# Informe de la herramienta Image-Magick

A continuación se presenta la información pertinente sobre el proyecto realizado en image-magick: (T. p. = tiempo promedio)

Tiempo total del procesamiento de las imagenes: 67.84768748283386 s.

T. p. para convetir a escala de grises: 0.9325311183929443 s.

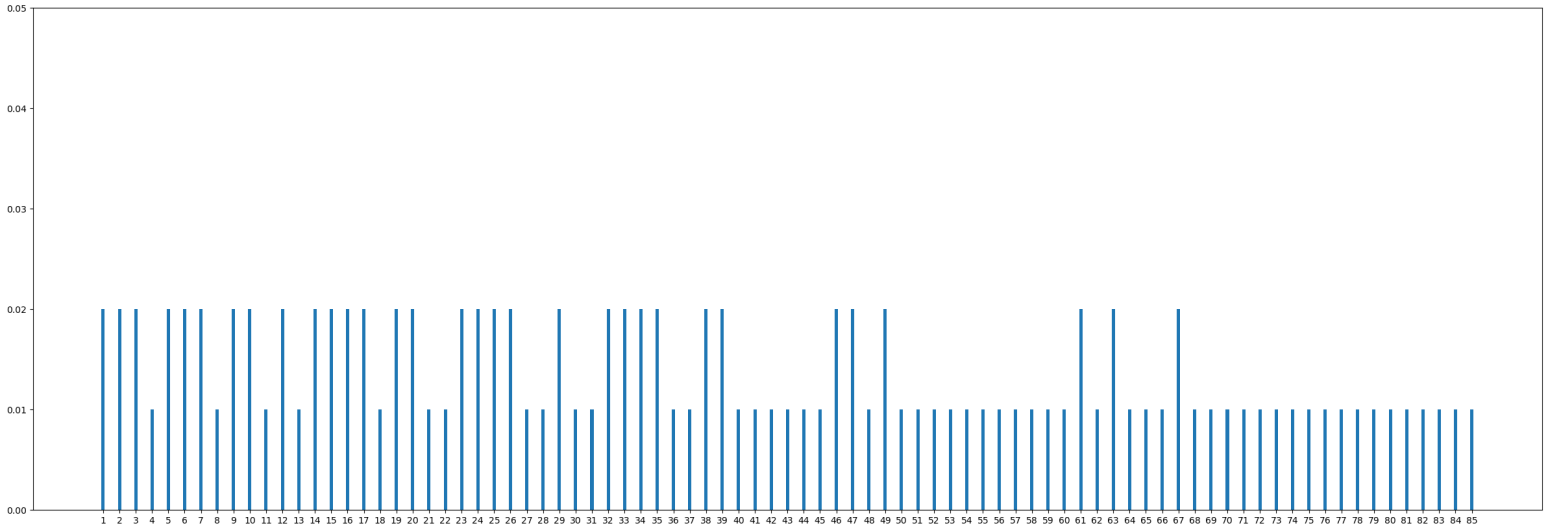
T. p. para el histograma de las imágenes originales: 65.65589785575867 s.

T. p. para el histograma de las imágenes en escala de grises: 1.2575101852416992 s.

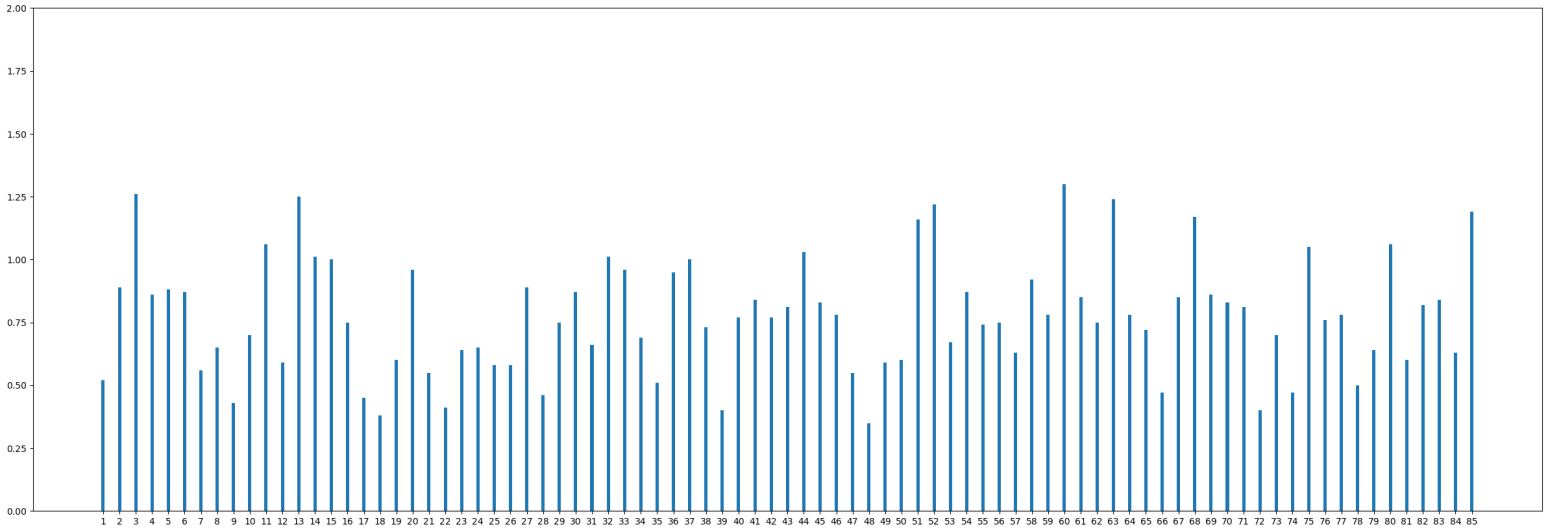
El peso original de las imágenes es:  1.2289628982543945

