EXPERIMENT 1

BMP FILE FORMAT

- *To read a BMP file and flip the image along its diagonal while converting a coloured image to grayscale.*

Group 16

Arnab Mondal – 14EC35031

Sandeep Mishra – 14EC35033

INTRODUCTION

The objective of the experiment is to read a Bitmap image file(grayscale or RGB), convert it to a grayscale image, if not one already and then writing it as image on the disk after flipping along the diagonal. This experiment was done without the help of OpenCV, using only library function in C++, in order to understand how image files are created and how they are manipulated and operated upon.

Bitmap or BMP files are quite old file format used by "Windows" operating system. BMP images can range from 1 bit per pixel (thus a black and white image) to 24 bits per pixel (providing 1.67 million colours). In the experiment we used an 8 bits per pixel (Grayscale image) and 24 bits per pixel (RGB color image) formats for operating upon.

Following are two parts of a BMP file:

1. Header: It contains information about file and image. This part can be broken into two parts:

(a) File Header, which contains general information related to the file like type of the image file (BM for the Bitmap file) and Size of the file. Other fields are reserved and are not to be edited by the user.

Bitmap File Header			
Offset (hex)	Offset (dec)	Size (bytes)	Purpose
00	0	2	The header field used to identify the BMP and DIP file is 0x42 0x4D in hexadecimal, same as BM in ASCII. The following entries are possible: • **BM** Windows 3.1x, 95, NT, ... etc • **BA** OS/2 struct bitmap array • **CI** OS/2 structcolor pointer • **CP** OS/2 constcolor pointer • **IC** OS/2 struct icon • **PT** OS/2 pointer
02	2	4	The size of the BMP file in bytes
06	6	2	Reserved; actual value depends on the application that creates the image

08	8	2	Reserved; actual value depends on the application that creates the image
0A	10	4	The offset, i.e. starting address, of the byte where the bitmap image data (pixel array) can be found.

Table 1: Description of Bitmap file header contents

(b) Information Header (BITMAPINFOHEADER), which contains information about the image, like Width, Height and Bits per pixel among other data.

Bitmap Information Header Windows BITMAPINFOHEADER			
Offset (hex)	Offset (dec)	Size (bytes)	Purpose
0E	14	4	The size of this header (40 bytes)
12	18	4	The bitmap width in pixels (signed integer)
16	22	4	The bitmap height in pixels (signed integer)
1A	26	2	The number of color planes (must be 1)
1C	28	2	The number of bits per pixel, which is the color depth of the image. Typical values are 1, 4, 8, 16, 24 and 32.
1E	30	4	The compression method being used. See the next table for a list of possible values.
22	34	4	The image size. This is the size of the raw bitmap data; a dummy 0 can be given for BI_RGB bitmaps.
26	38	4	The horizontal resolution of the image. (pixel per meter, signed integer)
2A	42	4	The vertical resolution of the image. (pixel per meter, signed integer)
2E	46	4	The number of colors in the color palette, or 0 to default to 2n
32	50	4	The number of important colors used, or 0 when every color is important; generally ignored

Table 2: Description of Bitmap info header contents

2. Image Data: It contains the pixel data or the color table contents which are to be manipulated to transform the image. The data starts from the address stored in the offset field of the BITMAPINFOHEADER. It was observed that for grayscale images, offset was generally 1078, while for RGB color images it was 54. The data is stored Bottom-to-Top and Left-to-Right, i.e. the data is stored in rows which start filling at bottom first and then keep filling to the top. The row size(in bytes) should be divisible by 4, otherwise it should be padded with zeros such that the row size become divisible by 4.

$$\text{RowSize} = \left\lfloor \frac{\text{BitsPerPixel} \cdot \text{ImageWidth} + 31}{32} \right\rfloor \cdot 4$$

For flipping, contents are swapped about the diagonal of the image data while, for the color to grayscale conversion, grayscale value is calculated as,

$$Grayscale = R \times 0.30 + G \times 0.59 + B \times 0.11$$

ALGORITHM

The following functions have been used in the program:

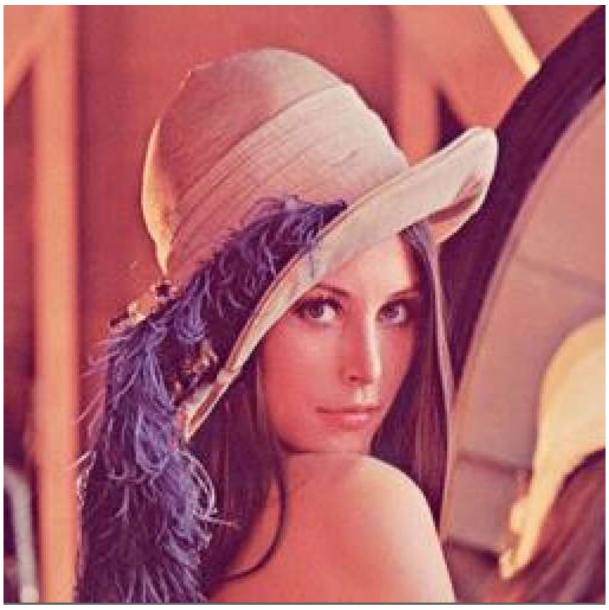
- **ReadBMP:** Function to read the BMP file and return the header. The header is saved in a data structure and the image data is loaded into memory after dynamic allocation of the memory.
  - Step 1: Reading the header.
  - Step 2: Allocating size to the array according to the information given in the header
  - Step 3: Loading the data in the memory
  - Step 4: Return the header During these operations, we also read the data that may appear as unknown (e.g. between end of the header and start of the image data OR after the image data), so that during writing, we can use them again to avoid error in writing image.

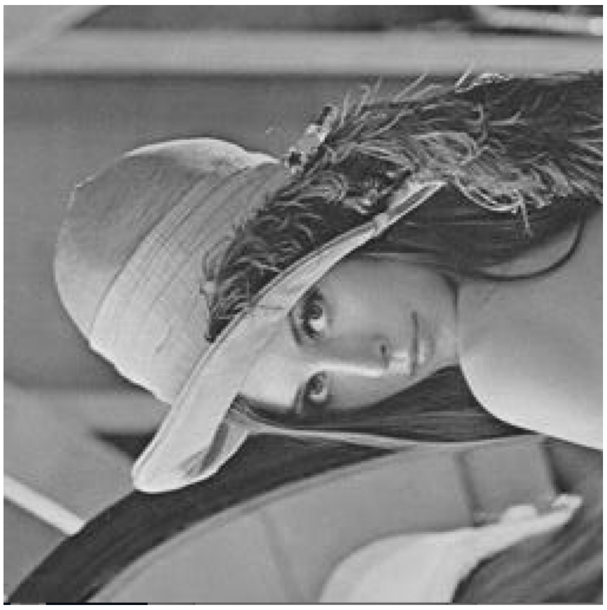
- **ConvertFlipGrayscale:** We use the formula mentioned in the introduction section and convert the image (if RGB) to grayscale. Then using simple swapping, we flip the image along the diagonal.

    - Step 1: If RGB image, calculate intensity for grayscale
    - Step 2: Flip the first channel(R) along the diagonal and store in an array
    - Step 3: If RGB, assign same value to remaining two channels as in the first channel
    - Step 4: Interchange the value of Width and Height in the header
    - Step 5: Return the array thus obtained.

- **WriteBMP:** For writing the image, we use the header and the extra data to get a correct image. Only the width, height and the pixel data can be different from the original image.
    - Step 1: Create new output file in writing mode
    - Step 2: Write the header given as input after gray-scale conversion and flipping
    - Step 3: Write the new image data and extra data(if any).

OUTPUT RESULTS


(a) Input Image


(b) Output Image

ANALYSIS

- The BMP file format, also known as bitmap image file or device independent bitmap (DIB) file format or simply a bitmap, is a raster graphics image file format used to store bitmap digital images, independently of the display device.
- BMP file format is well defined, the file format consists of file header, information header, and the pixel array. BMP file header gives important information such as height, width, size, offset and number of bits per pixel.
- Pixel array starts from different location in case of Red-Blue-Green(RGB) images and Gray-scale images. In case of RGB image the pixel array location is from 54 whereas in case of Gray-scale images it is from 1078.
- The size of images can be calculated as Header Size + Channels*Height*Width This holds true in case of RGB i.e. **54+3*256*256 = 192 KB** which is almost equal to size of the image, and in case of output gray-scale image the calculated size is **1078 + 256*256 = 65KB** which is almost equal to the size of the output image.
- For output images we have used the same file header but changed the number of channels from 3 to 1, which implies that we have converted the RGB image to gray-scale and the header-offset has also been changed to 1078 for the same reason. We have also altered the pixel array accordingly.

SOURCES

[1] https://en.wikipedia.org/wiki/BMP_file_format

[2] http://www.daubnet.com/en/file-format-bmp

[3] http://www.martinreddy.net/gfx/2d/BMP.txt