Image Processing Lab

BMP Image File Processing

GROUP-10

Submitted by:

Aditya Narayan (13EC35010)

Adarsh Kumar Kosta (13EC35008)

Objective

Write C/C++ modular functions to read, diagonally flip, and then write BMP image files. All functions must support at least 24-bit RGB, and 8-bit grayscale image formats.

1. **ReadBMP:**
   a. Input: Filename of input image
   b. Output: BMP header structure, Image pixel array loaded onto memory
2. **ConvertFlipGrayscale:**
   a. Input: Image pixel array
   b. Output: Grayscale-converted and diagonally flipped (transposed) pixel array
3. **WriteBMP:**
   a. Input: Filename of output (grayscale) image, BMP header structure, Image pixel array
   b. Output: BMP file written on disk

Use the above functions to read a 256 × 256 24-bit RGB colored Lena image, and write it as an 8-bit grayscale onto a different file on the disk.

Background

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.

The BMP file format is capable of storing two-dimensional digital images of arbitrary[citation needed] width, height, and resolution, both monochrome and color, in various color depths, and optionally with data compression, alpha channels, and color profiles.

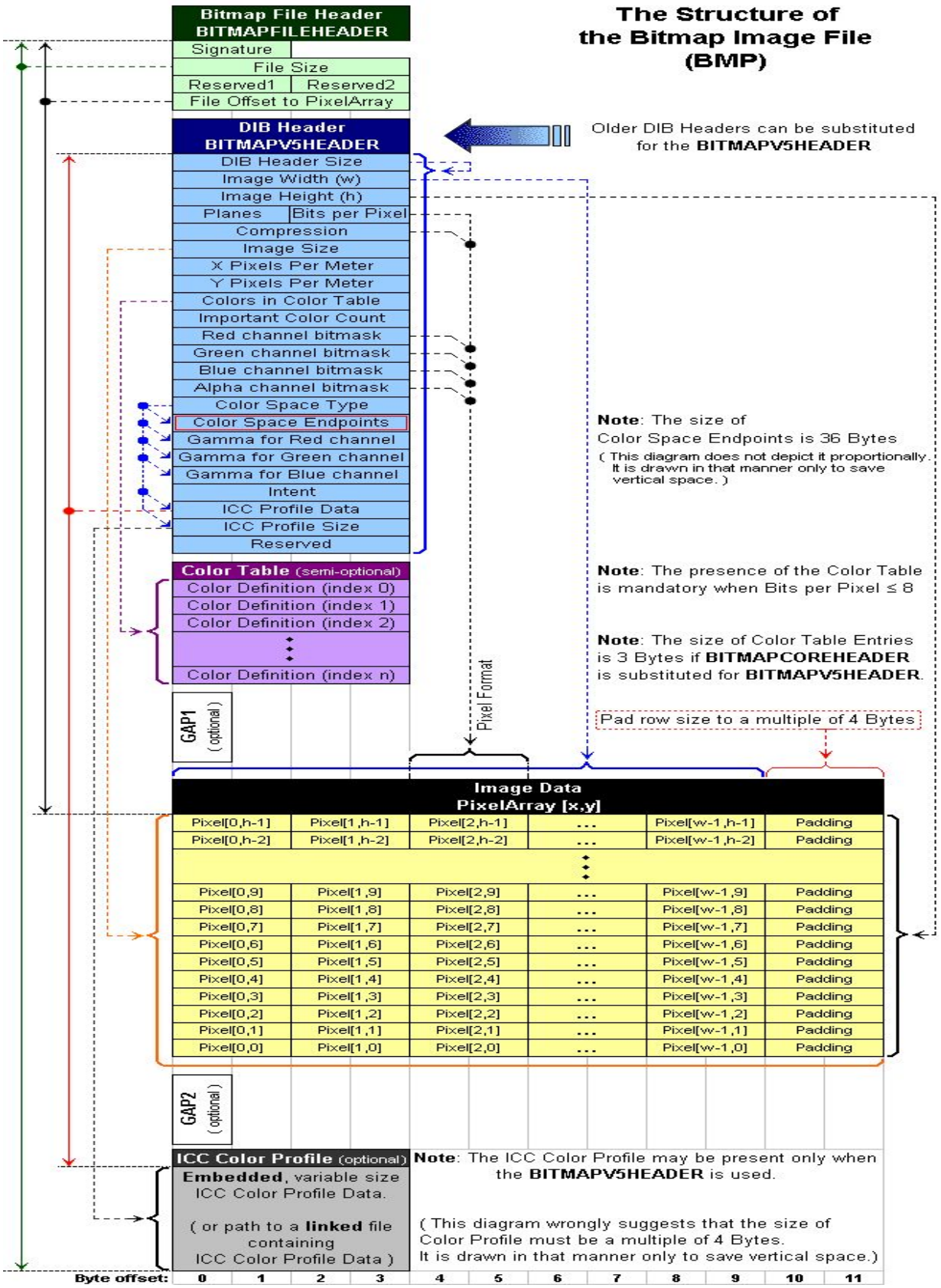
The BMP File Format

The bitmap image file consists of fixed-size structures (headers) as well as variable-size structures appearing in a predetermined sequence.

