ASSIGNMENT 1 - Image Processing

Assignment ONE - Image Processing

Due February 10th @ 7pm

This is an individual assignment.....team assignments will start on Assignment Three or four.

HIMP 442 Source code needed is linked as a separate link on today's lecture notes.

Ninety-nine things to do (actually only 6 or 7) :

1. Fix the reset function in the pulldown menu. I fixed it on mine now, but it's a good practice on learning the code and figuring out what's going on.
2. Rotate image 90 degrees, odd shaped images should work.....mine did when I went back and looked at.
3. Turn an image into grayscale and display it, use the "Luminosity" method for grayscale conversion. You can put the grayscale value using the three channel, you don't have to create a new image.
4. Blur the grayscale image, use an average of surrounding pixels to blur the image, you will need a second array so you don't use already blurred pixels in your calculations. First, call the grayscale method from above.
    1. The last thing set the original picture array to your temporary blurred array and call resetPicture()
5. Turn a color image into a grayscale image first and then do a minimum of 3x3 mask to do edge detection. 5x5 will work better and be worth more.
    1. See scoring rubrik below
6. Show a histogram of the colors in a separate window
    1. See notes below
7. Use the values in the histogram to equalize the image:
    1. Use the mapping function to normalize the distribution evenly
    2. https://en.wikipedia.org/wiki/Histogram_equalization

Use HIMP that I linked in today's lecture notes.

Don't forget to resetPicture()

Scoring Rubrik

- Rotate 90 Degrees (7 points)
- Grayscale (5 points)
- Blur the image (7 points)
- Edge Detection, you only get points for one or the other below
    - 3x3 mask (6 points)
    - 5x5 mask (10 points)
- Histograms of all the channels (10 points)
- Equalization (11 point)

HELPFUL HINTS:

Edge Detection

Start with a 3x3 mask (multiply each pixel and surrounding pixels by the multiplier) like this:

-1 -1 -1
-1 8 -1
-1 -1 -1

But if you want a much better line try this:

-1 -1 -1 -1 -1
-1 0   0 0 -1
-1 0 16 0 -1
-1 0   0 0 -1
-1 -1 -1 -1 -1

Histogram

//This is in my histogram function in IMP

//first count all pixel values in R and G and B array

// Then pass those arrays to MyPanel constructor

//Then when button is pushed call drawHistogram in MyPanel.....you write DrawHistogram

//Don't forget to call repaint();

```
  JFrame redFrame = new JFrame("Red");
  redFrame.setSize(305, 600);
  redFrame.setLocation(800, 0);
  JFrame greenFrame = new JFrame("Green");
  greenFrame.setSize(305, 600);
  greenFrame.setLocation(1150, 0);
  JFrame blueFrame = new JFrame("blue");
  blueFrame.setSize(305, 600);
  blueFrame.setLocation(1450, 0);
  redPanel = new MyPanel(red);
  greenPanel = new MyPanel(green);
  bluePanel = new MyPanel(blue);
  redFrame.getContentPane().add(redPanel, BorderLayout.CENTER);
  redFrame.setVisible(true);
  greenFrame.getContentPane().add(greenPanel, BorderLayout.CENTER);
  greenFrame.setVisible(true);
  blueFrame.getContentPane().add(bluePanel, BorderLayout.CENTER);
  blueFrame.setVisible(true);
   start.setEnabled(true);
```

My panel class stuff that inherits from JPanel:

```
//instance fields
BufferedImage grid;
 Graphics2D gc;
///PaintComponent Method
public void paintComponent(Graphics g)
  {
     super.paintComponent(g);
     Graphics2D g2 = (Graphics2D)g;
     if(grid == null){
       int w = this.getWidth();
       int h = this.getHeight();
       grid = (BufferedImage)(this.createImage(w,h));
       gc = grid.createGraphics();

     }
     g2.drawImage(grid, null, 0, 0);

  }
```