MODUL

Praktikum Pengolahan Citra

Prodi S1 Teknik Informatika

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS NEGERI MALANG

The Learning University

MODUL PRAKTIKUM I
IMAGE ENHANCEMENT IN SPATIAL DOMAIN:
GRAY LEVEL TRANSFORMATION

A. Tujuan

1. Mahasiswa mampu menganalisa dan mengimplementasikan transformasi citra RGB ke dalam citra *grayscale*
2. Mahasiswa mampu menganalisa dan mengimplementasikan transformasi citra RGB ke dalam citra *binary*
3. Mahasiswa mampu menganalisa dan mengimplementasikan metode *graylevel transformation: image negative*
4. Mahasiswa mampu menganalisa dan mengimplementasikan metode *graylevel transformation: Log Transformation*
5. Mahasiswa mampu menganalisa dan mengimplementasikan metode *graylevel transformation: Power Log Transformation*

B. Latihan

Latihan 1: Konversi citra RGB ke dalam citra *grayscale*

a. Implementasikan UML class diagram berikut dalam Java

Grayscale
- width : int - height : int - red : int - green : int - blue : int - avg : int - srcPath : String - desPath : String - mode : String - image : BufferedImage
+ Grayscale(srcPath : String, desPath : String, mode : String) + Grayscale(input : File, desPath : String, mode : String) + Grayscale(srcPath : String, desPath : String) + Grayscale(input : File, desPath : String) - doGrayscale()

```
import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.*;

public class Grayscale {

    private int width, height, red, green, blue, avg;
    private String srcPath, desPath, mode;
    private File input, output;
    private BufferedImage image;
```

```
private void doGrayscale() {
    try {
        for (int i = 0; i < height; i++) {
            for (int j = 0; j < width; j++) {
                Color c = new Color(image.getRGB(j, i));
                red = (int) (c.getRed());
                green = (int) (c.getGreen());
                blue = (int) (c.getBlue());
                avg = (red + green + blue) / 3;
                Color newColor = new Color(avg,
                        avg, avg);
                image.setRGB(j, i, newColor.getRGB());
            }
        }
        this.output = new File(desPath);
        ImageIO.write(image, mode, output);
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}
public Grayscale(String srcPath, String desPath, String mode) {
    try {
        input = new File(srcPath);
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.srcPath = srcPath;
        this.desPath = desPath;
        this.mode = mode;
        doGrayscale();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }

}

public Grayscale(File input, String desPath, String mode) {
    try {
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.desPath = desPath;
        this.mode = mode;
        doGrayscale();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

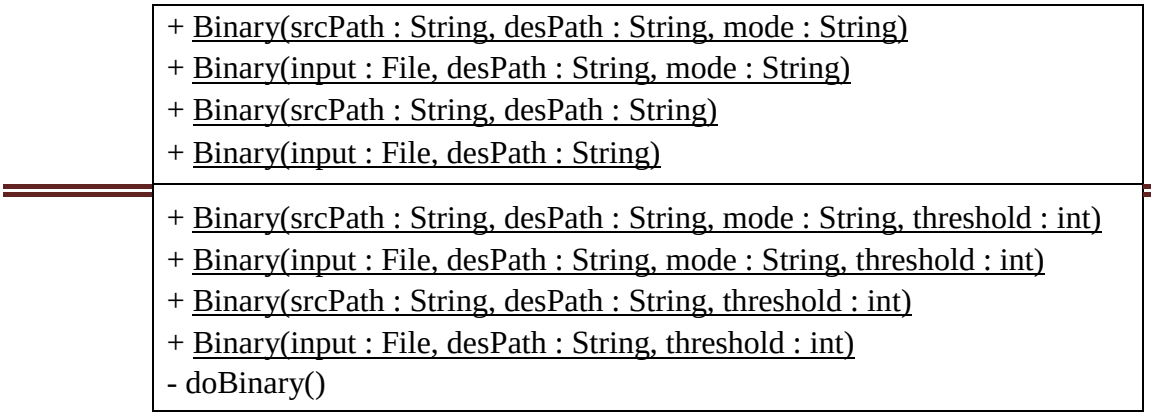
```
public Grayscale(String srcPath, String desPath) {
    try {
        input = new File(srcPath);
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.srcPath = srcPath;
        this.desPath = desPath;
        this.mode = "jpg";
        doGrayscale();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}

public Grayscale(File input, String desPath) {
    try {
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.desPath = desPath;
        this.mode = "jpg";
        doGrayscale();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

b. Buat *class* TesGrayscale.java untuk membuat object dari kelas di atas.

c. Output program di atas adalah:

Latihan 2: Konversi citra RGB / Grayscale ke dalam citra *binary* a. Implementasikan UML class diagram berikut dalam Java

