CPIT-380 PROJECT REPORT

Instructed by Dr. Saim Rasheed

February 2023

Name	Student ID	Section
Abdulaziz Al-Sharif	2036023	IT1
Rakan Salama	2037276	
Omar Al-Zahrani	2040569	
Fahad Al-Sifri	1943998	
Majed Al-Ghamdi	1942874	

Table of Contents

Main menu 5
0- Files Menu 6
Browse Code 7
Reset Code 7
Save Code 7
1- Color change 8
Change red,green,blue colors using sliders code 9
Clear red code 10
Clear green code 10
Clear blue code 10
Darker Code 10
Brighter Code 11
2- Copy and Crop 11
Crop Code 12
3- Convert RGB to Gray Scale Image 13
Convert RGB to Gray Scale Image code 13
4- 14
Convert Gray scale to Binary Image using thresholding technique code 15
5- Convert and RGB image to HSV colored image 15
Convert and RGB image to HSV colored image code 16
6- Blending Two or more images 17
Blending Two or more images code 18
7 + 8- Left/right/180 rotations: 19
Left Rotation code 19
Right Rotation code 20
180 degree Rotation code 20
9 + 10 + 11- Reflections (Vertical, horizontal, Diagonal) 21
Horizontal Code 23

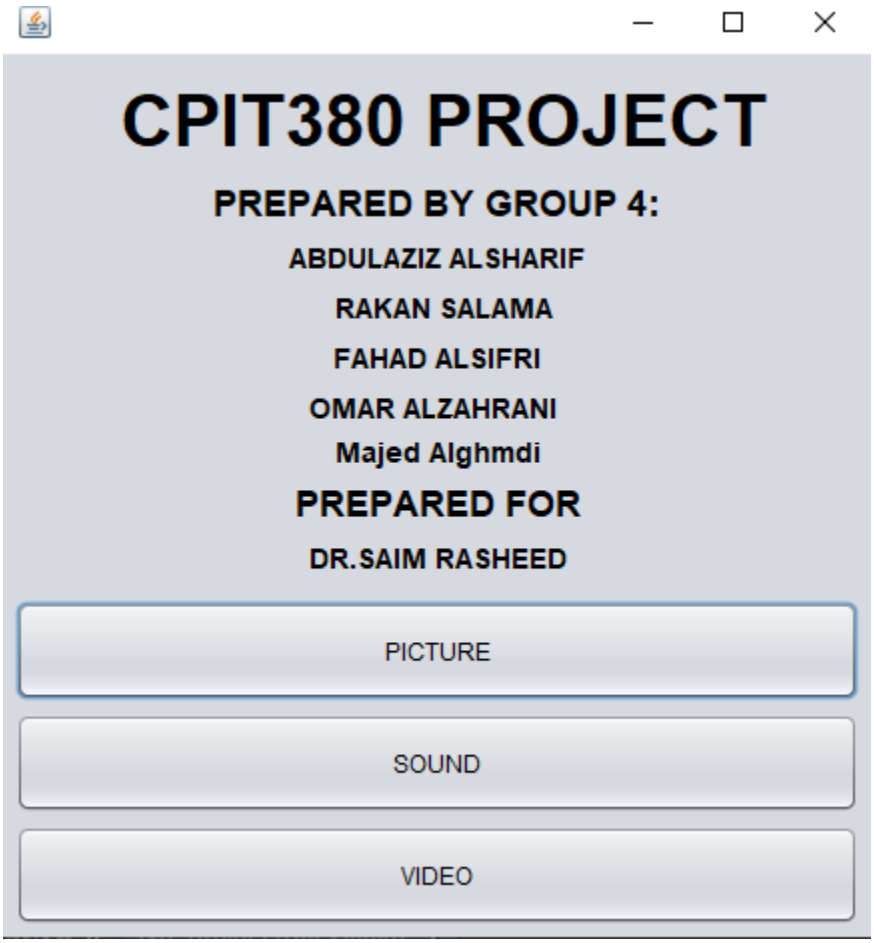
Vertical code 23
Diagonal D1 Code 24
Diagonal D2 Code 24
12 + 13- Scalling Up/Down: 25
Scaling Up/Down Code 25
14- Create Collage 26
Create Collage Code 27
15- Computing histograms 28
Computing histograms for red 29
Computing histograms for Green 29
Computing histograms for Blue 30
Computing histograms for gray scale 30
16- Reconstruction of Image using Histogram(BONUS): 31
Export Histogram Code 31
Import Histogram Code 32
17- Computing Contrast 33
Computing contrast code 34
Adjustment of contrast using a slider code 35
19 + 20 + 21 + 22 + 23- Filters(Box,Gaussian,Laplacian,Min,Max,Median and Weighted median) 36
Box Filter code 37
Min Filter Code 38
Max Filter Code 39
Gaussian Filter Code 40
Laplacian Filter Code 41
Weighted median Filter Code 43
24- Red Eye Reduction using thresholding technique 43
Red Eye Reduction using thresholding technique Code 44
25- Automatic detection of red eyes 45
Automatic detection of red eyes Code 46

26- Background Subtraction 46
Background Subtraction Code 47
27- Edge Detection methods 48
Edge Detection methods Code 49
28- Change Volume Using Slider 50
Change Volume using a slider code 50
29 - Clipping a given sound 51
Clip code 51
30- Splicing at least two sounds together keeping the silent zone 52
Splice code 53
31- Spread & squeeze a sound 54
Spread & squeeze code 55
32- Reversing and Mirroring 55
Reversing code 56
Mirror code 56
33- Blending Two or more sounds 57
Blend code 58
34- Apply simple average, weighted average, max, min and median filters on sounds 59
SoundFilters code 60
36- Create Rectangle Movie method, Show it is moving from top right corner to bottom left, curved trajectory 61
Top right to bottom left code 62
Top Left to bottom Right code 63
37- Create Rectangle Movie method, Show the trajectory of motion followed by a Sine function 63
Sine wave code 64
38- Create Rectangle Movie method Show the trajectory of motion, when a ball is dropped on the floor 64
Dropp ball code 65
39- CreateTickerTapeMovie – A string of text moves back and forth and does not disappear from screen 66

TickerTapeMovie code 67
40- Create a movie, cropping a part of given image that appears randomly on screen 68
Cropping and move in a video code 69
41- Create Sunset movie 69
Create Sunset movie Code 70
42- Back Ground Subtraction Movie 71
Back Ground Subtraction Movie Code 72
43- Edge Detection Movie 72
Edge Detection Movie Code: 73
44- Compare two Images 74
Compare two Images code 74
45- Shot boundary detection 75
Shot boundary detection code 76
47- Detecting a violation of traffic rule 77
Detecting a violation of traffic rule code 77
48- Object Detection 78
Object detection code 78

Note: We have a github format that has all the details like videos and sounds, please check it out.
https://github.com/Nero-69/Cpit380Project

Main menu:



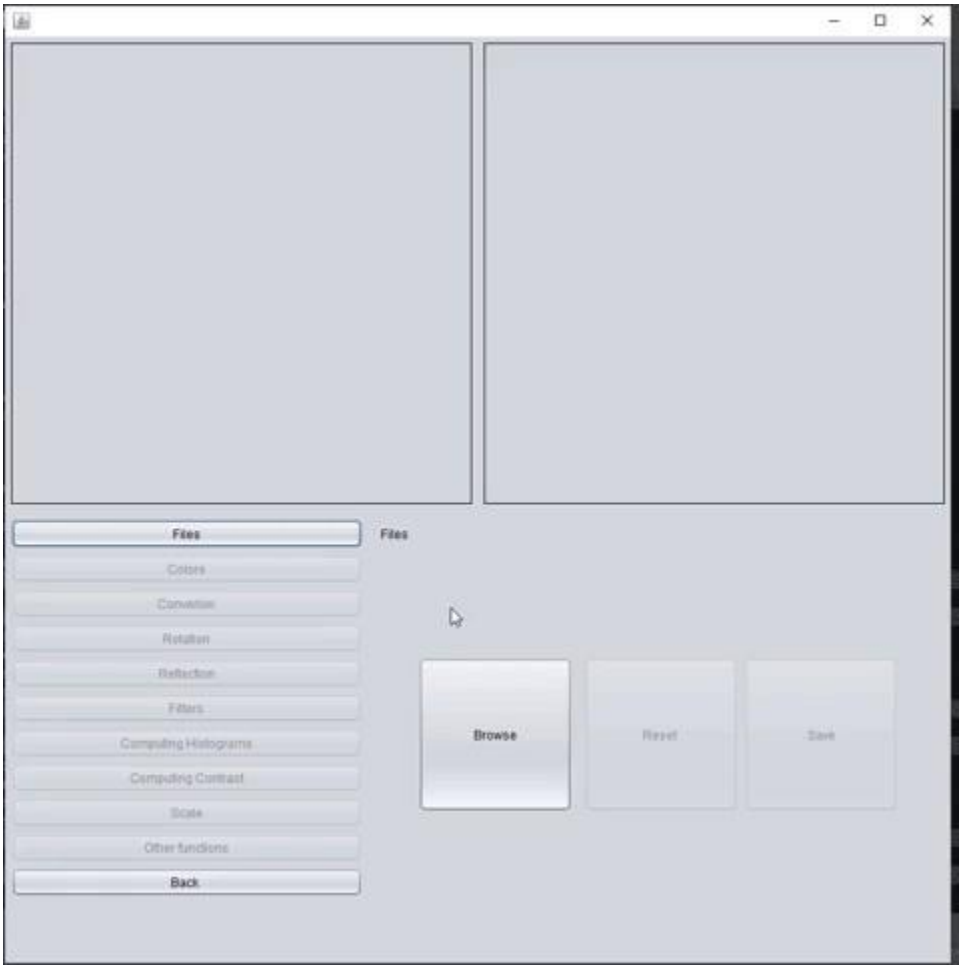
In the main menu, users can choose which section they want toedit by pressing the corresponding button.

Please Watch this video to learn how to upload our project and don't forget to fill the google form ❤️

Youtube videos : https://www.youtube.com/watch?v=fjpwWmMdilE&ab_channel=Nero

Google form: https://docs.google.com/forms/d/e/1FAIpQLSd_4BxOBD0WKhbCigF-rGkjjsyPQ1UEMUC1fCGNlhJKJ8YOaQ/viewform?fbzx=8509294189677587184

0-Files Menu:



User can browse for image to activate the buttons, and they can reset the image to the original photo, at the end they can save the edited image.

