Práctica 2
Dithering

Josué David Cárdenas Avila
Universidad Nacional Autónoma de México
16 de marzo del 2023

Candia Marcial Uriel
Universidad Nacional Autónoma de México
16 de marzo del 2023

Luis Tonatiuh Ornelas Madrid
Universidad Nacional Autónoma de México
16 de marzo del 2023

Reyes Luna Luis Ernesto
Universidad Nacional Autónoma de México
16 de marzo del 2023

***Resumen*—El dithering es una técnica utilizada en el procesamiento de imágenes digitales para reducir los efectos de la cuantización, que puede causar una pérdida significativa de información y detalles en una imagen. El dithering introduce ruido aleatorio en la imagen en áreas de transición entre diferentes valores de intensidad, utilizando una matriz de dithering que se suma al valor de intensidad original de cada píxel. Esto produce la ilusión de una mayor profundidad de bits o niveles de intensidad, mejorando así la calidad visual de la imagen. El dithering se utiliza comúnmente para mejorar la calidad visual de las imágenes en dispositivos de baja resolución o en pantallas de visualización, así como para reducir el tamaño de los archivos de imagen sin perder demasiada información o detalles en la imagen.**

***Index Terms*—Técnica, Procesamiento de imágenes, Cuantización, Pérdida de información, Reducción de niveles de intensidad, Ruido aleatorio, Matriz de dithering o matriz de patrón, Profundidad de bits, Niveles de intensidad, Mejora de la calidad visual, Dispositivos de baja resolución, Pantallas de visualización, Reducción del tamaño de archivos, Detalles de la imagen.**

I. Introducción

El dithering es una técnica que se emplea para minimizar los efectos de la cuantización en las imágenes digitales, ya que esta puede causar una disminución significativa en la información y los detalles que aparecen en una imagen cuando se reduce el número de niveles de intensidad. Con el objetivo de simular una mayor profundidad de bits o niveles de intensidad, el dithering introduce ruido en la imagen en zonas de transición entre diferentes valores de intensidad. Este ruido se produce mediante una matriz llamada matriz de dithering o matriz de patrón, que se suma al valor de intensidad original de cada píxel. Al aplicar esta matriz, los valores de intensidad de los píxeles se desplazan ligeramente hacia arriba o hacia abajo en una escala discreta de valores, lo que produce la impresión de que la imagen tiene una mayor profundidad de bits.

En el procesamiento de imágenes, el dithering es una técnica comúnmente usada para mejorar la calidad visual de las imágenes en dispositivos de baja resolución o en pantallas de visualización. Además, también se puede aplicar el dithering para reducir el tamaño de los archivos de imagen sin perder demasiada información o detalles en la imagen.

II. Desarrollo

II-A. Cuantice una imagen con 1 bit de profundidad

Para esta actividad utilizaremos la siguiente imágen:

Para cuantizar una imagen a 1 bit de profundidad, es necesario reducir los niveles de intensidad de cada pixel de la imagen a solo dos valores posibles, es decir, a 0 o a 1. Esto significa que cada pixel solo puede ser blanco o negro, y la imagen resultante será una imagen binaria. Mathlab nos ofrece múltiples herramientas en su paquete de procesamiento de imágenes, pero para esta actividad usaremos las siguientes funciones: Primero importaremos la imagen mediante imread:

```
% Cargar la imagen
I = imread('bird.png');
```

Lo siguiente que haremos sera Cuantizar la imágen a 1 bit de profundidad mediante:

```
% Cuantizar la imagen a 1 bit de
   profundidad
threshold = graythresh(I);
bw_image = im2bw(I, threshold);
```

Usamos graythresh para calcular automáticamente un umbral para una imagen en escala de grises mediante el método de Otsu. El umbral calculado separa los píxeles de la imagen

en dos grupos: uno con valores de intensidad mayores que el umbral y otro con valores menores o iguales al umbral. Es decir permite segmentar automáticamente una imagen en escala de grises en dos grupos, permitiendo la eliminación del ruido y la mejora de la calidad de la imagen. Por otra parte la función im2bw la utilizamos para convertir una imagen en escala de grises en una imagen binaria (blanco y negro) mediante la selección de un umbral.