Binary
- width : int - height : int - red : int - green : int - blue : int - avg : int - bin : int - threshold : int - srcPath : String - desPath : String - mode : String - image : BufferedImage


+ Binary(srcPath : String, desPath : String, mode : String)
+ Binary(input : File, desPath : String, mode : String)
+ Binary(srcPath : String, desPath : String)
+ Binary(input : File, desPath : String)
+ Binary(srcPath : String, desPath : String, mode : String, threshold : int)
+ Binary(input : File, desPath : String, mode : String, threshold : int)
+ Binary(srcPath : String, desPath : String, threshold : int)
+ Binary(input : File, desPath : String, threshold : int)
- doBinary()

```
import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.*;

public class Binary {

    int width, height, red, green, blue, avg, bin, threshold = 127;
    String srcPath, desPath, mode;
    File input, output;
    BufferedImage image;
```

```
private void doBinary() {
    try {
        for (int i = 0; i < height; i++) {
            for (int j = 0; j < width; j++) {
                Color c = new Color(image.getRGB(j, i));
                red = (int) (c.getRed());
                green = (int) (c.getGreen());
                blue = (int) (c.getBlue());
                avg = (red + green + blue) / 3;
                if (avg <= threshold) {
                    bin = 0;
                } else {
                    bin = 255;
                }
                Color newColor = new Color(bin,
                        bin, bin);
                image.setRGB(j, i, newColor.getRGB());
            }
        }
        this.output = new File(desPath);
        ImageIO.write(image, mode, output);
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

```
public Binary(String srcPath, String desPath, String mode, int threshold) {
    try {
        input = new File(srcPath);
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.srcPath = srcPath;
        this.desPath = desPath;
        this.mode = mode;
        this.threshold = threshold;
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }

}
```

```
public Binary(String srcPath, String desPath, String mode) {
    try {
        input = new File(srcPath);
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.srcPath = srcPath;
        this.desPath = desPath;
        this.mode = mode;
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}

public Binary(File input, String desPath, String mode, int threshold) {
    try {
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.desPath = desPath;
        this.mode = mode;
        this.threshold = threshold;
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

```
public Binary(File input, String desPath, String mode) {
    try {
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.desPath = desPath;
        this.mode = mode;
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}

public Binary(String srcPath, String desPath, int threshold) {
    try {
        input = new File(srcPath);
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.srcPath = srcPath;
        this.desPath = desPath;
        this.mode = "jpg";
        this.threshold = threshold;
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

```
public Binary(String srcPath, String desPath) {
    try {
        input = new File(srcPath);
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.srcPath = srcPath;
        this.desPath = desPath;
        this.mode = "jpg";
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}

public Binary(File input, String desPath, int threshold) {
    try {
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.desPath = desPath;
        this.mode = "jpg";
        this.threshold = threshold;
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}
public Binary(File input, String desPath) {
    try {
        image = ImageIO.read(input);
        width = image.getWidth();
        height = image.getHeight();
        this.desPath = desPath;
        this.mode = "jpg";
        doBinary();
    } catch (IOException ex) {
        Logger.getLogger(Grayscale.class.getName()).log(Level.SEVERE, null, ex);
    }
}

}
```

b. Buat *class* TesBinary.java untuk membuat object dari kelas di atas. Uji coba citra masukan dengan minimal 5 threshold yang berbeda.

c. Output program di atas adalah:

C. Praktikum

1. Implementasikan metode *image enhancement: image negative* dengan menggunakan masukan citra RGB / *grayscale* dengan menggunakan persamaan berikut:

$$s = L - r$$

Keterangan:

s	: Intensitas piksel citra hasil transformasi
L	: Intensitas maksimal citra *grayscale* (255)
r	: Intensitas piksel citra asal

2. Implementasikan metode *image enhancement: log transformation* dengan menggunakan masukan citra RGB / *grayscale* dengan menggunakan persamaan berikut:

$$s = c\ Log\ (1 + r)$$

Keterangan:	
r	: Intensitas piksel citra asal
c	: Konstanta
s	: Intensitas piksel citra hasil transformasi

Uji coba citra masukan dengan menggunakan variasi nilai c = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.

3. Implementasikan metode *image enhancement: power log transformation* dengan menggunakan masukan citra RGB / *grayscale* dengan menggunakan persamaan berikut:

$$s = c\ r^{\gamma}$$

Keterangan:	
r	: Intensitas piksel citra asal
c	: Konstanta = 1
γ	: Konstanta
s	: Intensitas piksel citra hasil transformasi

Uji coba citra masukan dengan menggunakan variasi nilai γ = 0.04, 0.1, 0.2, 0.4, 0.67, 1, 1.5, 2.5, 5, 10, 25.

D. TUGAS RUMAH

Buat GUI program untuk menggabungkan ketiga metode *gray level transformation* di atas.