Procesamiento Digital de Imágenes
Práctica 2
"Algoritmos: Dithering Aleatorio, Dithering Ordenado y Dithering Floyd-Steinberg"

Sergio Alvarado Ramos
Posgrado de Ingeniería
Universidad Nacional Autónoma de México
Ciudad de México, México
Email: sergioar98@hotmail.com

Mariana Rodríguez Castañeda
Posgrado de Ingeniería
Universidad Nacional Autónoma de México
Ciudad de México, México
Email: mariana.rodriguez.c@comunidad.unam.mx

***Resumen*—Se expone el uso de tres algoritmos, los cuales tienen la finalidad de representar una imagen con menos información pero, sin perder las características escenciales. Los tres algoritmos propuestos siguen un mismo principio, la difusión de las intensidades. Aprovechan las características de la visión humana, y el hecho de que esta se concentra principalmente en ciertos atributos de la información recibida.**

I. Objetivos

- Cuantizar una imagen con 1 bit de profundidad
- Implementar dithering aleatorio
- Implementar dithering con matrices de 2x2 y 4x4
- Implementar dithering con difusión del error de Floyd-Steinberg

II. Introducción

La optimización de los recursos siempre ha sido un tema fundamental en nuestra sociedad. El ahorro de los recursos en las imagenes, comenzó en su momento como un asunto comercial, ya que se necesitaba que las pantallas de televisón fueran accesible para el consumidor promedio, y costear una televisión de alta calidad y empleando grandes cantidades de hardware, no era una opción.

Actualmente sigue siendo escencial la optimización de recursos, probablemente más que nunca, ya que las razones son más importante. Cada línea de código y cada componente de hardware en una sociedad donde la información es infinita y la demanda al alza, necesita un manejo mesurado. Es por esto que en el trabajo presente se expondrán tres métodos para optimizar recursos en las imagenes.

La profundidad de una imagen tiene que ver con la resolución que se utiliza para representarla, esta representación tiene como finalidad tener más información a revelar, sin embargo, vivimos en un mundo analógico y la información necesita ser comprimida para poder utilizarla optimizadamente, es decir, no perder información valiosa y tampoco necesitar muchos recursos. En una computadora el número de bits por pixel muestra la resolución o profundidad de nuestra imagen.

En el trabajo aquí presentado, escogimos una imagen cuantizada uniformemente con una recolución de bits por pixel pequeña. Al tener una imagen con estas características los colores que se pueden representar para nuestra imagen son muy escasos. En el ejemplo de 1 bit, solo se puede representar cero (negro) o uno (blanco). Entre más bits por pixel más colores se pueden reflejar. Lo que se quiere demostrar en este trabajo es que una imagen que ha perdido mucha información y que es de pequeño tamaño puede representar las características principales de la misma con el correcto procesamiento.

Los tres algoritmos son; Dithering Aleatorio: Agrega ruido uniforme a la imagen para dar la ilusión de mayor profundidad en la imagen; Diethering Ordenado: Este algoritmo utiliza una matriz para saber cuando modificar los pixeles de una imagen, como su nombre lo dice difumina, evitando que haya altos contrastes, la matriz sirve como referencia de un numero promedio que debe haber de intensidad entre los pixeles de la imagen; Dithering Floyd-Steinberg: Este algoritmo se basa en la utilización de una máscara para modificar los pixeles de la imagen, por cada iteración cambiará un pixel central y los de su alrededor inmediato.

III. Desarrollo

La implementación se desarrollo en Matlab, se creó una matriz por cada ejemplo, las cuatro primeros algoritmos se probarion en el misimoi ciclo de iteraciones. El algoritmo Random Dithering"solo utiliza operaciones de suma, se realizaron dos pruebas. Para el algoritmo .ºrdered Dithering"se especificaron dos matrices, con numeros sucesivos pero separados lo más que se podía uno de otro, la primera imágen se realizó con una matriz de 2x2 por lo que la separacion no era mucha, sin

embargo, también se realizó la prueba con una matriz 4x4 en la que la separación de los valores estaba más alejada.