Browse Code:

```
private void jButton12ActionPerformed(java.awt.event.ActionEvent evt) {
    /*
    Browse
    this method make the user to browse for picture and enable all the buttons in the gui
    the input will be the picture user chose
    the output picObj and pathName
     */

    try {
        jLabel2.setIcon(null);
        pathName = FileChooser.pickAFile();
        picObj = new Picture(pathName);
        Image img = (picObj.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
        BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
        Graphics2D bGr = bimage.createGraphics();
        bGr.drawImage(img, 0, 0, null);
        bGr.dispose();
        File outputfile = new File("Tmp\\saved.png");
        ImageIO.write(bimage, "png", outputfile);
        pathName = "Tmp\\saved.png";
        picObj = new Picture(pathName);
        icon = new ImageIcon(img);
        jLabel1.setIcon(icon);
        jButton2.setEnabled(true);
        jButton3.setEnabled(true);
        jButton4.setEnabled(true);
        jButton5.setEnabled(true);
        jButton6.setEnabled(true);
        jButton7.setEnabled(true);
        jButton8.setEnabled(true);
        jButton9.setEnabled(true);
        jButton10.setEnabled(true);
        jButton13.setEnabled(true);
        jButton14.setEnabled(true);
    } catch (Exception e) {
        System.out.println("er");
    }
}
```

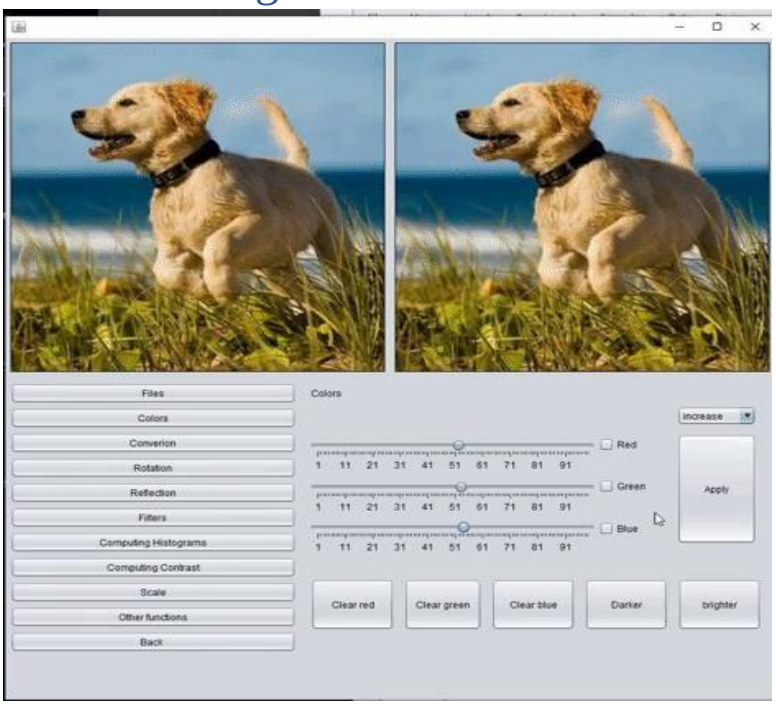
Reset Code:

```
private void jButton13ActionPerformed(java.awt.event.ActionEvent evt) {
    picObj = new Picture(pathName);
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

Save Code:

```
private void jButton14ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Image img = (picObj.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
        BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
        Graphics2D bGr = bimage.createGraphics();
        bGr.drawImage(img, 0, 0, null);
        bGr.dispose();
        File outputfile = new File("saved.png");
        ImageIO.write(bimage, "png", outputfile);
    } catch (IOException ex) {
        Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
    }
```

1-Color change:



- Users can customize the RGB colors of their choice by clicking on the color box, adjusting the slider to the desired amount, and selecting whether to increase or decrease the color value. Once they've adjusted the colors to their liking, they can click the "Apply" button to apply the changes.
- Users can remove red from the picture by clicking on the 'Clear red' button.
- Users can remove green from the picture by clicking on the 'Clear green' button.

- Users can remove blue from the picture by clicking on the 'Clear blue' button.
- Users can adjust the darkness of the image by clicking the "Darken" button.
- Users can adjust the brightness of the picture by clicking on the "Brighter" button.

Change red,green,blue colors using sliders code:

```
private void jButton20ActionPerformed(java.awt.event.ActionEvent evt) {
    double red = -1;
    double green = -1;
    double blue = -1;
    if (jCheckBox1.isSelected()) {
        red = jSlider1.getValue();
        if (jComboBox1.getSelectedIndex() == 0) {
            red = 1 + (red / 100);
        } else if (jComboBox1.getSelectedIndex() == 1) {
            red = 1 - (red / 100);
        }
    }
    if (jCheckBox2.isSelected()) {
        green = jSlider2.getValue();
        if (jComboBox1.getSelectedIndex() == 0) {
            green = 1 + (green / 100);
        } else if (jComboBox1.getSelectedIndex() == 1) {
            green = 1 - (green / 100);
        }
    }
    if (jCheckBox3.isSelected()) {
        blue = jSlider3.getValue();
        if (jComboBox1.getSelectedIndex() == 0) {
            blue = 1 + (blue / 100);
        } else if (jComboBox1.getSelectedIndex() == 1) {
            blue = 1 - (blue / 100);
        }
    }

    Pixel[] pixelArray = picObj.getPixels();
    double value = 0;
    for (Pixel pixelObj : pixelArray) {
        if (red != -1.0) {
            value = pixelObj.getRed();
            pixelObj.setRed((int) (value * red));
        }
        if (green != -1.0) {
            value = pixelObj.getGreen();
            pixelObj.setGreen((int) (value * green));
        }
        if (blue != -1.0) {
            value = pixelObj.getBlue();
            pixelObj.setBlue((int) (value * blue));
        }

    }
```

Clear red code:

```
private void jButton15ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel[] pixelArray = picObj.getPixels();
    for (Pixel pixelObj : pixelArray) {
        pixelObj.setRed(0);
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

Clear green code:

```
private void jButton16ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel[] pixelArray = picObj.getPixels();
    for (Pixel pixelObj : pixelArray) {
        pixelObj.setGreen(0);
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

Clear blue code:

```
private void jButton17ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel[] pixelArray = picObj.getPixels();
    for (Pixel pixelObj : pixelArray) {
        pixelObj.setBlue(0);
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

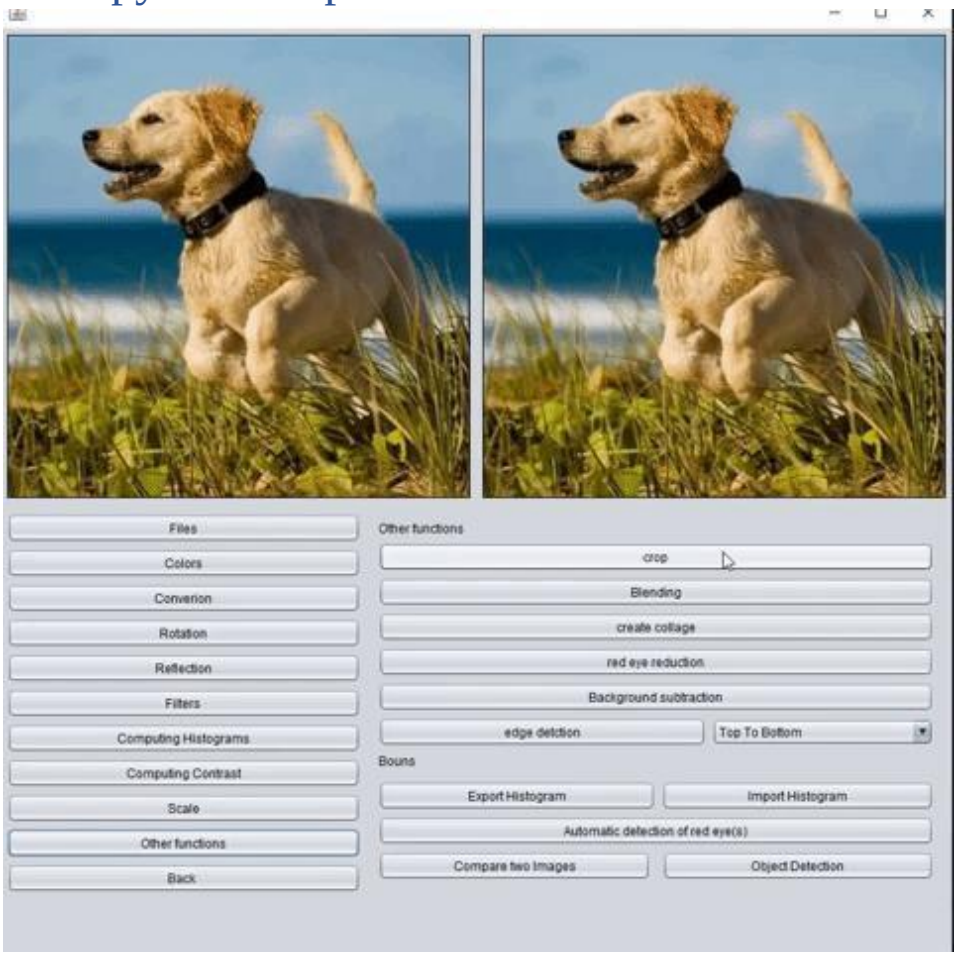
Darker Code:

```
private void jButton18ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel[] pixelArray = picObj.getPixels();
    Color color = null;
    for (Pixel pixelArray1 : pixelArray) {
        color = pixelArray1.getColor();
        color = color.darker();
        pixelArray1.setColor(color);
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

Brighter Code:

```
private void jButton19ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel[] pixelArray = picObj.getPixels();
    Color color = null;
    for (Pixel pixelArray1 : pixelArray) {
        color = pixelArray1.getColor();
        color = color.brighter();
        pixelArray1.setColor(color);
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

2- Copy and Crop

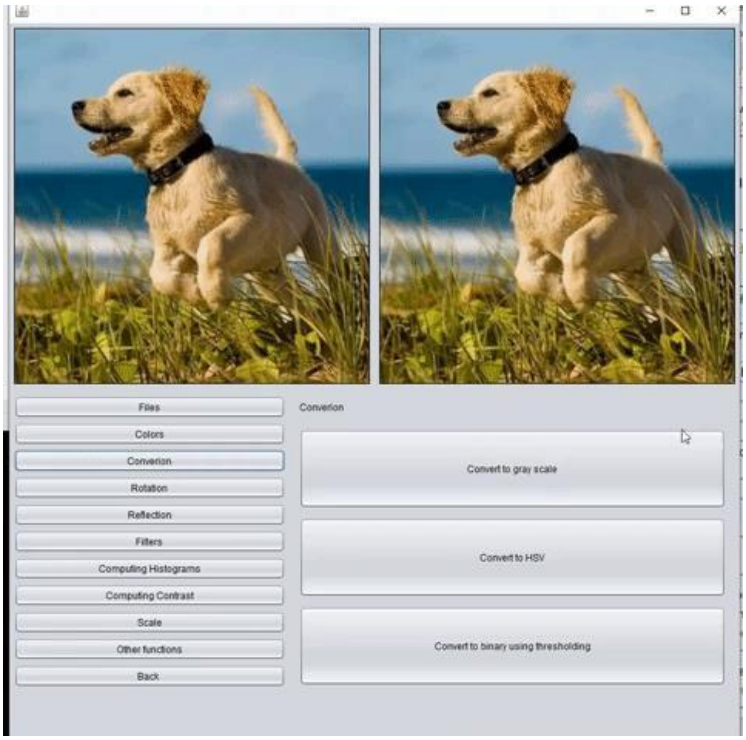


Users can crop any part of the picture by selecting the starting point and the ending point with two clicks. The first click will mark the beginning of the cropping area and the second click will mark the end of the cropping area.

Crop Code:

```
private void jButton46ActionPerformed(java.awt.event.ActionEvent evt) {
    // cropping
    JFrame parent = new JFrame();
    JOptionPane.showMessageDialog(parent, "Please click on two points in the image");
    jLabel1.addMouseListener(new MouseAdapter() {
        int numOfClicks = 0;
        int x1;
        int x2;
        int y1;
        int y2;

        @Override
        public void mouseClicked(MouseEvent e) {
            numOfClicks++;
            if (numOfClicks == 1) {
                x1 = e.getX();
                y1 = e.getY();

            } else if (numOfClicks == 2) {
                x2 = e.getX();
                y2 = e.getY();
                // call the crop method with both cordinates.

                double W = (picObj.getWidth() * 1.00 / jLabel1.getWidth());
                double H = (picObj.getHeight() * 1.00 / jLabel1.getHeight());

                x1 = (int) (W * x1);
                x2 = (int) (W * x2);
                y1 = (int) (H * y1);
                y2 = (int) (H * y2);

                Picture newPic = new Picture(picObj.getWidth(), picObj.getHeight());

                Pixel sourcePixel;
                Pixel targetPixel;

                for (int i = x1; i < x2; i++) {
                    for (int j = y1; j < y2; j++) {
                        sourcePixel = picObj.getPixel(i, j);
                        targetPixel = newPic.getPixel(i, j);
                        targetPixel.setColor(sourcePixel.getColor());
                    }

                }
                picObj = newPic;
                numOfClicks = 0;
                jLabel1.removeMouseListener(this);
                Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(),
                        jLabel2.getHeight(), Image.SCALE_SMOOTH);
                icon = new ImageIcon(img);
                jLabel2.setIcon(icon);
            }
        }
    });


}
```

3- Convert RGB to Gray Scale Image:

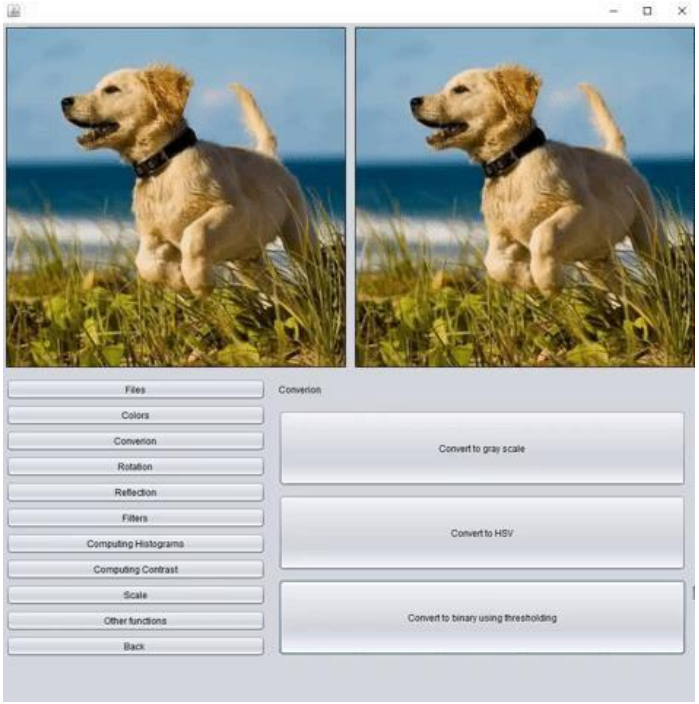


By pressing the "Convert to Gray Scale" button, users can change the colors of the picture to gray.

