**Assignment 2 Solution  
Computer Vision (CS-559)**

**Name: Dhaval Harish Sharma  
Red ID: 824654344**

1. **(a) What is an interchange format? (b) Give three examples of interchange formats. (c) What is the signature of a PGM format? Give two examples of the signature.**

**Solution:**Image file formats are standardized means of organizing and storing digital images. There are many different formats for storing image data. Some have been designed to fulfil a particular need; others have been designed around a particular operating system or environment. Interchange formats are one of the image file formats. They are designed to facilitate the exchange of image data between users. Also, they can be used with different hardware and software. Image compression is a standard feature of interchange formats.

There are many different types of interchange formats. They are listed and explained as follows:   
1. GIF (Graphics Interchange Format): GIF came about for images on WWW. The GIF is in normal use limited to an 8 bit palette or 256 colours. GIF is most suitable for storing graphics with few colours, such as simple diagrams, shapes, logos, and cartoon style images, as it uses LZW lossless compression, which is more effective when large areas have a single colour, and less effective for photographic or dithered images. Due to GIF's simplicity and age, it achieved almost universal software support.

2. JFIF (JPEG File Interchange Format): JFIF also appeared on WWW. The JFIF filename extension is JPG or JPEG. Nearly every digital camera can save images in the JPEG/JFIF format, which supports eight-bit greyscale images and 24-bit colour images. JPEG applies lossy compression to images, which can result in a significant reduction of the file size. Applications can determine the degree of compression to apply, and the amount of compression affects the visual quality of the result.

3. PNG (Portable Network Graphics): It is a replacement for GIF for legal reasons. The PNG file format supports eight-bit paletted images (with optional transparency for all palette colours) and 24-bit truecolor (16 million colours) or 48-bit truecolor with and without alpha channel - while GIF supports only 256 colours and a single transparent colour. Compared to JPEG, PNG excels when the image has large, uniformly coloured areas. Even for photographs – where JPEG is often the choice for final distribution since its compression technique typically yields smaller file sizes – PNG is still well-suited to storing images during the editing process because of its lossless compression.

The full form of PGM is Portable Greyscale Format. There are two signatures of greyscale format, (i) P2 (Storage Type – ASCII) and (ii) P5 (Storage Type – Raw Byte).

For example, we can have a P2 signature file as follows:

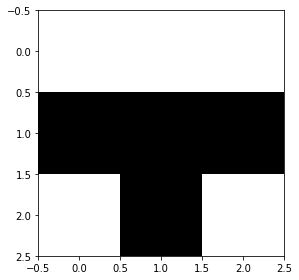
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P2 | | | | | | | |
| # An example of P2 PGM file | | | | | | | |
| 4 8 | | | | | | | |
| 255 | | | | | | | |
| 0 | 255 | 0 | 0 | 0 | 0 | 255 | 0 |
| 0 | 255 | 255 | 0 | 0 | 255 | 255 | 0 |
| 0 | 0 | 0 | 255 | 255 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The raw format is much more compact as compared to the ASCII format but the ASCII format is much easier to read than raw format. As another example, we can take a P5 signature file as follows:

|  |
| --- |
| P5 |
| # An example of P5 PGM file |
| 7 7 255 |
| xxxxxxxxxx ! xxxxxx ! xxxx !!!!! xxxx ! xxxxxx ! xxxxxxxxxx |

**2) (a) What is patterning in printing? (b) What is a dither matrix and how is it used? (c) Explain the principle and operation of error diffusion for printing.  
Solution:**Most printers produce a binary output that is, they either print a black dot or blank (white). Newspapers stimulate a grayscale by printing tiny black dots of varying sizes. The human visual system has a tendency to average brightness over small areas so that black dots and white background merge and are perceived as shades of grey. This process is called halftoning and several methods of implementing it are available.

Patterning is one of the methods of halftoning. Each pixel from the image is replaced by a certain predefined pattern. This means that each pixel replaces certain matrix in the printing page. After replacing the pixels, the actual figure looks as if it was made up of gray scale values but in fact it is just made from black and white dots. For example, let’s take a look at an example of an image having grayscale values from 0 to 9. It replaces grayscale value 5 to a 3 X 3 matrix as follows:

  
Grayscale value = 5

In this case the width and the height of the image are increased by a factor of 3 because every pixel gets replaced by a 3 X 3 matrix.

Dithering is another technique for applying halftoning. It consists of thresholding the image against a dither matrix. The dimensions of these matrices are always a power of 2. The image is enlarged by a factor of 2m, where m is the dither matrix size, to see better details. The elements of the matrix are thresholds. An example of a dither matrix for an 8 bit grayscale image is as follows:  
D2 =

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 128 | 32 | 160 |
| 192 | 64 | 224 | 96 |
| 48 | 176 | 16 | 144 |
| 240 | 112 | 208 | 80 |

This matrix is put on the image and the output pixel value becomes 255 (White) if the input pixel values are greater than the matrix element else becomes 0 (Black) if they are less than the matrix element.