Por último se desarrollo el algoritmo de Floyd-Steinberg, para este algorítmo se ocupó otro ciclo de iteraciones ya que las modificaciones no se realizan pixel por pixel como en los anteriores, sino que también se modifican los pixeles de abajo a la derecha, lo que provoca que el comienzo de la iteración deba ser diferente. para el ejemplo presentado se utilizó: $\delta = 1/16\gamma = 5/16\beta = 3/16\alpha = 7/16$

IV. Resultados

Figura 1. Original

Figura 2. Binary

Figura 3. Random Dithering

Figura 4. Ordered Dithering

Figura 5. Ordered Dithering

Figura 6. Floyd-Steinberg Dithering

En las imágenes presentadas hay tres zonas que se pueden observar con detenimiento para entender los algoritmos propuestos. La primera zona es el cabello y telefono de la jóven, aquí se puede notar que el dithering random de la primer imagen, no hizo un buen trabajo detectando la diferencia entre el pelo y el teléfono, en el segundo ejemplo sí marca la diferencia.

Por otro lado con el algoritmo de dithering ordenado no es difícil detectar esta zona. Sin embargo, en la zona de la torre eiffel entre el primer algoritmo utilizando una matriz 2x2 y una matriz 4x4, se aprecia mejor en el 2x2. Con respecto a la zona del chip, la segunda hizo un mejor trabajo.

Por último, es fácil de ver que el algoritmo de Floyd-Steinberg es el que hizo un mejor trabajo, puesto que recupera más información.

V. Código

```
I = imread('LenaOrig.jpg');
I=rgb2gray(I);

figure('Name','Original');
imshow(I)

[largo,alto]=size(I);
Bin=I;
Dithering_Random=I;
Dithering_Ordered=I;
Dithering_Ordered2=I;

D=[3.0 1.0;
    0.0 2.0]/4;

D4=[15.0 7.0 13.0 5.0;
    3.0 11.0 1.0 9.0;
    12.0 4.0 14.0 6.0;
    0.0 8.0 2.0 10.0]/16;

for x=1:largo
    for y=1:alto
        Bin(x,y) =   round(double(I(x,y))/255);
        Dithering_Random(x,y) =  round(double(I(x,y))/255+0.3*rand(1));
        Dithering_Ordered(x,y) =  round(double(I(x,y))/255+0.2*D(mod(y,2)+1,mod(x,2)+1));
        Dithering_Ordered2(x,y) = round(double(I(x,y))/255+0.2*D4(mod(y,4)+1,mod(x,4)+1));
    end
end

figure('Name','Binary');
Bin=logical(Bin);
imshow(Bin)

figure('Name','Random2');
Dithering_Random=logical(Dithering_Random);
imshow(Dithering_Random)

figure('Name','Ordered1');
Dithering_Ordered=logical(Dithering_Ordered);
imshow(Dithering_Ordered)

figure('Name','Ordered2');
Dithering_Ordered2=logical(Dithering_Ordered2);
imshow(Dithering_Ordered2)


Dithering_FS=I;

for x=2:largo-1
    for y=1:alto-1

        e = double(I(x,y))-255*round(double(I(x,y))/255);
        Dithering_FS(x,y) = 255*round(double(I(x,y))/255);
        I(x+1,y)   = I(x+1,y)   + e*7/16;
        I(x-1,y+1) = I(x-1,y+1) + e*3/16;
        I(x,y+1)   = I(x,y+1)   + e*5/16;
        I(x+1,y+1) = I(x+1,y+1) + e*1/16;

    end
end
%Dithering_FS(:,y)=0;
%Dithering_FS(1,:)=0;
%Dithering_FS(x,:)=0;
figure('Name','Dithering_FS');
Dithering_FS=logical(Dithering_FS);
imshow(Dithering_FS)
```

VI. Conclusiones

Con está práctica aprendimos técnicas que permiten al ojo humano visualizar detalles a pesar de que solo se tengan píxeles negros y blancos. Esta percepción de tonos grises se logra al agrupar los píxeles negros y blancos, entre mayor número de píxeles negros se tienen en un área se observan tonos más oscuros.

Con la imagen de Lena se puede observar que se perciben mejor los detalles con el algoritmo Floyd-Steinberg, sin embargo este algoritmo también consume más tiempo que Ordered Dithering y Random Dithering, por lo que para una imagen muy grande puede resultar más conveniente utilizar otro algoritmo.

Se puede afirmar que todos los algoritmos dan como resultado una imagen más agradable al ojo humano que la imagen binaria.