Te structure is illustrated by the table nd figure that follows.

Structure name	Optional	Size	Purpose	Comments
Bitmap file header	No	14 bytes	To store general information about the bitmap image file	Not needed after the file is loaded in memory
DIB header	No	Fixed-size (7 different versions exist)	To store detailed information about the bitmap image and define the pixel format	Immediately follows the Bitmap file header
Extra bit masks	Yes	3 or 4 DWORDs[5] (12 or 16 bytes)	To define the pixel format	Present only in case the DIB header is the BITMAPINFOHEADER and the Compression Method member is set to either BI_BITFIELDS or BI_ALPHABITFIELDS

Color table	Semi-optional	Variable-size	To define colors used by the bitmap image data (Pixel array)	Mandatory for color depths ≤ 8 bits
Gap1	Yes	Variable-size	Structure alignment	An artifact of the File offset to Pixel array in the Bitmap file header
Pixel array	No	Variable-size	To define the actual values of the pixels	The pixel format is defined by the DIB header or Extra bit masks. Each row in the Pixel array is padded to a multiple of 4 bytes in size
Gap2	Yes	Variable-size	Structure alignment	An artifact of the ICC profile data offset field in the DIB header
ICC color profile	Yes	Variable-size	To define the color profile for color management	Can also contain a path to an external file containing the color profile. When loaded in memory as "non-packed DIB", it is located between the color table and Gap1.[6]


The Structure of the Bitmap Image File (BMP)
Bitmap File Header
BITMAPFILEHEADER
Signature
File Size
Reserved1
Reserved2
File Offset to PixelArray
DIB Header
BITMAPV5HEADER
DIB Header Size
Image Width (w)
Image Height (h)
Planes
Bits per Pixel
Compression
Image Size
X Pixels Per Meter
Y Pixels Per Meter
Colors in Color Table
Important Color Count
Red channel bitmask
Green channel bitmask
Blue channel bitmask
Alpha channel bitmask
Color Space Type
Color Space Endpoints
Gamma for Red channel
Gamma for Green channel
Gamma for Blue channel
Intent
ICC Profile Data
ICC Profile Size
Reserved
Older DIB Headers can be substituted for the BITMAPV5HEADER
Note: The size of Color Space Endpoints is 36 Bytes
(This diagram does not depict it proportionally. It is drawn in that manner only to save vertical space.)
Color Table (semi-optional)
Color Definition (index 0)
Color Definition (index 1)
Color Definition (index 2)
Color Definition (index n)
Note: The presence of the Color Table is mandatory when Bits per Pixel ≤ 8
Note: The size of Color Table Entries is 3 Bytes if BITMAPCOREHEADER is substituted for BITMAPV5HEADER.
GAP1 (optional)
Pixel Format
Pad row size to a multiple of 4 Bytes
Image Data
PixelArray [x,y]
Pixel[0,h-1] Pixel[1,h-1] Pixel[2,h-1] ... Pixel[w-1,h-1] Padding
Pixel[0,h-2] Pixel[1,h-2] Pixel[2,h-2] ... Pixel[w-1,h-2] Padding
Pixel[0,9] Pixel[1,9] Pixel[2,9] ... Pixel[w-1,9] Padding
Pixel[0,8] Pixel[1,8] Pixel[2,8] ... Pixel[w-1,8] Padding
Pixel[0,7] Pixel[1,7] Pixel[2,7] ... Pixel[w-1,7] Padding
Pixel[0,6] Pixel[1,6] Pixel[2,6] ... Pixel[w-1,6] Padding
Pixel[0,5] Pixel[1,5] Pixel[2,5] ... Pixel[w-1,5] Padding
Pixel[0,4] Pixel[1,4] Pixel[2,4] ... Pixel[w-1,4] Padding
Pixel[0,3] Pixel[1,3] Pixel[2,3] ... Pixel[w-1,3] Padding
Pixel[0,2] Pixel[1,2] Pixel[2,2] ... Pixel[w-1,2] Padding
Pixel[0,1] Pixel[1,1] Pixel[2,1] ... Pixel[w-1,1] Padding
Pixel[0,0] Pixel[1,0] Pixel[2,0] ... Pixel[w-1,0] Padding
GAP2 (optional)
ICC Color Profile (optional)
Embedded, variable size ICC Color Profile Data.
(or path to a linked file containing ICC Color Profile Data)
Note: The ICC Color Profile may be present only when the BITMAPV5HEADER is used.
(This diagram wrongly suggests that the size of Color Profile must be a multiple of 4 Bytes. It is drawn in that manner only to save vertical space.)
Byte offset: 0 1 2 3 4 5 6 7 8 9 10 11

Algorithm

The algorithm of reading a BMP file without using OpenCV involves many steps and subsequent analysis.