El peso en escala de grises de las imágenes es:  0.9997129440307617

## Gráficos




Tiempo para calcular un histograma de 1 canal




Tiempo para calcular el histograma de 3 canales

Realizado por: Byron Calva

# Código fuente

```
import os
import time
from time import sleep
from reportlab.pdfgen import canvas
from reportlab.lib.pagesizes import A4
import pylab as pl
import numpy as np
from reportlab.lib.utils import ImageReader

inicio = time.time()
sunflowers = os.listdir('sunflower/')
sunflowers_gray = os.listdir('sunflower-gray/')
conversion_a_gris_tiempo = 0
histograma_3_canales_tiempo = 0
histograma_1_canal_tiempo = 0
dic_colores = {}
dic_grises = {}

def convertir_a_gris(sunflowers, conversion_a_gris_tiempo):
    ## Se leen todos los archivos
    print('*********************************************\
        +'*********************************************')
    print("Convirtiendo imágenes a escala de grises...")
    n = 0;
    for sunflower in sunflowers:
        inicio_gris = time.time()
        orden = 'convert \'sunflower/' + sunflower \
            + '\' -set colorspace Gray -separate -average \'sunflower-gray/' \
            + sunflower + '\''
        os.system(orden)
        conversion_a_gris_tiempo += (time.time()-inicio_gris)
        n += 1
    print('==> Tiempo promedio para convertir a escala de grises ==> '\
        ,str(conversion_a_gris_tiempo/n), 'segundos.')
    print("Conversión finalizada")
    print('*********************************************\
```

Realizado por: Byron Calva

```
        +'*****************************************')
    return conversion_a_gris_tiempo

def obtener_histograma_3_canales(sunflowers,\
    histograma_3_canales_tiempo, dic_colores):
    print('*********************************************\
        +'*****************************************')
    print("Obteniendo los histogramas de las imágenes a color...")
    n = 1
    for sunflower in sunflowers:
        inicio_histo_colores = time.time()
        carpeta = sunflower[0: len(sunflower)-4: 1]
        existe = os.path.exists('histogramas-3-canales/Histogramas-'+ carpeta)
        if existe != True:
            os.system('mkdir histogramas-3-canales/Histogramas-'+ carpeta)
        orden = 'convert sunflower/'+ sunflower\
            +' -define histogram:unique-colors=true -format %c histogram:histogramas-3-ca
            + carpeta + '/histograma.gif'
        os.system(orden)
        orden = 'convert  histogramas-3-canales/Histogramas-'\
            +carpeta+'/histograma.gif -strip -resize 200% -separate histogramas-3-canales/H
            + carpeta+'/canal-%d.gif'
        os.system(orden)
        dic_colores[str(n)] = round((time.time()-inicio_histo_colores),2)
        histograma_3_canales_tiempo += (time.time()-inicio_histo_colores)
        n += 1
    print('==> Tiempo promedio para obtener histogramas a color ==> '\
        ,str(histograma_3_canales_tiempo/n), 'segundos.')
    print("Histogramas obtenidos")
    print('*********************************************\
        +'*****************************************')
    return histograma_3_canales_tiempo

def obtener_histograma_1_canal(sunflowers, histograma_1_canal_tiempo, dic_grises):
    print('*********************************************\
        +'*****************************************')
    print("Obteniendo los histogramas de las imágenes en escala de grises...")
```