Convert RGB to Gray Scale Image code:

```
private void jButton21ActionPerformed(java.awt.event.ActionEvent evt) {
    //Convert To Gray Scale
    Pixel[] pixels = picObj.getPixels();
    int avg;
    for (Pixel pixel1 : pixels) {
        avg = (int) ((pixel1.getRed() + pixel1.getGreen() + pixel1.getBlue()) / 3);
        Color grayColor = new Color(avg, avg, avg);
        pixel1.setColor(grayColor);
    }

    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

4- Convert Gray scale to Binary Image using thresholding technique



Users will select a threshold value, which will be used to convert an image to grayscale and then to a binary Image.

Convert Gray scale to Binary Image using thresholding technique code:

```
private void jButton22ActionPerformed(java.awt.event.ActionEvent evt) {
    String threshold = JOptionPane.showInputDialog(null, "Please Enter the size: ");
    Pixel[] pixels = picObj.getPixels();
    int avg;
    for (Pixel pixell : pixels) {
        avg = (int) ((pixell.getRed() + pixell.getGreen() + pixell.getBlue()) / 3);
        Color grayColor = new Color(avg, avg, avg);
        pixell.setColor(grayColor);
    }
    int AVG;
    Picture target = new Picture(picObj.getWidth(), picObj.getHeight());
    for (int x = 0; x < picObj.getWidth(); x++) {
        for (int y = 0; y < picObj.getHeight(); y++) {
            AVG = (int) Math.abs(picObj.getPixel(x, y).getAverage());
            if (AVG < Integer.parseInt(threshold)) {
                target.getPixel(x, y).setColor(Color.BLACK);
            }
        }
    }

    picObj = target;
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

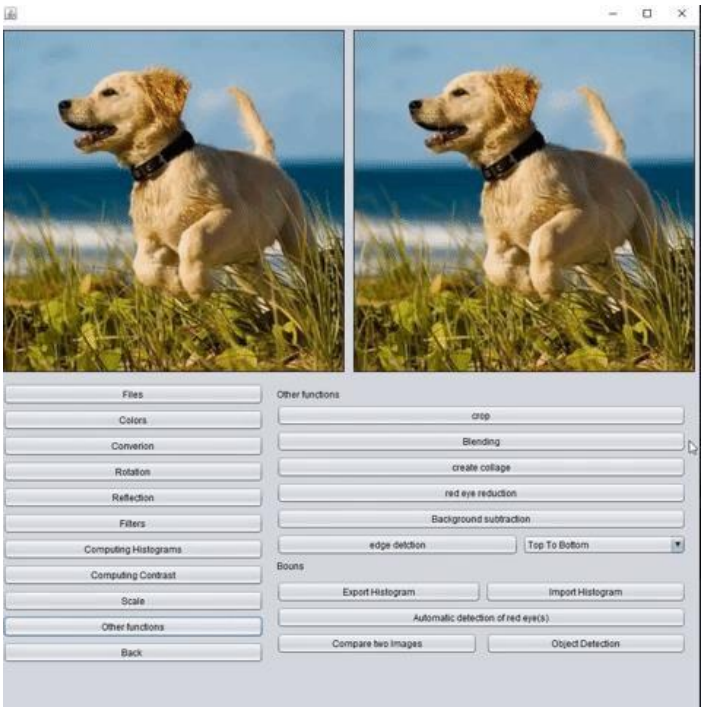
5- Convert and RGB image to HSV colored image:

When users press the button "convert to HSV" the values of the pixels will be converted from RGB into HSV . The output will be the same.

Convert and RGB image to HSV colored image code:

```
private void jButton23ActionPerformed(java.awt.event.ActionEvent evt) {
    //hsv
    Image img = (picObj.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
    BufferedImage rgbImage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_RGB);
    BufferedImage hsvImage = new BufferedImage(rgbImage.getWidth(), rgbImage.getHeight(), BufferedImage.TYPE_INT_RGB);
    for (int x = 0; x < rgbImage.getWidth(); x++) {
        for (int y = 0; y < rgbImage.getHeight(); y++) {
            Color rgb = new Color(rgbImage.getRGB(x, y));
            int r = rgb.getRed();
            int g = rgb.getGreen();
            int b = rgb.getBlue();
            int max = Math.max(r, Math.max(g, b));
            int min = Math.min(r, Math.min(g, b));
            float h = 0, s = 0, v = max / 255f;
            int delta = max - min;
            if (max != 0) {
                s = delta / (float) max;
            }
            if (delta != 0) {
                if (r == max) {
                    h = (g - b) / (float) delta;
                } else if (g == max) {
                    h = 2 + (b - r) / (float) delta;
                } else {
                    h = 4 + (r - g) / (float) delta;
                }
                h /= 6;
                if (h < 0) {
                    h++;
                }
            }
            int hsv = ((int) (h * 255)) | ((int) (s * 255)) | (int) (v * 255);
            hsvImage.setRGB(x, y, hsv);
        }
    }
    Graphics2D bGr = hsvImage.createGraphics();
    bGr.drawImage(img, 0, 0, null);
    bGr.dispose();
    File outputfile = new File("Tmp\\HSV.png");
    try {
        ImageIO.write(hsvImage, "png", outputfile);
    } catch (IOException ex) {
        Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
    }
    Picture s = new Picture("Tmp\\HSV.png");
    img = (s.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

6- Blending Two or more images:

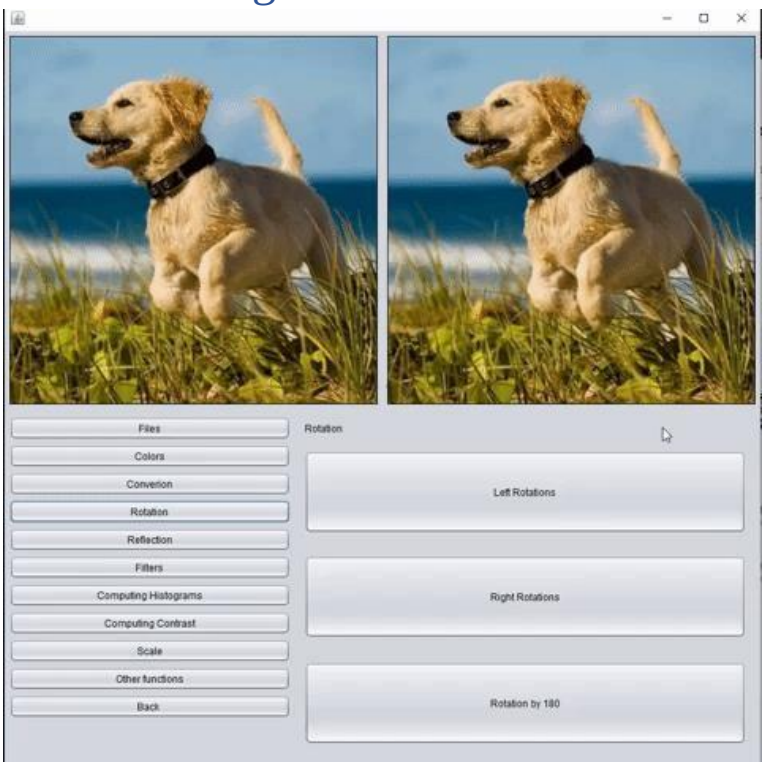


Users can blend 2 or more picture into one picture by pressing the button "Blending".

Blending Two or more images code:

```
private void jButton47ActionPerformed(java.awt.event.ActionEvent evt) {
    // blending
    String fileName2 = FileChooser.pickAFile();
    Picture picture1 = new Picture(fileName2);
    Image img = (picture1.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
    BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
    Graphics2D bGr = bimage.createGraphics();
    bGr.drawImage(img, 0, 0, null);
    bGr.dispose();
    File outputfile = new File("Tmp\\blend2.png");
    try {
        ImageIO.write(bimage, "png", outputfile);
    } catch (IOException ex) {
        Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
    }
    Picture picture2 = new Picture("Tmp\\blend2.png");
    Picture TargetPicture = new Picture(picObj.getWidth() + picture2.getWidth(), picObj.getHeight() + picture2.getHeight());

    Pixel pic1;
    Pixel pic2;
    Pixel targetPixel;
    int sourceX = 0;
    int targetX = 0;
    for (; sourceX < 150; sourceX++, targetX++) {
        for (int sourceY = 0, targetY = 0; sourceY < picObj.getHeight(); sourceY++, targetY++) {
            pic1 = picObj.getPixel(sourceX, sourceY);
            targetPixel = TargetPicture.getPixel(targetX, targetY);
            targetPixel.setColor(pic1.getColor());

        }
    }
    for (; sourceX < picObj.getWidth(); sourceX++, targetX++) {
        for (int sourceY = 0, targetY = 0; sourceY < picObj.getHeight(); sourceY++, targetY++) {
            pic1 = picObj.getPixel(sourceX, sourceY);
            pic2 = picture2.getPixel(sourceX - 150, sourceY);
            targetPixel = TargetPicture.getPixel(targetX, targetY);
            Color b = new Color((int) (pic1.getRed() * 0.5 + pic2.getRed() * 0.5), (int) (pic1.getGreen() * 0.5 + pic2.getGreen() * 0.5),
                    (int) (pic1.getBlue() * 0.5 + pic2.getBlue() * 0.5));
            targetPixel.setColor(b);
        }
    }
    int tmpx = 0;
    int tmpy = 0;
    sourceX = sourceX - 150;
    for (; sourceX < picture2.getWidth(); sourceX++, targetX++) {
        for (int sourceY = 0, targetY = 0; sourceY < picture2.getHeight(); sourceY++, targetY++) {
            pic2 = picture2.getPixel(sourceX, sourceY);
            targetPixel = TargetPicture.getPixel(targetX, targetY);
            targetPixel.setColor(pic2.getColor());
            tmpy = targetY;
        }
        tmpx = targetX;
    }

    Pixel tmpc = null;
    Pixel targetPixel1 = null;
    Picture TargetPicture1 = new Picture(tmpx, tmpy);
    for (int i = 0; i < tmpx; i++) {
        for (int j = 0; j < tmpy; j++) {
            tmpc = TargetPicture.getPixel(i, j);
            targetPixel1 = TargetPicture1.getPixel(i, j);
            targetPixel1.setColor(tmpc.getColor());
        }
    }
    picObj = TargetPicture1;
    Image img1 = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img1);
    jLabel2.setIcon(icon);
}
```

7 + 8- Left/right/180 rotations:



Pressing "Left Rotations" allows users to rotate the selected picture to the left, while "Right Rotations" rotates it to the right. For a 180-degree rotation, press the button "Rotation by 180".

Left Rotation code:

```
private void jButton24ActionPerformed(java.awt.event.ActionEvent evt) {
    //Left Rotation
    Picture sourcePicture = picObj;
    Picture TargetPicture = new Picture(sourcePicture.getHeight(), sourcePicture.getWidth());
    Pixel sourcePixel;
    Pixel targetPixel;
    int targetX, targetY = 0;
    for (int sourceX = 0; sourceX < sourcePicture.getWidth(); sourceX++) {
        for (int sourceY = 0; sourceY < sourcePicture.getHeight(); sourceY++) {
            sourcePixel = sourcePicture.getPixel(sourceX, sourceY);
            targetX = sourceY;
            targetY = sourcePicture.getWidth() - 1 - sourceX;
            targetPixel = TargetPicture.getPixel(targetX, targetY);
            targetPixel.setColor(sourcePixel.getColor());
        }
    }
    Image img = (TargetPicture.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
    picObj = TargetPicture;
}
```

Right Rotation code:

```
private void jButton26ActionPerformed(java.awt.event.ActionEvent evt) {
    //Right Rotation
    Picture sourcePicture = picObj;
    Picture TargetPicture = new Picture(sourcePicture.getHeight(), sourcePicture.getWidth());
    Pixel sourcePixel;
    Pixel targetPixel;
    int targetX, targetY = 0;
    for (int sourceX = 0; sourceX < sourcePicture.getWidth(); sourceX++) {
        for (int sourceY = 0; sourceY < sourcePicture.getHeight(); sourceY++) {
            sourcePixel = sourcePicture.getPixel(sourceX, sourceY);
            targetX = sourcePicture.getHeight() - 1 - sourceY;
            targetY = sourceX;
            targetPixel = TargetPicture.getPixel(targetX, targetY);
            targetPixel.setColor(sourcePixel.getColor());
        }
    }
    Image img = (TargetPicture.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
    picObj = TargetPicture;
}
```

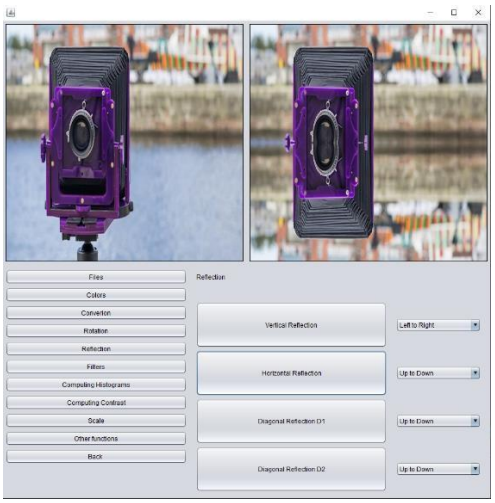
180 degree Rotation code:

```
private void jButton25ActionPerformed(java.awt.event.ActionEvent evt) {
    // Rotation 180
    Picture sourcePicture = picObj;
    Picture TargetPicture = new Picture(sourcePicture.getWidth(), sourcePicture.getHeight());
    Pixel sourcePixel;
    Pixel targetPixel;
    int targetX, targetY = 0;
    for (int sourceX = 0; sourceX < sourcePicture.getWidth(); sourceX++) {
        for (int sourceY = 0; sourceY < sourcePicture.getHeight(); sourceY++) {
            sourcePixel = sourcePicture.getPixel(sourceX, sourceY);
            targetX = sourcePicture.getWidth() - 1 - sourceX;
            targetY = sourcePicture.getHeight() - 1 - sourceY;
            targetPixel = TargetPicture.getPixel(targetX, targetY);
            targetPixel.setColor(sourcePixel.getColor());
        }
    }
    Image img = (TargetPicture.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
    picObj = TargetPicture;
}
```

9 + 10 + 11- Reflections (Vertical, horizontal, Diagonal):

Vertical LR:

Horizontal UD:



Diagonal D1 UD:

Diagonal D2 UD:

Horizontal Code:

```
private void jButton42ActionPerformed(java.awt.event.ActionEvent evt) {
    //diagnoul dl
    if (jComboBox8.getSelectedIndex() == 0) {
        Pixel leftPixel = null;
        Pixel rightPixel = null;
        for (int y = 1; y < jLabel2.getHeight(); y++) {
            for (int x = 0; x < y; x++) {
                rightPixel = picObj.getPixel(x, y);
                leftPixel = picObj.getPixel(y, x);
                rightPixel.setColor(leftPixel.getColor());
            }
        }
    } else if (jComboBox8.getSelectedIndex() == 1) {
        Pixel leftPixel = null;
        Pixel rightPixel = null;
        for (int y = 1; y < jLabel2.getHeight(); y++) {
            for (int x = 0; x < y; x++) {
                leftPixel = picObj.getPixel(x, y);
                rightPixel = picObj.getPixel(y, x);
                rightPixel.setColor(leftPixel.getColor());
            }
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);

}
```

Vertical code:

```
private void jButton41ActionPerformed(java.awt.event.ActionEvent evt) {
    //Vertical reflection
    int width = picObj.getWidth();
    int mirrorPoint = width / 2;
    Pixel leftPixel = null;
    Pixel rightPixel = null;
    //left to right
    if (jComboBox6.getSelectedIndex() == 0) {
        for (int y = 0; y < picObj.getHeight(); y++) {
            for (int x = 0; x < mirrorPoint; x++) {
                leftPixel = picObj.getPixel(x, y);
                rightPixel = picObj.getPixel(width - 1 - x, y);
                rightPixel.setColor(leftPixel.getColor());
            }
        }
        // right to left
    } else if (jComboBox6.getSelectedIndex() == 1) {
        for (int y = 0; y < picObj.getHeight(); y++) {
            for (int x = 0; x < mirrorPoint; x++) {
                leftPixel = picObj.getPixel(x, y);
                rightPixel = picObj.getPixel(width - 1 - x, y);
                leftPixel.setColor(rightPixel.getColor());
            }
        }
    }

    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);

}
```

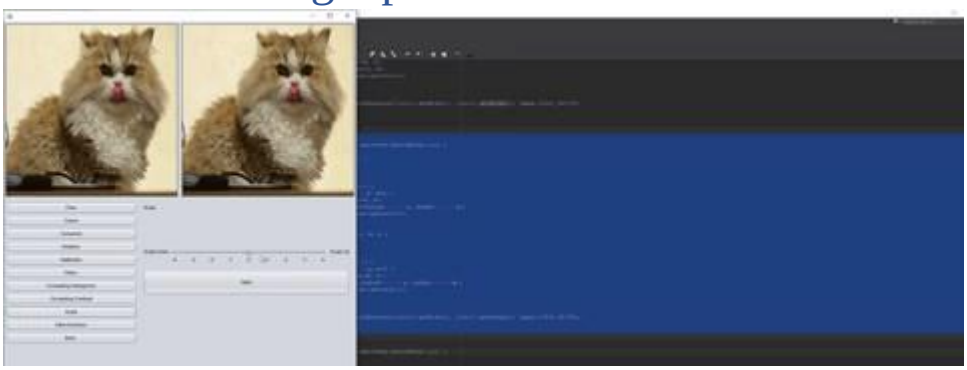
Diagonal D1 Code:

```
private void jButton43ActionPerformed(java.awt.event.ActionEvent evt) {
    // Horizontal Reflection
    int height = picObj.getHeight();
    int mirrorPoint = height / 2;

    Pixel topPixel = null;
    Pixel bottomPixel = null;
    //Up to down
    if (jComboBox7.getSelectedIndex() == 0) {
        for (int x = 0; x < picObj.getWidth(); x++) {
            for (int y = 0; y < mirrorPoint; y++) { // becareful you must start with x loop then y
                topPixel = picObj.getPixel(x, y);
                bottomPixel = picObj.getPixel(x, height - 1 - y);
                bottomPixel.setColor(topPixel.getColor());

            }
        }
        //Down to up
    } else if (jComboBox7.getSelectedIndex() == 1) {
        for (int x = 0; x < picObj.getWidth(); x++) {
            for (int y = 0; y < mirrorPoint; y++) { // becareful you must start with x loop then y
                topPixel = picObj.getPixel(x, y);
                bottomPixel = picObj.getPixel(x, height - 1 - y);
                topPixel.setColor(bottomPixel.getColor());
            }
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

Diagonal D2 Code:

```
private void jButton44ActionPerformed(java.awt.event.ActionEvent evt) {
    //diagnoul d2
    if (jComboBox9.getSelectedIndex() == 0) {
        Pixel leftPixel = null;
        Pixel rightPixel = null;
        int height = picObj.getHeight();
        for (int y = height - 2; y >= 0; y--) {
            for (int x = 0; x < height - 1 - y; x++) {
                leftPixel = picObj.getPixel(x, y);
                rightPixel = picObj.getPixel(height - 1 - y, height - 1 - x);
                rightPixel.setColor(leftPixel.getColor());
            }
        }
    } else if (jComboBox9.getSelectedIndex() == 1) {
        Pixel leftPixel = null;
        Pixel rightPixel = null;
        int height = picObj.getHeight();
        for (int y = height - 2; y >= 0; y--) {
            for (int x = 0; x < height - 1 - y; x++) {
                rightPixel = picObj.getPixel(x, y);
                leftPixel = picObj.getPixel(height - 1 - y, height - 1 - x);
                rightPixel.setColor(leftPixel.getColor());
            }
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

12 + 13- Scalling Up/Down:

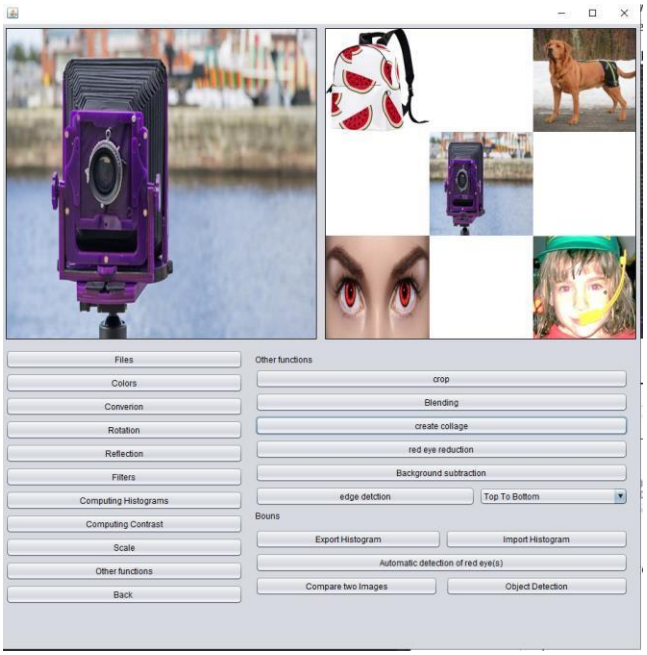


Users can adjust the size of the picture by using the slider. Moving it to the left will cause the picture to scale down, and moving it to the right will cause it to scale up. To apply the changes, press the "apply" button.

Scaling Up/Down Code:

```java
private void jButton39ActionPerformed(java.awt.event.ActionEvent evt) {
    //Scalling
    int value = jSliderS.getValue();
    int scaleValue = 0;
    if (value == -4) {
        scaleValue = 5;
    } else if (value == -3) {
        scaleValue = 4;
    } else if (value == -2) {
        scaleValue = 3;
    } else if (value == -1) {
        scaleValue = 2;
    } else if (value == 1) {
        scaleValue = 2;
    } else if (value == 2) {
        scaleValue = 3;
    } else if (value == 3) {
        scaleValue = 4;
    } else if (value == 4) {
        scaleValue = 5;
    } else {
        scaleValue = 1;
    }
    if (value < 0) {
        Picture targetPicture = new Picture((picObj.getWidth() / scaleValue + 1), (picObj.getHeight() / scaleValue + 1));
        Pixel sourcePixel = null;
        Pixel targetPixel = null;
        for (int sourceX = 0, targetX = 0; sourceX < picObj.getWidth(); sourceX += scaleValue, targetX++) {
            for (int sourceY = 0, targetY = 0; sourceY < picObj.getHeight(); sourceY += scaleValue, targetY++) {
                sourcePixel = picObj.getPixel(sourceX, sourceY);
                targetPixel = targetPicture.getPixel(targetX, targetY);
                targetPixel.setColor(sourcePixel.getColor());
            }
        }
        targetPicture.show();
        picObj = targetPicture;
    } else {

        Picture targetPicture = new Picture((picObj.getWidth() * scaleValue), (picObj.getHeight() * scaleValue));
        Pixel sourcePixel;
        Pixel targetPixel;
        int targetX = 0;
        int targetY = 0;
        for (int sourceX = 0; sourceX < picObj.getWidth(); sourceX++) {
            for (int sourceY = 0; sourceY < picObj.getHeight(); sourceY++) {
                sourcePixel = picObj.getPixel(sourceX, sourceY);
                for (int indexY = 0; indexY < scaleValue; indexY++) {
                    for (int indexX = 0; indexX < scaleValue; indexX++) {
                        targetX = (sourceX * scaleValue + indexX);
                        targetY = (sourceY * scaleValue + indexY);
                        targetPixel = targetPicture.getPixel(targetX, targetY);
                        targetPixel.setColor(sourcePixel.getColor());
                    }
                }
            }
        }
        targetPicture.show();
        picObj = targetPicture;
    }

}
```

14- Create Collage:



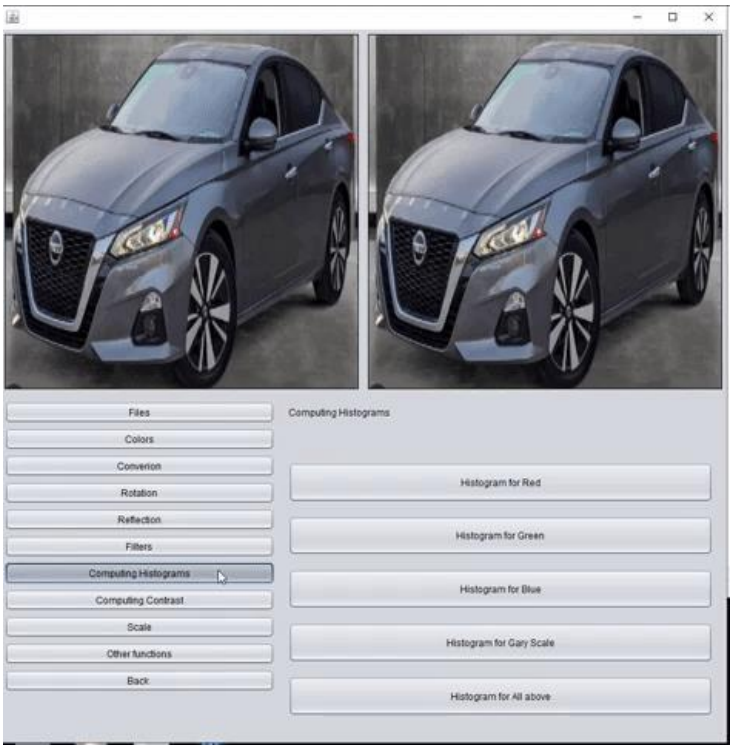
Users can create a collage by placing four different pictures into the corners, with the previously selected image in the center.

Create Collage Code:

```java
private void jButton48ActionPerformed(java.awt.event.ActionEvent evt) {

    Picture sourcePicture = picObj;
    // ask for 4 picture
    Picture second = null;
    Picture third = null;
    Picture forth = null;
    Picture fifth = null;
    JFileChooser FileChooser = new JFileChooser("");
    int conf = JOptionPane.showConfirmDialog(null, "Choose the Left Top image", "Choose image", JOptionPane.OK_CANCEL_OPTION);
    if (conf == 0) {
        int val = FileChooser.showOpenDialog(null);
        if (val == JFileChooser.APPROVE_OPTION) {
            second = new Picture(FileChooser.getSelectedFile().getAbsolutePath());
            Image img = (second.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
            BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
            Graphics2D bGr = bimage.createGraphics();
            bGr.drawImage(img, 0, 0, null);
            bGr.dispose();
            File outputfile = new File("Tmp\\2.png");

            try {
                ImageIO.write(bimage, "png", outputfile);
            } catch (IOException ex) {
                Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
            }
            second = new Picture("Tmp\\2.png");
        }
    }
    conf = JOptionPane.showConfirmDialog(null, "Choose the Right Top image", "Choose image", JOptionPane.OK_CANCEL_OPTION);
    if (conf == 0) {
        int val = FileChooser.showOpenDialog(null);
        if (val == JFileChooser.APPROVE_OPTION) {
            third = new Picture(FileChooser.getSelectedFile().getAbsolutePath());
            Image img = (third.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
            BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
            Graphics2D bGr = bimage.createGraphics();
            bGr.drawImage(img, 0, 0, null);
            bGr.dispose();
            File outputfile = new File("Tmp\\3.png");
            try {
                ImageIO.write(bimage, "png", outputfile);
            } catch (IOException ex) {
                Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
            }
            third = new Picture("Tmp\\3.png");
        }
    }
    conf = JOptionPane.showConfirmDialog(null, "Choose the Left Down image", "Choose image", JOptionPane.OK_CANCEL_OPTION);
    if (conf == 0) {
        int val = FileChooser.showOpenDialog(null);
        if (val == JFileChooser.APPROVE_OPTION) {
            forth = new Picture(FileChooser.getSelectedFile().getAbsolutePath());
            Image img = (forth.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
            BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
            Graphics2D bGr = bimage.createGraphics();
            bGr.drawImage(img, 0, 0, null);
            bGr.dispose();
            File outputfile = new File("Tmp\\4.png");
            try {
                ImageIO.write(bimage, "png", outputfile);
            } catch (IOException ex) {
                Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
            }
            forth = new Picture("Tmp\\4.png");
        }
    }
```

```java
conf = JOptionPane.showConfirmDialog(null, "Choose the Right Down image", "Choose image", JOptionPane.OK_CANCEL_OPTION);
if (conf == 0) {
    int val = FileChooser.showOpenDialog(null);
    if (val == JFileChooser.APPROVE_OPTION) {
        fifth = new Picture(FileChooser.getSelectedFile().getAbsolutePath());
        Image img = (fifth.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
        BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
        Graphics2D bGr = bimage.createGraphics();
        bGr.drawImage(img, 0, 0, null);
        bGr.dispose();
        File outputfile = new File("Tmp\\5.png");
        try {
            ImageIO.write(bimage, "png", outputfile);
        } catch (IOException ex) {
            Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
        }
        fifth = new Picture("Tmp\\5.png");
    }
}

int width = picObj.getWidth() + second.getWidth() + third.getWidth();
int height = picObj.getHeight() + second.getHeight() + forth.getHeight();
picObj = new Picture(width, height);

Pixel sourcePixel;
Pixel targetPixel;

// top left
for (int i = 0; i < second.getWidth(); i++) {
    for (int j = 0; j < second.getHeight(); j++) {
        sourcePixel = second.getPixel(i, j);
        targetPixel = picObj.getPixel(i, j);
        targetPixel.setColor(sourcePixel.getColor());
    }
}
// top right
for (int j = 0; j < third.getWidth(); j++) {
    for (int k = 0; k < third.getHeight(); k++) {
        sourcePixel = third.getPixel(j, k);
        targetPixel = picObj.getPixel((sourcePicture.getWidth() + j + second.getWidth()), (k));
        targetPixel.setColor(sourcePixel.getColor());
    }
}
// original picture in the middle
for (int j = 0; j < sourcePicture.getWidth(); j++) {
    for (int k = 0; k < sourcePicture.getHeight(); k++) {
        sourcePixel = sourcePicture.getPixel(j, k);
        targetPixel = picObj.getPixel((j + second.getWidth()), (k + second.getHeight()));
        targetPixel.setColor(sourcePixel.getColor());
    }
}
// bottom left
for (int j = 0; j < forth.getWidth(); j++) {
    for (int k = 0; k < forth.getHeight(); k++) {
        sourcePixel = forth.getPixel(j, k);
        targetPixel = picObj.getPixel((j), (k + second.getHeight() + sourcePicture.getHeight()));
        targetPixel.setColor(sourcePixel.getColor());
    }
}
// bottom right
for (int j = 0; j < fifth.getWidth(); j++) {
    for (int k = 0; k < fifth.getHeight(); k++) {
        sourcePixel = fifth.getPixel(j, k);
        targetPixel = picObj.getPixel((j + second.getWidth() + sourcePicture.getWidth()),
                (k + third.getHeight() + sourcePicture.getHeight()));
        targetPixel.setColor(sourcePixel.getColor());
    }
}
Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
icon = new ImageIcon(img);
jLabel2.setIcon(icon);
```

15- Computing histograms:



Computing histograms For all RGB color and the Gray scale, every color in separate button GUI, and it will represent it in GUI graph of the choosing histogram color.

Computing histograms for red :

```
private void jButton27ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel_LL[][] Histograms = new Pixel_LL[1][256];

    int maxR = 0;

    for (int i = 0; i < 256; i++) {
        Histograms[0][i] = new Pixel_LL();

    }
    for (int i = 0; i < picObj.getWidth(); i++) {
        for (int j = 0; j < picObj.getHeight(); j++) {
            int intensityR = picObj.getPixel(i, j).getRed();

            Histograms[0][intensityR].addPixel(new PixelLinkedList_node(i, j));

            if (Histograms[0][intensityR].getTotal() > maxR) {
                maxR = Histograms[0][intensityR].getTotal();

            }

        }
    }
```

Computing histograms for Green :

```
private void jButton29ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel_LL[][] Histograms = new Pixel_LL[1][256]; // [0] red, [1] green [2] blue

    int maxG = 0;
    int maxG_index = 0;

    for (int i = 0; i < 256; i++) { // Inisilazing all the arrays
        Histograms[0][i] = new Pixel_LL();

    }
    for (int i = 0; i < picObj.getWidth(); i++) {
        for (int j = 0; j < picObj.getHeight(); j++) {
            int intensityG = picObj.getPixel(i, j).getGreen();

            Histograms[0][intensityG].addPixel(new PixelLinkedList_node(i, j));

            if (Histograms[0][intensityG].getTotal() > maxG) {
                maxG = Histograms[0][intensityG].getTotal();
                maxG_index = intensityG;
            }

        }
    }

    int maxHeight = 0;
    for (int i = 0; i < Histograms[0].length; i++) {
        if (Histograms[0][i].getTotal() > maxHeight) {
            maxHeight = Histograms[0][i].getTotal();
        }
    }
    Picture histogram = new Picture(256, 256, Color.white);
    Color c = Color.GREEN;
    for (int i = 0; i < 256; i++) {
        int max = (int) (Histograms[0][i].getTotal() * 256 / maxHeight);
        for (int j = 255; j >= (256 - max); j--) {
            histogram.getPixel(i, j).setColor(c);
        }
    }
    histogram.scaleUp(2).show();
}
```

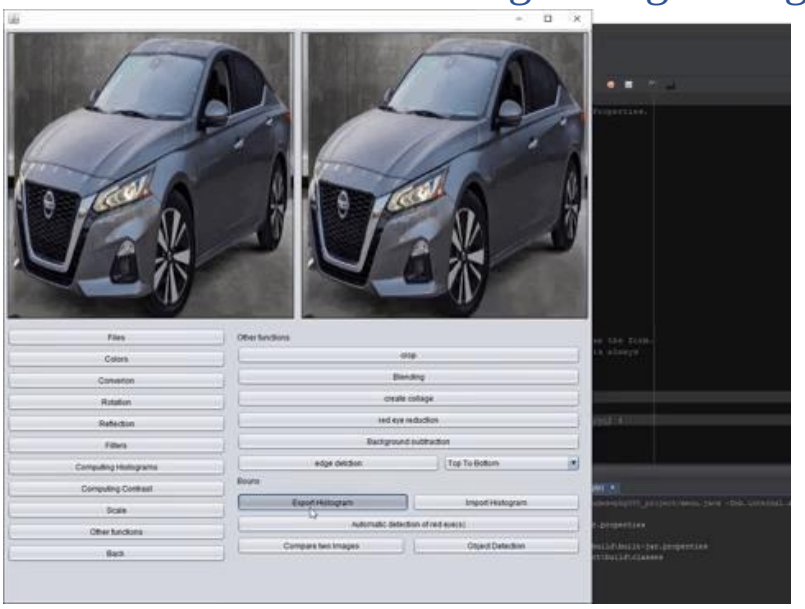
Computing histograms for Blue :