Finalmente mostramos la imagen cuantizada a 1 bit de profundidad.

```
% Mostrar la imagen cuantizada
imshow(bw_image);
```

II-B. Implemente el método de dithering aleatorio

Para realizar esta actividad vamos a partir del código anterior, lo primero que haremos sera :

```
% Convertir la imagen en blanco y negro
    en una matriz de doble precision
bw_double = im2double(bw_image);
```

La conversión de la imagen a una matriz de doble precisión se realiza para evitar la pérdida de precisión en los cálculos. Lo siguiente que haremos sera generar una matriz aleatoria del mismo tamaño que la imágen para posteriormente aplicarla a la imágen:

```
% Crear una matriz de dithering aleatoria
    del mismo tamano que la imagen
dither = rand(size(bw_double));
% Sumar la matriz de dithering a la
    imagen en blanco y negro
dithered_image = bw_double + dither;
% Convertir la imagen dithered_image de
    vuelta a una imagen binaria
dithered_bw_image = im2bw(dithered_image,
0.5);
```

Y finalmente mostramos el resultado:

```
% Mostrar la imagen cuantizada y dithered
imshow(dithered_bw_image);
```

II-C. Implemente el método de dithering con matrices de 2x2 y 4x4

Para implementar el método con una matriz de 2x2 haremos lo siguiente: Primero importar la imágen y aplicarle le aplicaremos im2double para convertir la imagen en formato de punto fijo (por ejemplo, uint8 o uint16) a punto flotante de doble precisión, normalizando los valores de los píxeles en el rango [0,1].

```
% Cargar la imagen en escala de grises
I = imread('C:\Users\josue\Desktop\
Tarea2\bird.png');
I = im2double(I);
```

Lo siguiente que haremos sera crear nuestra matriz 2x2, para definir los valores que esta tendrá usaremos la ordenación por dispersión (scatter), esta consiste en asignar valores de dithering a cada celda de la matriz en función de su posición relativa en la matriz.

```
% Definir la matriz de dithering de 2x2
dither_matrix = [0 2; 3 1];
```

Lo siguiente que haremos sera preparar las variables:

```
% Obtener el tamano de la imagen
[n, m] = size(I);
% Inicializar la imagen ditherizada
dithered_image = zeros(n, m);
```

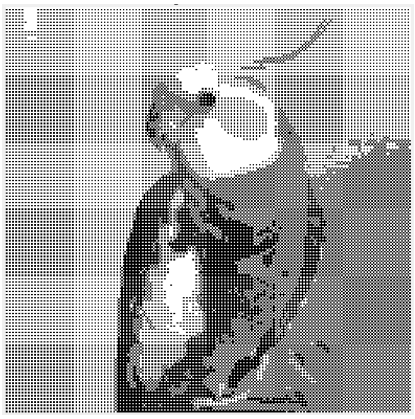
Ahora aplicaremos el dithering:

```
% Iterar sobre cada bloque de 2x2
y aplicar dithering
for i = 1:2:n-1
    for j = 1:2:m-1
    % Obtener el bloque de 2x2
```

```
    block = I(i:i+1, j:j+1);

    % Obtener los valores de dithering
    para el bloque de 2x2
    dither_values = dither_matrix * 0.25;

    % Calcular la imagen ditherizada para
    el bloque de 2x2
    dithered_block = (block >
    dither_values);

    % Asignar la imagen ditherizada del
    bloque de 2x2 a la imagen ditherizada
    dithered_image(i:i+1, j:j+1) =
    dithered_block;

    end
end
```

