizar linealmente la imagen img1 e img2_g: imv = (imx - minimo(imx)) / (maximo(imx) - minimo(imx))

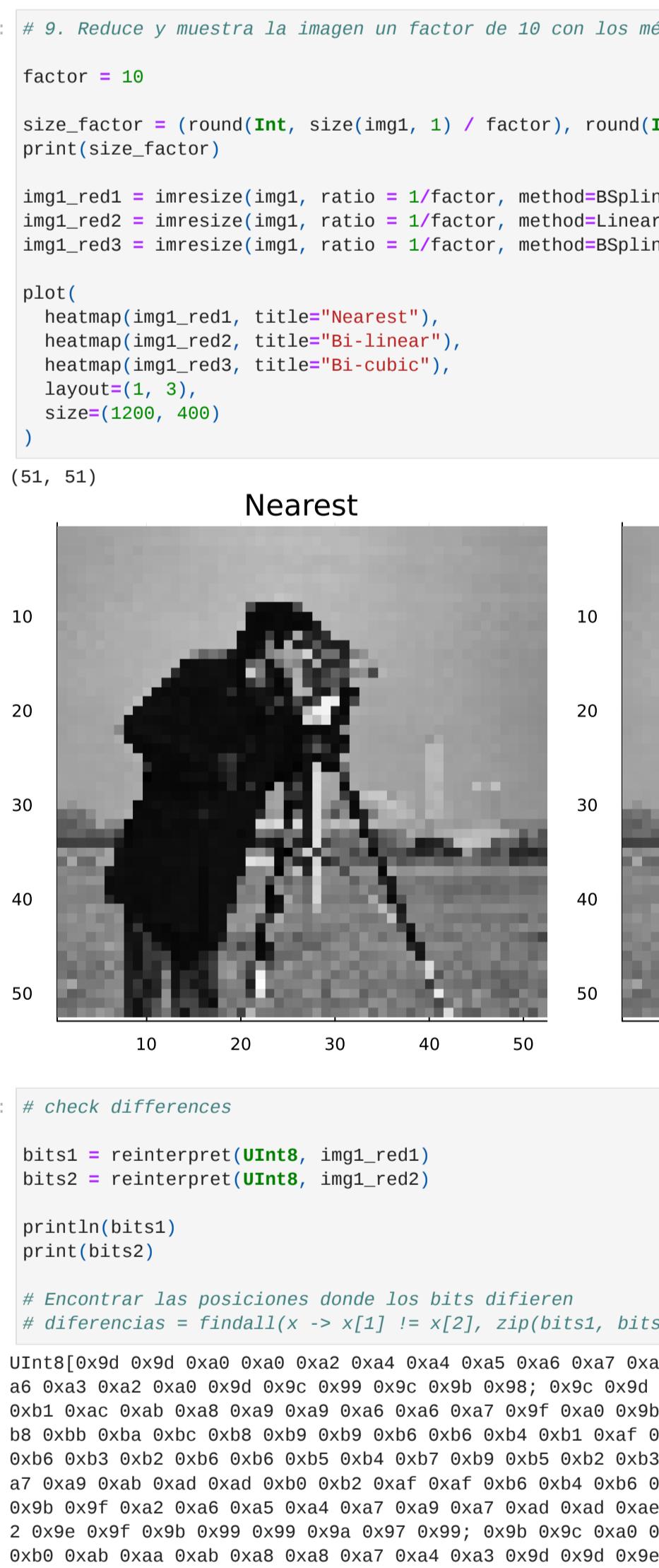
function equalize(imx)
    min = minimum(imx)
    max = maximum(imx)
    return (imx - min) ./ (max - min)
end

img1_n = equalize(img1)
img2_g_n = equalize(img2_g)

img1_n = adjust_histogram(img1, Equalization(mbins = 2^(4)))
img2_g_n = adjust_histogram(img2_g, Equalization(mbins = 2^(4)))

# 5. Representa img1 y su correspondiente imagen equalizada en ix2. Pon titulos
using Plots

# Create el grafico de ix2 con titulos
plot(
    heatmap(img1, color=:gray, title="Imagen Original"),
    heatmap(img1_n, color=:gray, title="Imagen Equalizada"),
    layout=(1, 2),
    size=(600, 400)
)



In [ ]: # 6. Representa img2 en escala de grises y su correspondiente imagen equalizada en ix2. Ponga titulos por subplot
img2_g = Gray.(img2)
display(img2_g)

# Create el grafico de ix2 con titulos
plot(
    heatmap(img2_g, color=:gray, title="Imagen Original"),
    heatmap(img2_g_n, color=:gray, title="Imagen Equalizada"),
    layout=(1, 2),
    size=(600, 400)
)



In [ ]: # 7. Representa la imagen img2 a color
plot(
    heatmap(img2, color=:gray, title="Imagen 2 a color"),
    layout=(1, 1),
    size=(600, 400),
    axis=false
)



In [ ]: # 8. Representa en una figura ix3 cada canal de la imagen img2
img2_channels = channelview(img2)
# Create el grafico de ix3 con titulos
plot(
    heatmap(img2_channels[1,:,:], title="R"),
    heatmap(img2_channels[2,:,:], title="G"),
    heatmap(img2_channels[3,:,:], title="B"),
    layout=(1, 3),
    size=(600, 400)
)




```