```
private void jButton28ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel_LL[][] Histograms = new Pixel_LL[1][256];

    int maxB = 0;

    for (int i = 0; i < 256; i++) { // Inisilazing all the arrays
        Histograms[0][i] = new Pixel_LL();

    }
    for (int i = 0; i < picObj.getWidth(); i++) {
        for (int j = 0; j < picObj.getHeight(); j++) {
            int intensityB = picObj.getPixel(i, j).getBlue();

            Histograms[0][intensityB].addPixel(new PixelLinkedList_node(i, j));

            if (Histograms[0][intensityB].getTotal() > maxB) {
                maxB = Histograms[0][intensityB].getTotal();

            }

        }
    }

    int maxHeight = 0;
    for (int i = 0; i < Histograms[0].length; i++) {
        if (Histograms[0][i].getTotal() > maxHeight) {
            maxHeight = Histograms[0][i].getTotal();
        }
    }
    Picture histogram = new Picture(256, 256, Color.white);
    Color c = Color.BLUE;
    for (int i = 0; i < 256; i++) {
        int max = (int) (Histograms[0][i].getTotal() * 256 / maxHeight);
        for (int j = 255; j >= (256 - max); j--) {
            histogram.getPixel(i, j).setColor(c);
        }
    }
    histogram.scaleUp(2).show();
}
```

Computing histograms for gray scale:

```
private void jButton30ActionPerformed(java.awt.event.ActionEvent evt) {
    Picture temp = picObj;
    Pixel[] pixels = picObj.getPixels();
    int avg;
    for (Pixel pixell : pixels) {
        avg = (int) ((pixell.getRed() + pixell.getGreen() + pixell.getBlue()) / 3);
        Color grayColor = new Color(avg, avg, avg);
        pixell.setColor(grayColor);
    }

    Pixel_LL[][] Histograms = new Pixel_LL[1][256];

    int maxGS = 0;

    for (int i = 0; i < 256; i++) {
        Histograms[0][i] = new Pixel_LL();

    }
    for (int i = 0; i < picObj.getWidth(); i++) {
        for (int j = 0; j < picObj.getHeight(); j++) {
            int intensityGS = picObj.getPixel(i, j).getBlue();

            Histograms[0][intensityGS].addPixel(new PixelLinkedList_node(i, j));

            if (Histograms[0][intensityGS].getTotal() > maxGS) {
                maxGS = Histograms[0][intensityGS].getTotal();

            }

        }
    }

    int maxHeight = 0;
    for (int i = 0; i < Histograms[0].length; i++) {
        if (Histograms[0][i].getTotal() > maxHeight) {
            maxHeight = Histograms[0][i].getTotal();
        }
    }
    Picture histogram = new Picture(256, 256, Color.white);
    Color c = Color.GRAY;
    for (int i = 0; i < 256; i++) {
        int max = (int) (Histograms[0][i].getTotal() * 256 / maxHeight);
        for (int j = 255; j >= (256 - max); j--) {
            histogram.getPixel(i, j).setColor(c);
        }
    }
    histogram.scaleUp(2).show();
    picObj = new Picture(pathName);
}
```

16- Reconstruction of Image using Histogram(BONUS):



Export Histogram TXT that contain saved location information of a pixel and the intensity of the pixel using linked list, first the user browse for photo they want to save it than when they need it press import to reconstruct the photo.

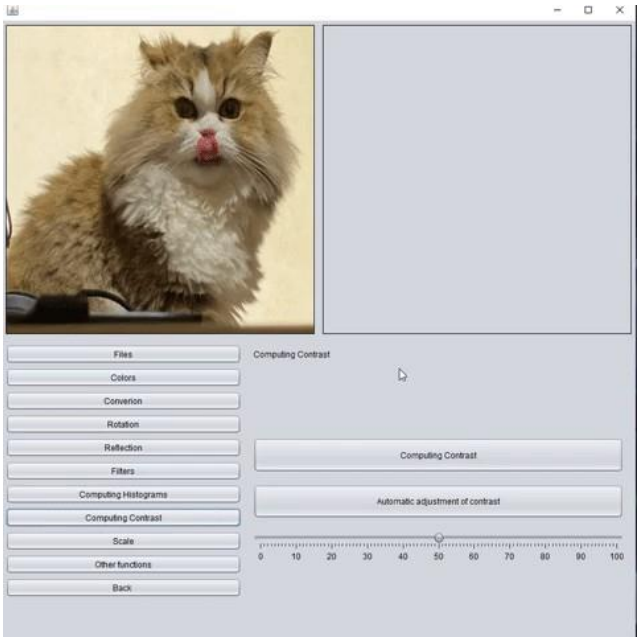
Export Histogram Code:

```
private void jButton11ActionPerformed(java.awt.event.ActionEvent evt) {
    this.toBack();
    setVisible(false);
    new menu().toFront();
    new menu().setState(java.awt.Frame.NORMAL);
}

private void jButton50ActionPerformed(java.awt.event.ActionEvent evt) {
    // exporting the histogram to files
    Pixel_LL[][] Histograms = ComputeHistograms();
    // ask the user where to save it
    JFileChooser f = new JFileChooser();
    f.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
    f.showSaveDialog(null);
    File export = new File(f.getSelectedFile() + "\\histograms.txt");

    PrintWriter pen = null;
    try {
        pen = new PrintWriter(export);
    } catch (FileNotFoundException ex) {
        Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
    }
    pen.println(picObj.getWidth() + " " + picObj.getHeight());
    pen.println("RED");
    for (int i = 0; i < 256; i++) {
        Histograms[0][i].exportHistogram(pen);
    }
    pen.println("GREEN");
    for (int i = 0; i < 256; i++) {
        Histograms[1][i].exportHistogram(pen);
    }
    pen.println("BLUE");
    for (int i = 0; i < 256; i++) {
        Histograms[2][i].exportHistogram(pen);
    }
    pen.close();
    JOptionPane.showMessageDialog(null, "Finished\n Exported to " + export.getAbsolutePath(), "Done", JOptionPane.INFORMATION_MESSAGE);
}
```

Import Histogram Code:

```java
private void jButton53ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        // TODO add your handling code here:

        JOptionPane.showMessageDialog(null, "Please choose the histogram txt", "File", JOptionPane.OK_OPTION);

        JFileChooser f = new JFileChooser();
        f.setFileSelectionMode(JFileChooser.FILES_AND_DIRECTORIES);
        f.showSaveDialog(null);
        File export = f.getSelectedFile();

        Scanner input = new Scanner(export);
        Picture replot = new Picture(input.nextInt(), input.nextInt());

        Pixel_LL[] HistogramsRed = new Pixel_LL[256];
        Pixel_LL[] HistogramsGreen = new Pixel_LL[256];
        Pixel_LL[] HistogramsBlue = new Pixel_LL[256];
        int total;
        String color = input.next();
        System.out.println("Importing the color " + color);
        for (int i = 0; i < 256; i++) { // Reading all level 0 pixels from red
            total = input.nextInt();
            //System.out.println("Importing level " + i + " of " + color + " it has: " + total + " pixels");
            HistogramsRed[i] = new Pixel_LL(input, total);
        }
        color = input.next();
        System.out.println("Importing the color " + color);
        for (int i = 0; i < 256; i++) { // Reading all level 0 pixels from green
            total = input.nextInt();
            //System.out.println("Importing level " + i + " of " + color + " it has: " + total + " pixels");
            HistogramsGreen[i] = new Pixel_LL(input, total);
        }
        color = input.next();
        System.out.println("Importing the color " + color);
        for (int i = 0; i < 256; i++) { // Reading all level 0 pixels from blue
            total = input.nextInt();
            //System.out.println("Importing level " + i + " of " + color + " it has: " + total + " pixels");
            HistogramsBlue[i] = new Pixel_LL(input, total);
        }
        PixelLinkedList_node helpPtr = null;

        for (int i = 0; i < 256; i++) { // 0 - 256
            if (HistogramsRed[i].getHead() != null) {
                helpPtr = HistogramsRed[i].getHead();
                while (helpPtr != null) {
                    replot.getPixel(helpPtr.getX(), helpPtr.getY()).setRed(i);
                    helpPtr = helpPtr.getNext();
                }
            }

            if (HistogramsGreen[i].getHead() != null) {
                helpPtr = HistogramsGreen[i].getHead();
                while (helpPtr != null) {
                    replot.getPixel(helpPtr.getX(), helpPtr.getY()).setGreen(i);
                    helpPtr = helpPtr.getNext();
                }
            }

            if (HistogramsBlue[i].getHead() != null) {
                helpPtr = HistogramsBlue[i].getHead();
                while (helpPtr != null) {
                    replot.getPixel(helpPtr.getX(), helpPtr.getY()).setBlue(i);
                    helpPtr = helpPtr.getNext();
                }
            }
        }
        System.out.println("DONE");
        replot.show();
    } catch (FileNotFoundException ex) {
        JOptionPane.showMessageDialog(null, "ERROR FILE IS NOT FOUND", "File", JOptionPane.OK_OPTION);
    } catch (Exception x) {
        JOptionPane.showMessageDialog(null, "ERROR", "EROR", JOptionPane.OK_OPTION);
    }
```

17- Computing Contrast:



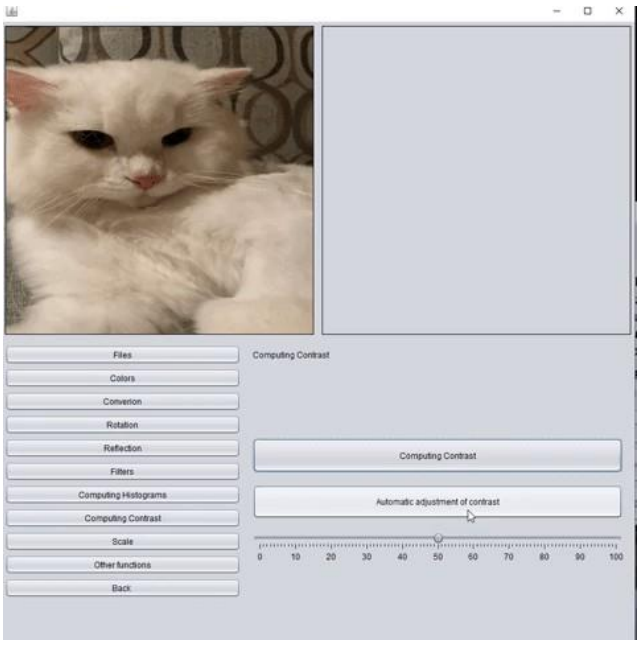
By pressing the "Computing contrast" button, users can see what the contrast level of an inserted picture is, ranging from 0 to 1.

Computing contrast code:

```
private void jButton38ActionPerformed(java.awt.event.ActionEvent evt) {
    //constrant
    int Contrast;
    double max_val = 0;
    double min_val = Integer.MAX_VALUE;
    Pixel[] pixels = picObj.getPixels();
    for (Pixel pixel : pixels) {
        Contrast = (int) ((pixel.getRed() + pixel.getGreen() + pixel.getBlue()) / 3);
        if (Contrast > max_val) {
            max_val = Contrast;
        }
        if (Contrast < min_val) {
            min_val = Contrast;
        }
    }
    double contrast = ((max_val - min_val) / (max_val + min_val));
    JOptionPane.showMessageDialog(null, "The level of Contrast is = " + contrast);
}
```

18-

18-Automatic adjustment of contrast using slider:



Users can adjust the contrast of the chosen picture by using a slider. Moving the slider to the right increases the contrast, while moving it to the left decreases it. To ensure the changes take effect, users must press the "Automatic adjustment of contrast" button.

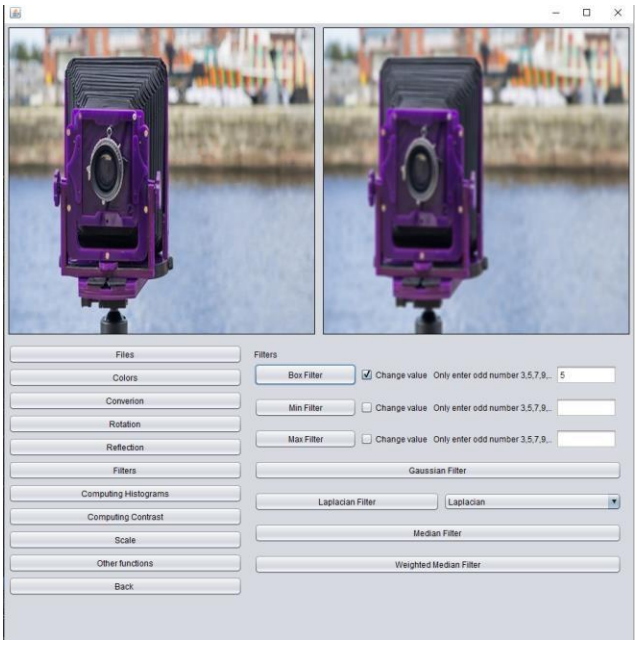
Adjustment of contrast using a slider code:

```
private void jButton40ActionPerformed(java.awt.event.ActionEvent evt) {
    int x = jSlider4.getValue();
    int y = (int) ((x / 100.00) * 255);
    int aLowR = 257, aHighR = -1, aLowG = 257, aHighG = -1, aLowB = 257, aHighB = -1;
    for (int i = 0; i < picObj.getWidth(); i++) {
        for (int j = 0; j < picObj.getHeight(); j++) {
            int tempR = picObj.getPixel(i, j).getRed();
            if (tempR < aLowR) {
                aLowR = tempR;
            }
            if (tempR > aHighR) {
                aHighR = tempR;
            }
            int tempG = picObj.getPixel(i, j).getGreen();
            if (tempG < aLowG) {
                aLowG = tempG;
            }
            if (tempG > aHighG) {
                aHighG = tempG;
            }
            int tempB = picObj.getPixel(i, j).getBlue();
            if (tempB < aLowB) {
                aLowB = tempB;
            }
            if (tempB > aHighB) {
                aHighB = tempB;
            }

        }
    }
    int aMin = 0;
    for (int i = 0; i < picObj.getWidth(); i++) {
        for (int j = 0; j < picObj.getHeight(); j++) {
            int equatR = aMin + (int) ((picObj.getPixel(i, j).getRed() - aLowR) * ((double) (y - aMin) / (aHighR - aLowR)));
            int equatG = aMin + (int) ((picObj.getPixel(i, j).getGreen() - aLowG) * ((double) (y - aMin) / (aHighG - aLowG)));
            int equatB = aMin + (int) ((picObj.getPixel(i, j).getBlue() - aLowB) * ((double) (y - aMin) / (aHighB - aLowB)));
            picObj.getPixel(i, j).setColor(new Color(equatR, equatG, equatB));
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

19 + 20 + 21 + 22 + 23- Filters(Box,Gaussian,Laplacian,Min,Max,Median and Weighted median)

To apply Box, min and max filters, users must enter an odd number, such as 3, 5, 7 or 9, and then check the corresponding box and press the button for the filter they want. For Gaussian, median and weighted median filters, simply press the button to apply the changes. For Laplacian filter, users must select which Laplacian change to implement and then press the “Laplacian filter” button to apply the changes.

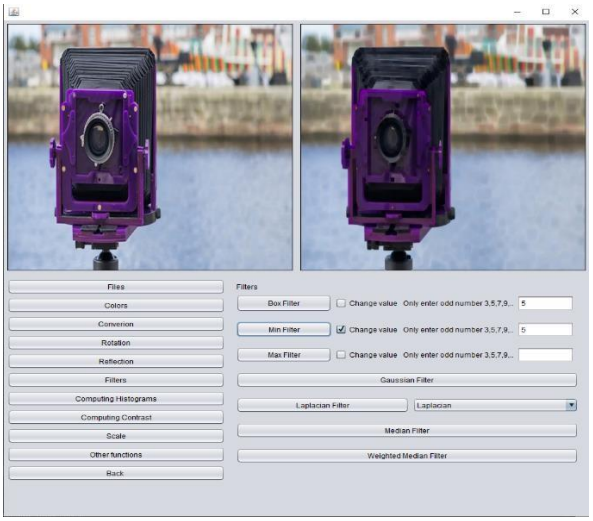
Box Filter:



Box Filter code:

```java
private void jButton31ActionPerformed(java.awt.event.ActionEvent evt) {

    int sof = 3;
    if (jCheckBox5.isSelected()) {

        sof = Integer.valueOf(jTextField2.getText());
    }

    int x = picObj.getWidth();
    int y = picObj.getHeight();
    int q = sof / 2;

    Picture Image = new Picture(x, y);
    Image.copy(picObj, 0, 0, x, y, 0, 0);
    for (int v = q; v <= y - (q + 1); v++) {
        for (int u = q; u <= x - (q + 1); u++) {

            int sumRed = 0, sumGreen = 0, sumBlue = 0;
            for (int j = -q; j <= q; j++) {
                for (int i = -q; i <= q; i++) {

                    int red = Image.getPixel(u + i, v + j).getRed();
                    sumRed += red;
                    int green = Image.getPixel(u + i, v + j).getGreen();
                    sumGreen += green;
                    int blue = Image.getPixel(u + i, v + j).getBlue();
                    sumBlue += blue;
                }
            }
            int Red = (int) Math.round(sumRed / (sof * sof));
            int Green = (int) Math.round(sumGreen / (sof * sof));
            int Blue = (int) Math.round(sumBlue / (sof * sof));
            picObj.getPixel(u, v).setColor(new Color(Red, Green, Blue));
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
    // TODO add your handling code here:
}
```

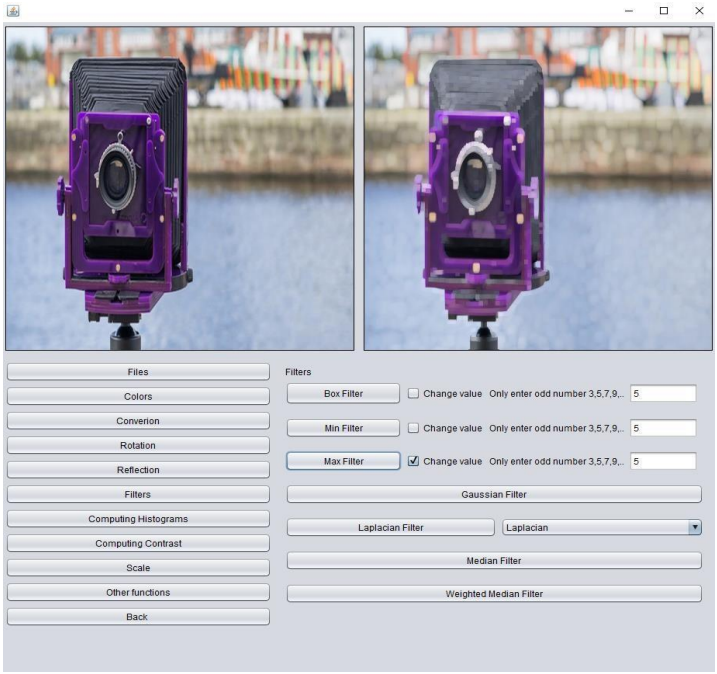
Min Filter:



Min Filter Code:

```
private void jButton32ActionPerformed(java.awt.event.ActionEvent evt) {

    int sof = 3;
    if (jCheckBox6.isSelected()) {

        sof = Integer.valueOf(jTextField3.getText());
    }
    int x = picObj.getWidth();
    int y = picObj.getHeight();
    int q = sof / 2;
    Picture copy = new Picture(x, y);
    copy.copy(picObj, 0, 0, x, y, 0, 0);
    int Red[] = new int[sof * sof];
    int Green[] = new int[sof * sof];
    int Blue[] = new int[sof * sof];
    for (int v = q; v <= y - (q + 1); v++) {
        for (int u = q; u <= x - (q + 1); u++) {
            int k = 0;
            for (int j = -q; j <= q; j++) {
                for (int i = -q; i <= q; i++) {
                    Red[k] = copy.getPixel(u + i, v + j).getRed();
                    Green[k] = copy.getPixel(u + i, v + j).getGreen();
                    Blue[k] = copy.getPixel(u + i, v + j).getBlue();
                    k++;
                }
            }
            Arrays.sort(Red);
            Arrays.sort(Green);
            Arrays.sort(Blue);
            picObj.getPixel(u, v).setColor(new Color(Red[0], Green[0], Blue[0]));
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

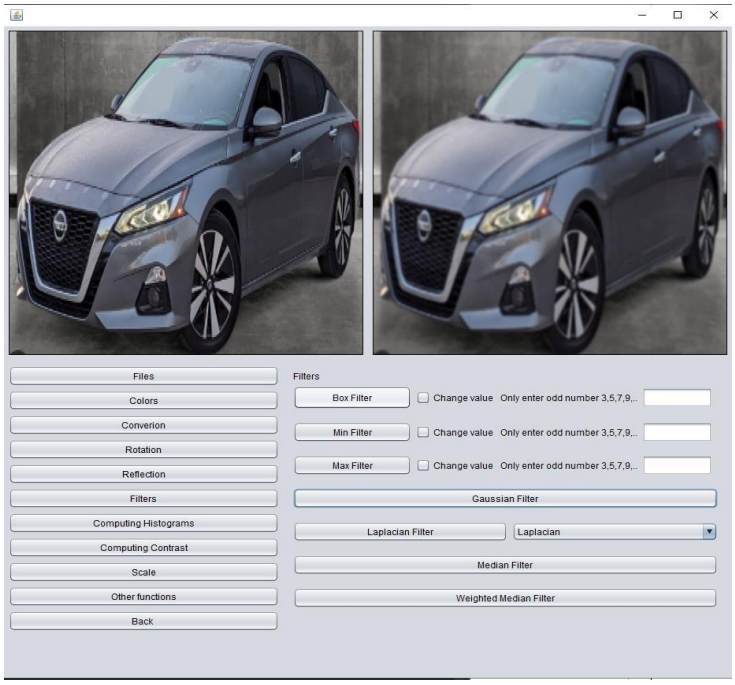
Max Filter:



Max Filter Code:

```
private void jButton33ActionPerformed(java.awt.event.ActionEvent evt) {
    int sof = 3;
    if (jCheckBox7.isSelected()) {

        sof = Integer.valueOf(jTextField4.getText());
    }
    int x = picObj.getWidth();
    int y = picObj.getHeight();
    int q = sof / 2;
    Picture copy = new Picture(x, y);
    copy.copy(picObj, 0, 0, x, y, 0, 0);
    int Red[] = new int[sof * sof];
    int Green[] = new int[sof * sof];
    int Blue[] = new int[sof * sof];
    for (int v = q; v <= y - (q + 1); v++) {
        for (int u = q; u <= x - (q + 1); u++) {
            int k = 0;
            for (int j = -q; j <= q; j++) {
                for (int i = -q; i <= q; i++) {
                    Red[k] = copy.getPixel(u + i, v + j).getRed();
                    Green[k] = copy.getPixel(u + i, v + j).getGreen();
                    Blue[k] = copy.getPixel(u + i, v + j).getBlue();
                    k++;
                }
            }
            Arrays.sort(Red);
            Arrays.sort(Green);
            Arrays.sort(Blue);
            picObj.getPixel(u, v).setColor(new Color(Red[sof * sof - 1], Green[sof * sof - 1], Blue[sof * sof - 1]));
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

