Digital Image Processing Project 3
Image Compression by Bit-plane Slicing

Introduction:

In this project we are going to load an image file into MATLAB and slice it bitwise into 8 planes, then try to compress it by eliminating the unnecessary bits to reduce the whole image size.

Each digital image consists basically of a number of pixels (rows x columns), each pixel has a specific intensity value, which is stored on the computer's memory as a binary number.

Usually this binary number consists of 8 digits; for example 11111111 represents white, while 00000000 represents black.

Bit-Plane Slicing:

In this Project we will load an Image of a Jordanian Dinar, separate it into 8 different images, each one is a bit layer, i.e. the first layer is by masking all bits (making them zero) but keeping the LSB (Least Significant Bit) unchanged. And so on for all other bits.

This operation can be done in MATLAB using the >> bitand(image,mask); command. i.e. we will, bitwise, AND each pixel with a specific mask to generate each layer.

Remember that the AND result is 1 when both inputs are ones, and 0 elsewise.

For example to obtain the 1st layer (for the LSB) we use: bitand(image,1);

Image Compression:

After that we will try to remove some bits in order to reduce the image size.

The Original Image:

It has the size of 344 x 620 = 213280 pixels.
Each pixel has an 8-bit intensity.

The theoretical total size of it must be 624 kb which can be seen for bmp file.

But because our image here is a jpg image; it has only 62.5 kb.

So that, we will try here to reduce this 62.5 kb, as small as possible.

**MATLAB Program and Results:**

On the next page, you can see the MATLAB code for this operation.

The following pages are the results of the program.

You can find that eliminating the least significant 4 bits, doesn’t affect the overall image resolution.

That means we can reduce the image size into half; if the grayscale image size = 62.5/3 kb = 20.8 kb

**The compressed image will need only 10.4 kb, which is very good.**

Now, take a look at the results on the following pages.
clear all, close all, clc

% load and display image
I=imread('C:\Users\Mahmood\Desktop\Dinar.jpg');
imshow(I)

% Grayscale
Igray=rgb2gray(I);
figure, imshow(Igray)

% Size
[row , col] = size(Igray);

% Bitwise Separation
Inew = zeros(row,col);
mask = 1;
for a = 1:8
    for i = 1:1:row
        for j = 1:1:col
            Inew(i,j)=bitand(Igray(i,j),mask);
        end
    end
    mask = mask * 2;
    figure, imshow(Inew)

%figure, imagesc(Inew)
%colormap gray
end

% Compressing Image
% by eliminating the least significant four bits
% then the picture will need only 4 bits
% (16 levels) instead of 8 bits (256 levels)
% i.e. half storage size :)

mask = 240;
for i = 1:1:row
    for j = 1:1:col
        Inew(i,j) = bitand(Igray(i,j),mask);
    end
end
figure, imshow(uint8(Inew))

% The Difference
Idif = Igray - uint8(Inew);
figure, imagesc(Idif), colormap gray
| **RGB (colored) image** | 3 Layers (red, green & blue)  
Each pixel 3 Intensity values  
$3^x(xxxxxxxx)$ |
|------------------------|---------------------------------------------------------------|

| **Grayscale image** | 1 layer  
Each pixel one Intensity value  
of 8 binary digits  
$Xxxxxxx$ |
|---------------------|---------------------------------------------------------------|

<table>
<thead>
<tr>
<th><strong>1st bit only (LSB)</strong></th>
<th>0000000x</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>2nd bit only</strong></th>
<th>000000x0</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>3rd bit only</strong></th>
<th>00000x00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Image</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>4th</td>
<td><img src="4th_bit.png" alt="Image" /> 0000x000</td>
</tr>
<tr>
<td>5th</td>
<td><img src="5th_bit.png" alt="Image" /> 000x0000</td>
</tr>
<tr>
<td>6th</td>
<td><img src="6th_bit.png" alt="Image" /> 00x00000</td>
</tr>
<tr>
<td>7th</td>
<td><img src="7th_bit.png" alt="Image" /> 0x000000</td>
</tr>
<tr>
<td>8th (MSB)</td>
<td><img src="8th_bit.png" alt="Image" /> x0000000</td>
</tr>
</tbody>
</table>
Compressed Image
Using only the most significant 4 bits
5\text{th}, 6\text{th}, 7\text{th}, 8\text{th} bits only
\text{xxxx0000}
Half storage space

The difference between the grayscale image and the compressed one.
\text{0000xxxx}

**Conclusions:**

- In this project we have used the MATLAB functions to do bit-plane slicing on an image in order to compress its size.

- Bit-plane slicing can be achieved by bitwise ANDing each pixel in the image with a mask of the 2 multipliers.

- In this project we have compressed the Dinar image into its half size, by eliminating the least significant 4 bits, and the image remains quite good.

- Bit-plane slicing is an efficient method to understand digital images and compress them.

**Attachments:**

- With this report the following files are attached:
  1. *ImageProcessProj3.m* (the MATLAB program)
  2. *Dinar.jpg* (the image file)