Realizado por: Byron Calva

```
    n = 1
    for sunflower in sunflowers:
        inicio_histo_gris = time.time()
        carpeta = sunflower[0: len(sunflower)-4: 1]
        orden = 'convert sunflower/'+ sunflower\
            +' -colorspace Gray -define histogram:unique-colors=false histogram:histograma
                + carpeta +'.gif'
        os.system(orden)
        os.system('mv histograma-'+ carpeta +'.gif histogramas-1-canal/')
        dic_grises[str(n)] = round((time.time()-inicio_histo_gris),2)
        histograma_1_canal_tiempo += (time.time()-inicio_histo_gris)
        n += 1
    print('==> Tiempo promedio para obtener histogramas en gris ==> '\
        ,str(histograma_1_canal_tiempo/n), ' segundos.')
    print("Histogramas obtenidos")
    print('*****************************************\
        +'*****************************************')
    return histograma_1_canal_tiempo

def calcular_tamanio(ruta):
    """Get size of a directory tree in bytes."""
    tamanio = 0
    for path, dirs, files in os.walk(ruta):
        for archivo in files:
            tamanio += os.path.getsize(os.path.join(path, archivo))
    return tamanio

def graficar_colores(dic_colores):
    f, ax = pl.subplots(figsize=(30,10))
    x = np.arange(len(dic_colores))
    pl.bar(x, dic_colores.values(), align='center', width=0.2)
    pl.xticks(x, dic_colores.keys())
    ymax = 2
    pl.ylim(0, ymax)
    pl.savefig('Figura2', bbox_inches='tight',pad_inches=0.1)

def graficar_grises(dic_grises):
```

Realizado por: Byron Calva

```
    f, ax = pl.subplots(figsize=(30,10))
    x = np.arange(len(dic_grises))
    pl.bar(x, dic_grises.values(), align='center', width=0.2)
    pl.xticks(x, dic_grises.keys())
    ymax = 0.05
    pl.ylim(0, ymax)
    pl.savefig('Figura1', bbox_inches='tight',pad_inches=0.1)

conversion_a_gris_tiempo = convertir_a_gris(sunflowers, conversion_a_gris_tiempo)

histograma_3_canales_tiempo = obtener_histograma_3_canales\
(sunflowers, histograma_3_canales_tiempo, dic_colores)

histograma_1_canal_tiempo = obtener_histograma_1_canal\
    (sunflowers_gray, histograma_1_canal_tiempo, dic_grises)

peso_original = calcular_tamanio('sunflower/')/1048576
print('Las imágenes originales poseen un peso de: ' , peso_original,'MB')
peso_gris = calcular_tamanio('sunflower-gray/')/1048576
print('Las imágenes en escala de gris poseen un peso de: ' , peso_gris,'MB')

tiempo_total = (time.time()-inicio)

print('El programa se ha ejecutado en', tiempo_total, 'segundos.')

graficar_grises(dic_grises)
graficar_colores(dic_colores)

print('Generando PDF')
documento = canvas.Canvas('Reporte Image-Magick', pagesize=A4)
documento.setFont("Helvetica", 23)
documento.setFillColor('red')
documento.drawString(100,800,'Informe de la herramienta Image-Magick')
documento.setFont("Helvetica", 15)
documento.setFillColor('black')
x = 15
y = 770
```

Realizado por: Byron Calva

```
documento.drawString(x,y,\
    'A continuación se presenta la información pertinente sobre el proyecto realizado en')
documento.drawString(x,y-20,'       image-magick: (T. p. = tiempo promedio)')
documento.drawString(x,y-40,'Tiempo total del procesamiento de las imagenes: '\
    +str(tiempo_total)+' s.')
documento.drawString(x,y-60,'T. p. para convetir a escala de grises: '\
    +str(conversion_a_gris_tiempo)+' s.')
documento.drawString(x,y-80,'T. p. para el histograma de las imágenes originales: '\
    +str(histograma_3_canales_tiempo)+' s.')
documento.drawString(x,y-100,\
    'T. p. para el histograma de las imágenes en escala de grises: '\
    +str(histograma_1_canal_tiempo)+' s.')
documento.drawString(x,y-130,'El peso original de las imágenes es:  '\
    +str(peso_original))
documento.drawString(x,y-150,'El peso en escala de grises de las imágenes es:  '\
    +str(peso_gris))
documento.setFont("Helvetica", 23)
documento.setFillColor('red')
documento.drawString(x+240,y-180,'Gráficos')
documento.drawImage("Figura1.png", 10, y-400, width=580, height=200)
documento.setFont("Helvetica", 15)
documento.setFillColor('blue')
documento.drawString(x+120,y-425,'Tiempo para calcular un histograma de 1 canal')
documento.drawImage("Figura2.png", 10, y-630, width=580, height=200)
documento.drawString(x+120,y-650,'Tiempo para calcular el histograma de 3 canales')
documento.setFillColor('black')

documento.drawString(x,20,'Realizado por: Byron Calva')

documento.showPage()
documento.setFont("Helvetica", 23)
documento.setFillColor('red')
documento.drawString(210,800,'Código fuente')
documento.setFont("Helvetica", 15)
documento.setFillColor('black')
documento.drawString(x,20,'Realizado por: Byron Calva')
f = open("Script_Image_Magick.py","r")
```

Realizado por: Byron Calva

```
lineas = f.readlines()
f.close()
for linea in lineas:
    y = y-20
    if y < 40:
        documento.showPage()
        documento.setFont("Helvetica", 15)
        documento.drawString(x,20,'Realizado por: Byron Calva')
        y = 770
    documento.drawString(x,y,linea[0:len(linea)-1])
documento.save()
print('PDF generado')
```

Realizado por: Byron Calva