As illustrated by the BMP File format above, the Bitmap file header data, DIB header data and the colour table data is used to extract the features of the image such as Height, Width, RGB or Grayscale etc and then used for further processing of the image.

The code utilizes 3 functions and takes the input and output image filenames as command line arguments. The functions described with their **Pseudocode** as follows:

ReadBMP:
Input: Filename of input image
Output: BMP header structure, Image pixel array loaded onto memory

- Open file. Return if file cannot be read
- Read Bitmap File Header into approprite variables (Signature, Filesize, Pixel Array Offset).
- Read DIB Header data into appropriate variables (Height, Width, Planes, Compression Method, Depth etc).
- if Depth == 8 (i.e. Grayscale image)
    - for i = 0...HEIGHT-1
        - for j = 0...WIDTH-1
            - Read single-channel pixel value of (HEIGHT - 1 - i)*th* row and j*th* column.
- else (i.e. Depth = 24, RGB image)
    - for i = 0...HEIGHT-1
        - for j = 0...WIDTH-1
            - Read three-channel pixel value of (HEIGHT - 1 - i)*th* row and j*th* column.
- Return Headers and Image Pixel Array

ConvertFlipGrayscale:

Input: Image pixel array
Output: Grayscale converted and diagonally flipped (transposed) pixel array

- `If Depth == 24:  (Convert RGB image to Grayscale using BT-709)`
    - `for i = 0...HEIGHT-1`
        - `for j = 0...WIDTH-1`
            - `grayscale[i][j] = (red[i][j] * 0.2126 + green[i][j] * 0.7152 + blue[i][j] * 0.0722);`
    - `ColorPaletteSize = 4*256  (Using RGBA32 format)`
    - `FileSize = (FileSize - PixelArraySize) + PixelArraySize / 3 + PaletteSize;`
    - `Depth = Depth/3;`
    - `PixelArrayOffset += PaletteSize;`
- `for i = 0...HEIGHT-1`
    - `for j = 0...i`
        - `Swap grayscale[i][j] and grayscale[j][i]`
- `Swap height and width`
- `Swap horizontalResolution and verticalResolution`

WriteBMP:

Input: Filename of output (grayscale) image, BMP header structure, Image pixel array

Output: BMP file written on disk

- Write BMP file header (Signature, NewFileSize, NewPixelArrayOffset).
- Write DIB Header (DIB Header Size, Height, Width, Planes, Compression Method, ImageSize etc.)
- for i = 0...255
    - Write the ith pixel of the colour palette in RGBA format. colourPalette[i] = (R=i, G=i, B=i, A=0)
- for i = 0...HEIGHT-1
    - For j = 0...WIDTH-1
        - Write single-channel pixel value of (HEIGHT - 1 - i)th row and jth column.

Results

Input Image

Output Image

Analysis

Aditya Narayan [13EC35010]

1. Three channel RGB images can be converted to grayscale images using different algorithms. The most popular ones are:
    a. *Averaging*: `Gray = (Red + Green + Blue)/3`
    b. *BT-709*: `Gray = (Red*0.2126 + Green*0.7152 + Blue*0.0722)`
    c. *BT-601:* `Gray = (Red*0.299 + Green*0.587 + Blue*0.114)`

2. The different BT standards were developed to account for the priority with which the human eye puts on different colours. The human perceives green more strongly than red, and red more strongly than blue. A simple averaging of the three channels would reduce the luminance (brightness) of the image.
   BT-709 is the most widely used standard while digital cameras may rely on BT-601. We use the BT-709 standard in the experiment.

3. There exists padding within the pixel array to ensure that each row of the pixel array is rounded to a multiple of four. This allows for the processor to fetch the entire row in sets of 32-bit DWORDs, at the expense of a minor wastage of space.

4. There exists a colour palette in a 8-bit single channel BMP file. This allows us to map any of the 256 intensity levels of a pixel to a set of 256 colours (RGBA format). For grayscale 8-bit single channel BMP files, each individual intensity level of the pixel is mapped to a colour with equal RED, GREEN and BLUE intensities.

Analysis

Adarsh Kumar Kosta [13EC35008]

1. The BMP or Bitmap Image Format consists of an array of data with the 2D Image pixel array embedded into a 1D array along with some extra dependency data and description about the image.

2. The Image Pixel array is scanned in a up down fashion instead of the usual Raster Scan fashion. Thus the first pixel element is at the leftmost bottom of the array proceeding to the right and then upwards scanning all rows.

3. The image pixel array must have row length a multiple of 4 as most processors read data half-word at a time. Thus, if the size is not a multiple of 4 then extra padding is added to the rows to make it a multiple of 4.

4. Apart from the Headers (Bitmap and DIB) the BMP file also consists of a color palette which defines the colors used in the image. The standard used for defining the color palette is RGBA32.

5. The RGB image is converted into grayscale image using some standard formulae which uses weighted means of the RGB components to compute the grayscale intensity value. We used the BT-709 standard which is the most popular and was giving better results compared to other standards.

   Grayscale = 0.2126*R +0.7152*G + 0.0722*B