Primero obtenemos el bloque 2x2 de nuestra imágen, luego normalizamos los valores para que esten en un rango de 0 a 1, posteriormente compramos los valores de la imágen con los valores ditherisados, si el valor del pixel es mayor que el valor correspondiente en la matriz de dithering, se asigna un valor 1 a dithered block, y si es menor, se asigna un valor 0 y finalmente se crea la imagen ditherizada.
Finalmente imprimimos las imagenes para ver el resultado.

```
% Mostrar las imagenes original y
    ditherizada
figure;
subplot(1,2,1);
imshow(I);
title('Imagen original');
subplot(1,2,2);
imshow(dithered_image);
title('Imagen ditherizada 2x2');
```

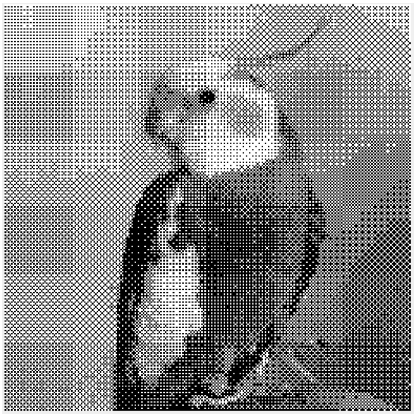


Para el siguiente caso haremos prácticamente lo mismo solo que con una matriz de 4x4:

```
% Cargar la imagen en escala de grises
I = imread('C:\Users\josue\Desktop\
Tarea2\bird.png');
I = im2double(I);
% Definir la matriz de dithering de 4x4
dither_matrix = [0 8 2 10; 12 4 14 6;
3 11 1 9; 15 7 13 5];
```

En este caso utilizamos una matriz predefinida conocida como matriz de Bayer. Posteriormente seguimos los mismos pasos que ya explicamos anteriormente:

```
% Obtener el tamano de la imagen
[n, m] = size(I);

% Inicializar la imagen ditherizada
dithered_image = zeros(n, m);

% Iterar sobre cada bloque de 4x4 y
    aplicar dithering
for i = 1:4:n-3
    for j = 1:4:m-3
    % Obtener el bloque de 4x4
    block = I(i:i+3, j:j+3);
    % Obtener los valores de dithering
    para el bloque de 4x4
    dither_values = dither_matrix *
    0.0625;
    % Calcular la imagen ditherizada para
    el bloque de 4x4
    dithered_block = (block >
    dither_values);
    % Asignar la imagen ditherizada del
    bloque de 4x4 a la imagen ditherizada
    dithered_image(i:i+3, j:j+3) =
    dithered_block;
    end
end
% Mostrar las imagenes original y
    ditherizada
figure;
subplot(1,2,1);
imshow(I);
title('Imagen original');
subplot(1,2,2);
imshow(dithered_image);
title('Imagen ditherizada 4x4');
```



II-D. Implemente el método de dithering con difusión del error de Floyd-Steinberg

```
% Cargar la imagen.
I = imread('C:\Users\josue\Desktop\Tarea2\bird.png');
% Determinamos el numero de colores para la imagen de salida.
% En este caso deseamos solo dos colores.
num_colors = 2;
% Calculamos el umbral para la cuantizacion
threshold = max(I(:)) / num_colors;
```

Max(I(:)) devuelve el valor máximo de la matriz I, es decir, el valor máximo posible que puede tener un pixel en la imagen. Dividimos este valor por el número de colores deseado (en este caso 2) dando como resultado el umbral para la cuantización, que se utiliza para determinar si cada pixel debe ser asignado al valor más cercano de blanco o negro.