Error diffusion is yet another technique used for printing the images. The first step for error diffusion is to select a threshold, let’s say 128 for an 8-bit grayscale image. If f(x, y) is greater than 128 than, the pixel at (x, y) in the output is set to white otherwise it is set to black. For pixels close to 128, the error rate is high, so in order to compensate the error, it is spread or diffused to neighboring pixels. A method for diffusing is shown below, where the error is spread across 4 neighbours that are ahead of the pixel. The values are the factors by which the error is multiplied and the result is added to the corresponding pixel.  
 (x, y) (x, y + 1)

|  |  |  |
| --- | --- | --- |
|  | | 7 / 16 |
| 3 / 16 | 5 / 16 | 1 / 16 |

(x + 1, y - 1) (x + 1, y) (x+1, y+1)

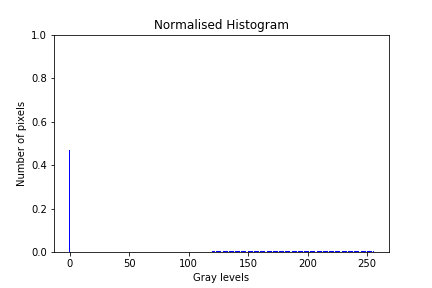
The algorithm for dithering is as follows:  
threshold = (black + white) / 2  
for (all x and y)  
{  
 if (f(x, y) < threshold)  
 {  
 g(x, y) = black  
 error = f(x, y) – black  
 }  
 else  
 {  
 g(x, y) = white  
 error = f(x, y) – white  
 }  
 f(x, y + 1) = f(x, y + 1) + 7 \* error / 16  
 f( x +1, y -1) = f(x – 1, y + 1) + 3 \* error / 16  
 f(x + 1, y) = f(x + 1, y) + 5 \* error / 16  
 f(x + 1, y + 1) = f(x + 1, y + 1) + error / 16  
}

**3) Explain conditions under which the use of a lookup table (LUT), instead of calculating the mapping pixel by pixel, reduces the computation. Express the conditions in terms of number of graylevels (L) and resolution n.  
Solution:**Changing the pixel values of the input image to produce the output image is called gray level linear mapping. It is also known as contrast stretching. For example, g(x, y) = a \* f(x, y) + b is a linear function which is applied on the input image to get an output image.

There are various ways to implement mappings which are as follows:  
Approach one:  
The algorithm below shows one of the two possible algorithms to implement the square root mapping. Let us assume that the input image is a square image with height = width = n.  
for (x = 0; x < n; x++)  
{  
 for (y = 0; y < n; y++)  
 {  
 g(x, y) = 1.5 \* √f(x, y) + 5  
 }  
}  
The above algorithm requires n2 calculations containing square roots as well as arithmetic operations which can be costly.

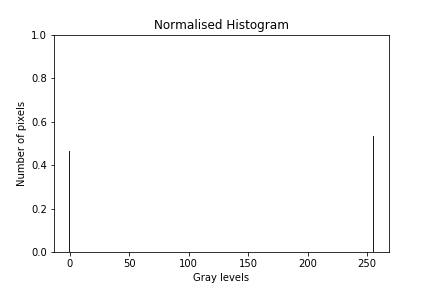
Approach two:  
Let us use a look up table instead of performing calculations each time.  
for (I = 0; I < 256; i++)  
{  
 table[i] = 1.5 \* √i + 5  
}  
for (x = 0; x < n; x++)  
{  
 for (y = 0; y < n; y++)  
 {  
 g(x, y) = table[f(x, y)]  
 }  
}  
The look up implementation requires only 256 calculations and n2 assignments. This takes less computing time than n2 calculations required by the first approach.

In general, look up table is preferred if the number of gray levels are much lesser than the number of pixels in the image, that is L << n2. This is due to the fact that instead of calculating the expression for each pixel, it is better to evaluate once for every possible value and that refer those instead of evaluating same expression again and again. This is approach is very useful in cases where the number of pixels in the image is huge such as 2048 X 2048 and the number of grayscales are just 256. Instead of calculating the expression 2048 X 2048 times, it is just evaluated 256 times and than all the pixels refer those values which reduces the time complexity by a large factor.

**4) Many (infinite number) of passes of contrast stretching with the mapping  is applied to a perfectly equalized 8-bit input image. Draw and carefully label the normalized histogram of the output image.  
Solution:**The program for this question can be found in the same folder. The working of the program is as follows:  
It begins with initializing the input image by inserting 256 rows and 256 columns with each row containing 256 colors. This process results in an 8-bit equalized image. Then, it initializes an output image with the same dimensions as the input image. It applies the process **g** on input image traversing pixel by pixel and storing the resulting value in the output image. This process is repeated 100 times (let’s assume we reach to infinite passes). The equalized histogram is then made for this output image which can be given as,   


As we go through several iterations, the gray levels lesser than 120 starts getting smaller and smaller because of getting multiplied by 0.4. After infinite passes, they eventually become 0 because the pixel value gets nearer to zero but the gray levels can only be integer values, so the system rounds off the value. Every other gray level pixel values remain as they are.

1. **5) An 8-bit image has a normalized histogram defined by  and  
   .**

**Suppose that each pixel is ANDed with 0100 0000, and is set to white if the result of ANDing is non-zero, otherwise is set to black if the result is zero (this operation is called bit-plane slicing). Draw and carefully label the normalized histogram of the resulting output image.  
Solution:**The program for this question can be found in the same folder. The working of the program is as follows:  
The program begins by initializing the input image whose number of pixels with corresponding gray level values are the same as they are given in the question. Then, it begins by performing the bit plane slicing operation on the input image. That is, if the pixel value after being ANDed with 0100 0000 becomes 0, then its resulting value is zero else its resulting value becomes 255. After getting the output image, we get a normalized histogram which looks like this;   
****

Therefore by looking at the image, we can say that the number of white pixels in the resulting output image is greater than the number of black pixels because when we perform bitwise AND on the pixels having gray level greater than 63, the resulting output becomes non zero for all the pixels as they contain 1 in the 27th bit in them. This will be true till 128 gray levels; then again we have zero output after bit plane slicing. After 191, the output of bit plane slicing becomes non zero as the input pixel value contains 1 in the 27th bit.