Digital Image Processing with Java – 2 Dimensional Arrays

We will be modifying an existing program. It looks scarey – lots of things we don’t know. But we can figure it out. The attached code will take a color jpeg image and convert it to a black and white jpeg image. The idea is as follows:

Any jpeg image is really a 2 dimensional array of color pixels. A pixel is the individual dot that makes up the image. An image that is 480 x 640 is made up of 480 x 640 = 307, 200 pixels . Each pixel is composed of three numbers from the RGB (red-green-blue) scale. Each number can range from 0 (indicating no color) to 255 (indicating full saturation). So a fully saturated red pixel would be (255,0,0), a fully saturated green pixel would be (0,255,0).

Some things to note: the color white would have a pixel rgb value of (255,255,255) and the color black would have a pixel rgb value of (0,0,0). We can create pixels colors like that as follows:

 Color white=new Color(255,255,255);

 Color black=new Colow(0,0,0);

We should note that we get various shades of gray by specifying RGB values that are the same:

 Color gray\_but\_almost\_black= new Color(10,10,10);

 Color gray\_but\_almost\_white= new Color( 240,240,240);

Again, referring to the attached code, we access each pixel as follows:

 Color c=image.getRGB(row,column);

We can access individual values of RGB as follows:

int red= c.getRed();

int green=c.getGreen();

int blue=c.getBlue();

The attached code reads a color image and converts it to black and white. There are many algorithms that can do this

 int red = (int)(c.getRed() \* 0.299);

 int green = (int)(c.getGreen() \* 0.587);

 int blue = (int)(c.getBlue() \*0.114);

Color newColor = new Color(red+green+blue,red+green+blue,red+green+blue);

 image.setRGB(j,i,newColor.getRGB());

Lets follow the logic here:

* get red, green and blue values

 int red = (int)(c.getRed() \* 0.299);

 int green = (int)(c.getGreen() \* 0.587);

 int blue = (int)(c.getBlue() \*0.114);

* create a gray-scale pixel based upon these value. Remember, a gray pixel will have the same value for red, green and blue. But what value should this be ? One possibility would be to average the three values. In the example provided, they have weighted each color differently; this is a commonly used formula to compensate for bias that exists in our own eyes.

Color newColor = new Color(red+green+blue,red+green+blue,red+green+blue);

* update the pixel to contain the new gray pixel values

 image.setRGB(j,i,newColor.getRGB());

You should read through the rest of the code.

Assignment:

1. Download an image from

http://www.emunix.emich.edu/~sverdlik/mysteryBird.jpg

1. Convert it to a gray-scale image using the attached code
2. A goal of image processing is to remove as much unimportant information as you can. This makes it easier to identify what remains. Removing color (and leaving black and white) is one way to remove information. But this still may leave too much information. Frequently, one wishes to discretize a black and white image. The most extreme case would have pixels that are only black and white (values 0,0,0 or 255,255,255); this process is called *threshholding*

Lets simplify the notation , and say if we are speaking in gray-scale we specify only one value (say 137) instead of giving a triple (137,137,137).

For threshholding, a threshold value is selected. Lets say for example, we pick a threshold of 130. Then if a pixel value is greater than or equal to 130, it is assigned a value of 255, otherwise, it is assigned a value of 0.

But how does one pick the threshold value ? A common way is to create a histogram of pixel values. That is, we create an array of integers:

 int hist[ ]=new int[256];

Initialize this array to all zeroes. Then read each pixel in the gray-scale image. If you get a pixel value of 197, then add 1 to hist[197].

After you have read all the pixels, find the pixel value that occurred the most (or in other words, find the index j that maximizes hist[j]). This should be your threshold value.

What to submit:

1. the gray-scale image
2. the threshold image
3. the value of the threshold used in 2)

EXTRA CREDIT:

What is the species of bird in the photograph ?

import java.awt.\*;

import java.awt.image.BufferedImage;

import java.io.\*;

import javax.imageio.ImageIO;

import javax.swing.JFrame;

public class GrayScale {

 BufferedImage image;

 int width;

 int height;

 public GrayScale() {

 try {

 File input = new File("digital\_image\_processing.jpg");

 image = ImageIO.read(input);

 width = image.getWidth();

 height = image.getHeight();

 for(int i=0; i<height; i++){

 for(int j=0; j<width; j++){

 Color c = new Color(image.getRGB(j, i));

 int red = (int)(c.getRed() \* 0.299);

 int green = (int)(c.getGreen() \* 0.587);

 int blue = (int)(c.getBlue() \*0.114);

 Color newColor = new Color(red+green+blue,

 red+green+blue,red+green+blue);

 image.setRGB(j,i,newColor.getRGB());

 }

 }

 File ouptut = new File("grayscale.jpg");

 ImageIO.write(image, "jpg", ouptut);

 } catch (Exception e) {}

 }

 static public void main(String args[]) throws Exception

 {

 GrayScale obj = new GrayScale();

 }

}