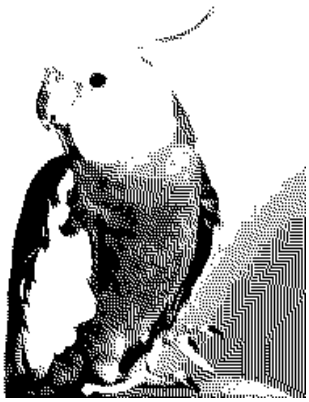
```
    % Creamos una matriz vacia para la imagen ditherizada.
I_dithered = zeros(size(I));
%hacemos un loop sobre cada pixel en la imagen, iniciacndo por el de la
%esquina superior izquierda.
for y = 1:size(I, 1)
    for x = 1:size(I, 2)
        % Obtenemos el valor del pixel actual.
        old_value = I(y, x);
        %Quantizamos el valor del pixel a negro o blanco.
        if old_value >= threshold
            new_value = 255;
        else
            new_value = 0;
        end
```

Lo que hacemos en esta parte es: si el valor del pixel en la imagen original es mayor o igual que el umbral (threshold), se asigna un valor de 255 (blanco) al pixel en la imagen dithered (I dithered). Si el valor del pixel en la imagen original es menor que el umbral, se asigna un valor de 0 (negro) al pixel en la imagen dithered.

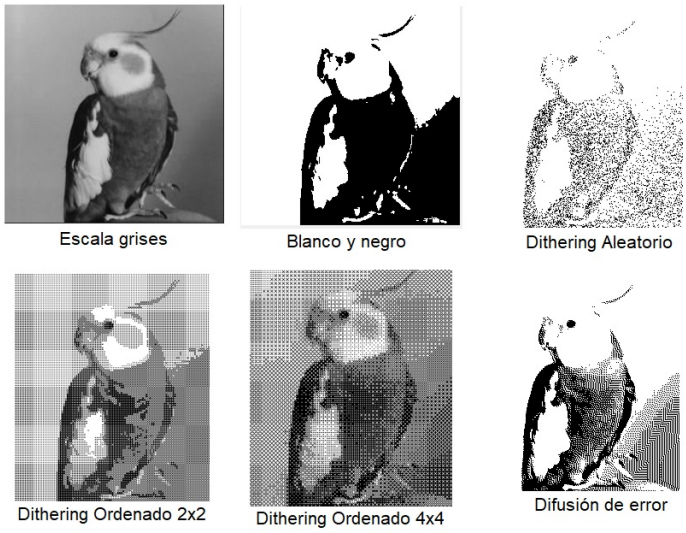
```
    %Asignamos el valor del pixel ditherado.
        I_dithered(y, x) = new_value;
        %Calculamos el error de cuantizacion.
        quant_error = old_value - new_value;
        % Difundimos el error de cuantificacion a los pixeles vecinos.
        if x < size(I, 2)
            I(y, x+1) = I(y, x+1) + quant_error * 7 / 16;
        end
        if y < size(I, 1) && x > 1
            I(y+1, x-1) = I(y+1, x-1) + quant_error * 3 / 16;
        end
        if y < size(I, 1)
            I(y+1, x) = I(y+1, x) + quant_error * 5 / 16;
        end
        if y < size(I, 1) && x < size(I, 2)
            I(y+1, x+1) = I(y+1, x+1) + quant_error * 1 / 16;
        end
    end
end
```

Esta parte del código se encarga de difundir el error de cuantificación a los píxeles vecinos, utilizando la técnica de difusión de error de Floyd-Steinberg. La idea es que el error de cuantificación se distribuya a los píxeles vecinos de manera que el dithering se vea más natural y no tan abrupto. Al píxel inmediatamente a la derecha, se le asigna el 7/16 del error. Al píxel inmediatamente debajo y a la izquierda, se le asigna el 3/16 del error. Al píxel inmediatamente debajo, se le asigna el 5/16 del error. Al píxel inmediatamente debajo y a la derecha, se le asigna el 1/16 del error. Esto se realiza dentro del bucle que recorre todos los píxeles de la imagen, y se hace para cada píxel que se ha cuantificado. Finalmente mostramos el resultado:

```
    % Mostramos el resultado
    imshow(I_dithered);
```



II-E. Compare los resultados de los 5 métodos



Se observó que la cuantización de un bit de profundidad genera un mayor contraste en la imagen, pero a expensas de la pérdida de información y forma en la imagen debido al color negro que se destaca demasiado.

