Visión por computadora
Reporte - Tarea 4: Histogram Equalization

Jaime Rangel Ojeda
zS22000513@estudiantes.uv.mx

Maestría en Inteligencia Artificial

IIIA Instituto de Investigaciones en Inteligencia Artificial
Universidad Veracruzana
Campus Sur, Calle Paseo Lote II, Sección 2a, No 112
Nuevo Xalapa, Xalapa, Ver., México 91097

9 de mayo de 2023

1. Objetivo

Histogram Equalization es un proceso fundamental en el procesamiento de imágenes, consiste en una técnica para ajustar los valores de píxeles en una imagen para mejorar su contraste, haciendo que las intensidades se ajusten.

Típicamente, un histograma se asemeja a una distribución normal, pero la ecualización se enfoca en una distribución uniforme.

2. Metodología

Necesitamos crear nuestro histograma con base en una fórmula matemática, obteniendo un arreglo en 1d y computar el histograma de la imagen basándonos en la frecuencia de los valores de los píxeles.

$$P_x(j) = \sum_{i=0}^{j} P_x(i)$$

Posteriormente, necesitamos una suma acumulativa de nuestro histograma

Una vez teniendo nuestra suma acumulativa de nuestro histograma necesitamos normalizar con rangos entre 0 - 255.

$$S_k = \sum_{j=0}^{k} \frac{n_j}{N}$$

Finalmente necesitamos redimensionar nuestra imagen original a 1d y ajustar los valores con los resultados de nuestra suma acumulativa normalizada para modificar la intensidad de nuestros valores en la imagen. Tomando nuestro arreglo en 1d y usando el indexador del vector para buscar valores relacionados con el vector del histograma acumulativo.

3. Resultados

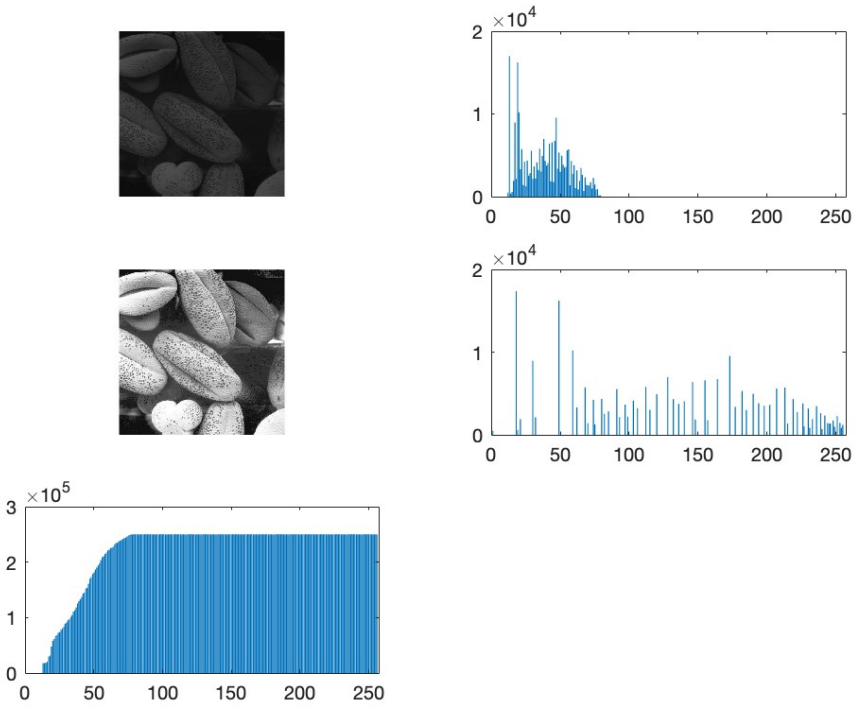

Figura 1: Resultados de histogram equalization

4. Conclusiones

La parte importante acerca de este proceso es que histogram equalization ajusta las intensidades a un nivel global tomando en cuenta todos los píxeles haciendo un reajuste en la distribución de los pixeles dentro del rango de 0 a 255 dando como resultado una imagen mejorada.

5. Código

```
close all

[img_1, img_2, img_3] = read_images();

[with_1,height_1] = get_sizes(img_1);
[with_2,height_2] = get_sizes(img_2);
[with_3,height_3] = get_sizes(img_3);

hist_img1 = zeros(1,256);
hist_img1 = generate_histogram(img_1,with_1,height_1,hist_img1);

%Cumulative histogram function
sumhist_img1 = generate_cumulative_hist(hist_img1);

%Normalize
%cast it back to uint8 since we can't use floating point values in images
normhist_img1 = renormalize(sumhist_img1);

%Reshape final result
final_img1 = reshape_vector(normhist_img1,with_1,height_1,img_1);

%Hist of the final image
hist_final_img1 = zeros(1,256);
hist_final_img1 = generate_histogram(final_img1,with_1,height_1,hist_final_img1);

% hist_img2 = zeros(1,256);
% hist_img2 = generate_histogram(img_1,with_1,height_1,hist_img2);
%
% hist_img3 = zeros(1,256);
% hist_img3 = generate_histogram(img_1,with_1,height_1,hist_img3);

plot_figures(img_1,hist_img1,final_img1,hist_final_img1)

function plot_figures(original_img,hist_img,final_img1,hist_final_img1)
    subplot(2,2,1)
    imshow(original_img)
    subplot(2,2,2)
    bar(hist_img)
```

```
    subplot(2,2,3)
    imshow(final_img1)
    subplot(2,2,4)
    bar(hist_final_img1)
end

function img_new = reshape_vector(hist,x,y,img)
    flat = reshape(img,1,[]);
    img_new = hist(flat);
    img_new = reshape(img_new,[x,y]);
end

function h = renormalize(hist)
    nj = (hist - min(hist)) * 255;
    N = max(hist) - min(hist);
    hist = nj / N;
    h = cast(hist,"uint8");
end

function [rows,cols] = get_sizes(img)
    [rows, cols] = size(img);
end

function h = generate_cumulative_hist(hist)

   for i = 1:255
        a = hist(i);
        b = hist(i + 1);
        hist(i + 1) = a + b;
   end
   h = hist;
end

function h = generate_histogram(img,x,y,hist)
    for i=1:x
        for j=1:y
            color_pixel = img(i,j);

            if(color_pixel == 0)
                color_pixel = color_pixel + 1;
```

```
            end

            hist(color_pixel) = hist(color_pixel) + 1;
        end
    end
    h = hist;
end

function [img_a,img_b,img_c] = read_images()
    img_a = imread('Fig3.15(a)1top.jpg');
    img_b = imread('Fig3.15(a)2.jpg');
    img_c = imread('Fig3.15(a)3.jpg');
end


```

[Wal]

Referencias

[Wal] Tory Walker. *Histogram Equalization in Python from Scratch*. URL: https://medium.com/hackernoon/histogram-equalization-in-python-from-scratch-ebb9c8aa3f23#:~:text=Histogram%20Equalization%20is%20one%20of,more%20equal%20across%20the%20board..