En cuanto a las técnicas de dithering, se encontró que tanto el dithering aleatorio como la difusión de error de Floyd Steinberg producen resultados similares, aunque la escala de grises en la difusión de error es mayor. Por otro lado, el dithering con matrices de 2x2 y 4x4 también generan resultados similares, pero la escala de grises en el método de matrices de 4x4 permite una mejor visualización de la imagen.

No hay una técnica de dithering que sea mejor que la otra. La elección de la técnica de dithering depende del tipo de imagen y del efecto deseado. Cada técnica tiene sus fortalezas y debilidades y se debe elegir la técnica adecuada para obtener el resultado deseado.

III. Conclusión

Cárdenas Avila Josué David: En conclusión, se pueden aplicar distintos métodos para reducir la profundidad de una imagen, y así cuantizarla. Uno de ellos es la cuantización binaria, que reduce la profundidad a 1 bit. Sin embargo, esta técnica suele generar una pérdida significativa de información y una imagen final de baja calidad. Para solucionar esto, se pueden aplicar métodos de dithering, que añaden ruido a la imagen para simular tonalidades intermedias. Se implementaron dos de estos métodos: el dithering aleatorio, que utiliza una distribución aleatoria de ruido para generar la imagen ditherizada, y el dithering con matrices de 2x2 y 4x4, que utiliza una matriz predefinida para añadir el ruido a la imagen. La elección del método dependerá de la imagen y el efecto deseado.

Reyes Luna Luis Ernesto: En esta práctica se pudo aplicar y comprender los métodos vistos en clase para mejorar la visualización de las imágenes. En general, esta práctica permitió comprender la importancia de aplicar técnicas de dithering para mejorar la calidad de las imágenes y cómo elegir la técnica adecuada en función del resultado deseado.

Si se desea una imagen con un aspecto más natural y menos artificial, el dithering aleatorio es una buena opción. Esta técnica utiliza una distribución aleatoria de ruido para simular tonos intermedios y produce resultados más suaves y naturales, especialmente en áreas degradadas o gradientes suaves.

Por otro lado, si se busca una imagen con un aspecto más uniforme y predecible, el dithering con matrices es una buena opción. Esta técnica utiliza una matriz predefinida para agregar ruido a la imagen, lo que produce patrones visibles y uniformes en la imagen final.

Candia Marcial Uriel: Con la realización de esta practica se pudo comprender de mejor manera los distintos tipos de dithering que existen, y su implementación en este caso en matlab.

En conclusión, el dithering es una buena técnica para el mejoramiento de la calidad visual en una imagen, al reducir el numero de colores utilizados, cada tipo de dithering tiene sus propias ventajas y desventajas, sin embargo con Floy-Steinberg podemos obtener una imagen mas suave ya que este es una mejora del dithering, pero al ser mas complejo puede ser mas lento.

Ornelas Madrid Luis Tonatiuh: Esta practica nos mostró una primera instancia de la forma de un preprocesado de una imagen. Haciendo uso de dos metodos básicos como son la cuantización y el Dithering. Dos metodos que nos ayudan a reinterpretar una imagen digital reduciendo su tamaño manteniendo en ciertas medias las caracteristicas principales de la misma.

IV. Referencias

Wikiwand - Algoritmo de Floyd-Steinberg. https://www.wikiwand.com/es/Algoritmo$_{d}e_{F}loyd$ – $Steinberg.Accedidoel16demarzode2023$.

.Algoritmo de Floyd-Steinberg - frwiki.wiki". Sección de quince vingtos - frwiki.wiki. https://es.frwiki.wiki/wiki/Algorithme$_{d}e_{F}loyd$ – $Steinberg.Accedidoel16demarzode2023$.

Çurso de Procesado de Imagen (c)GPI". Universitat de València.
https://www.uv.es/gpoei/eng/Pfc$_{w}eb/generalidades/dithering.htm.$
$Accedidoel16demarzode2023.$