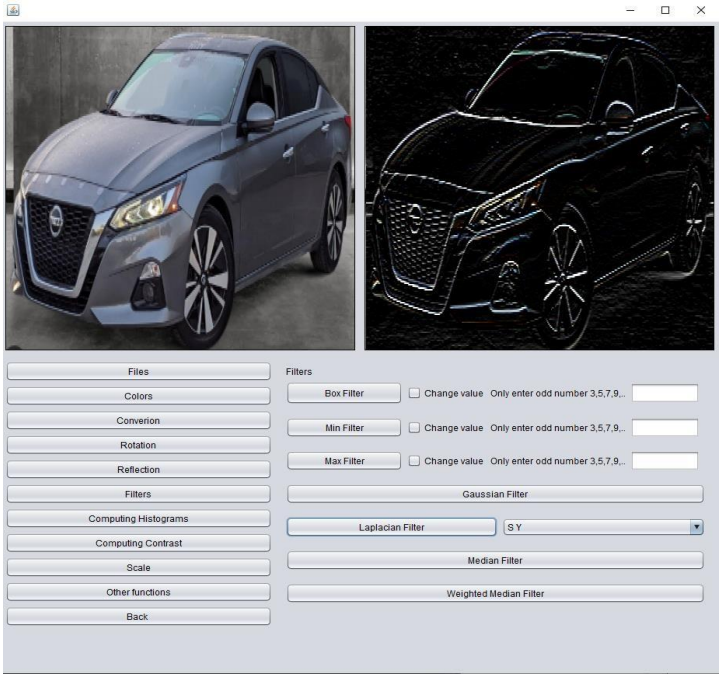
Gaussian Filter:



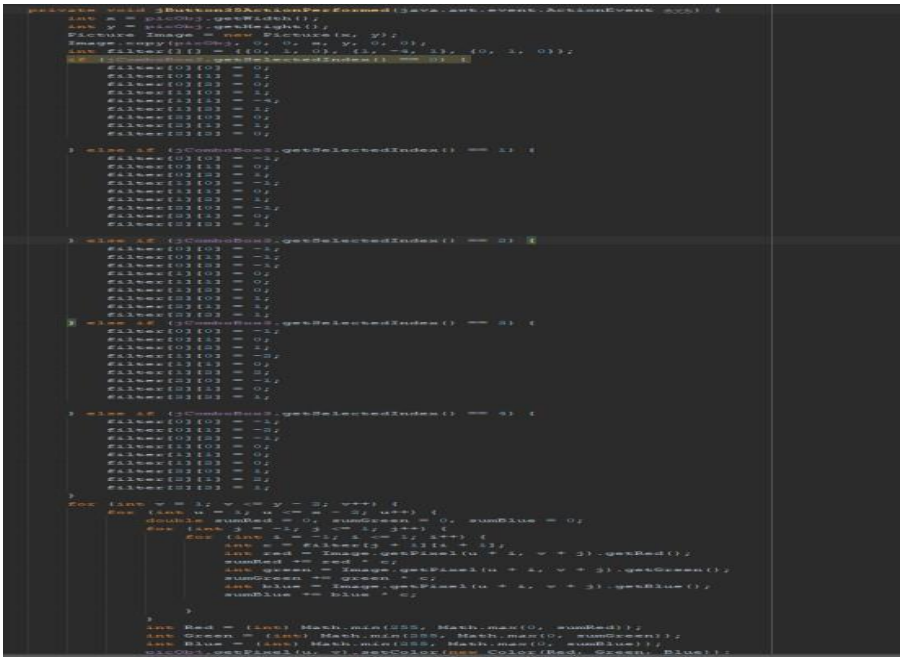
Gaussian Filter Code:

```java
private void jButton34ActionPerformed(java.awt.event.ActionEvent evt) {
    int x = picObj.getWidth();
    int y = picObj.getHeight();
    Picture Image = new Picture(x, y);
    Image.copy(picObj, 0, 0, x, y, 0, 0);
    double filter[][] = {{0.075, 0.125, 0.075},
    {0.125, 0.200, 0.125},
    {0.075, 0.125, 0.075}};
    for (int v = 1; v <= y - 2; v++) {
        for (int u = 1; u <= x - 2; u++) {
            double sumRed = 0, sumGreen = 0, sumBlue = 0;
            for (int j = -1; j <= 1; j++) {
                for (int i = -1; i <= 1; i++) {
                    double c = filter[j + 1][i + 1];
                    int red = Image.getPixel(u + i, v + j).getRed();
                    sumRed += red * c;
                    int green = Image.getPixel(u + i, v + j).getGreen();
                    sumGreen += green * c;
                    int blue = Image.getPixel(u + i, v + j).getBlue();
                    sumBlue += blue * c;

                }
            }
            int Red = (int) Math.round(sumRed);
            int Green = (int) Math.round(sumGreen);
            int Blue = (int) Math.round(sumBlue);
            picObj.getPixel(u, v).setColor(new Color(Red, Green, Blue));
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

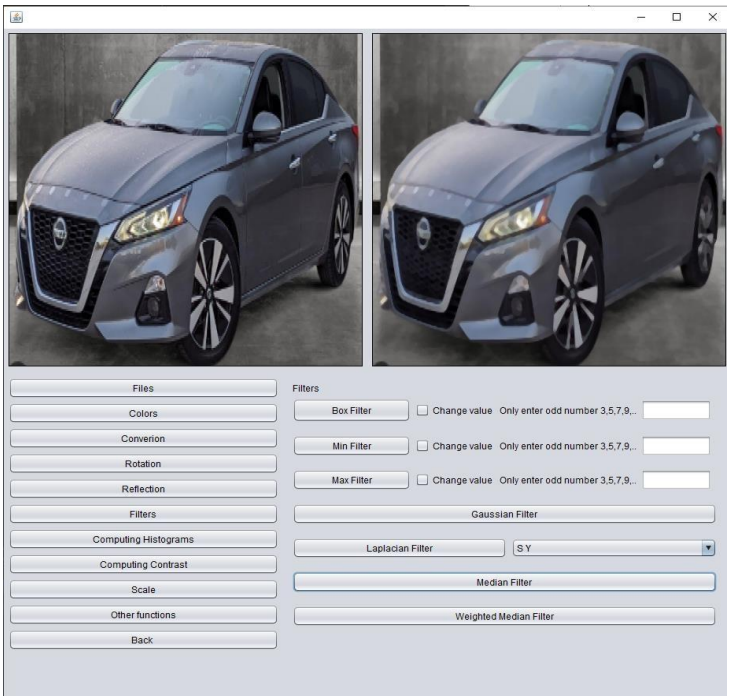
Laplacian Filter:



Laplacian Filter Code:



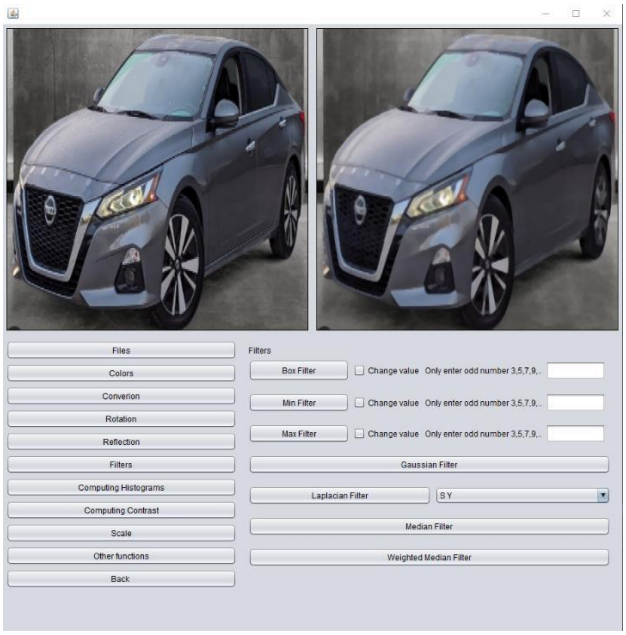
Median Filter:



Median Filter Code:

```
private void jButton36ActionPerformed(java.awt.event.ActionEvent evt) {
    int x = picObj.getWidth();
    int y = picObj.getHeight();
    Picture copy = new Picture(x, y);
    copy.copy(picObj, 0, 0, x, y, 0, 0);
    int Red[] = new int[9];
    int Green[] = new int[9];
    int Blue[] = new int[9];
    for (int v = 1; v <= y - 2; v++) {
        for (int u = 1; u <= x - 2; u++) {
            int k = 0;
            for (int j = -1; j <= 1; j++) {
                for (int i = -1; i <= 1; i++) {
                    Red[k] = copy.getPixel(u + i, v + j).getRed();
                    Green[k] = copy.getPixel(u + i, v + j).getGreen();
                    Blue[k] = copy.getPixel(u + i, v + j).getBlue();
                    k++;
                }
            }

            Arrays.sort(Red);
            Arrays.sort(Green);
            Arrays.sort(Blue);
            picObj.getPixel(u, v).setColor(new Color(Red[4], Green[4], Blue[4]));
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

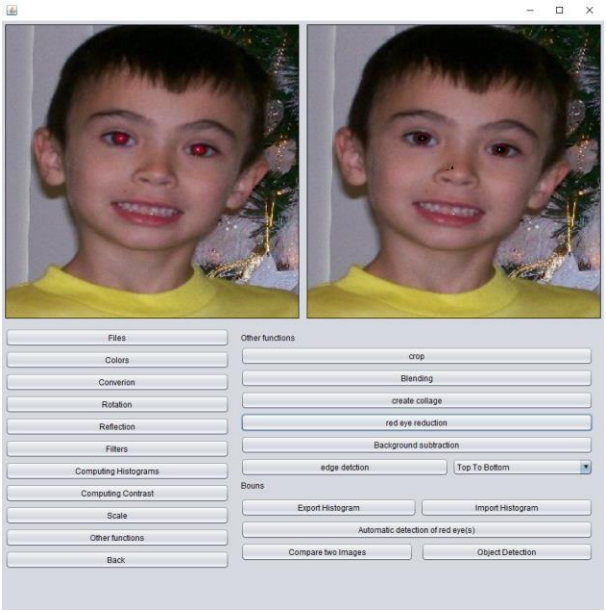
Weighted median Filter:



Weighted median Filter Code:

```
private void jButton37ActionPerformed(java.awt.event.ActionEvent evt) {
    int x = picObj.getWidth();
    int y = picObj.getHeight();
    Picture copy = new Picture(x, y);
    copy.copy(picObj, 0, 0, x, y, 0, 0);
    int Weight[] = {1, 2, 1, 2, 3, 2, 1, 2, 1};
    int sumW = 0;
    for (int i = 0; i < Weight.length; i++) {
        sumW += Weight[i];
    }
    int Red[] = new int[9];
    int Green[] = new int[9];
    int Blue[] = new int[9];
    int Redw[] = new int[sumW];
    int Greenw[] = new int[sumW];
    int Bluew[] = new int[sumW];

    for (int v = 1; v <= y - 2; v++) {
        for (int u = 1; u <= x - 2; u++) {
            int k = 0;
            for (int j = -1; j <= 1; j++) {
                for (int i = -1; i <= 1; i++) {
                    Red[k] = copy.getPixel(u + i, v + j).getRed();
                    Green[k] = copy.getPixel(u + i, v + j).getGreen();
                    Blue[k] = copy.getPixel(u + i, v + j).getBlue();
                    k++;
                }
            }
            int time;
            int red;
            int green;
            int blue;
            int counter = 0;
            for (int i = 0; i < Weight.length; i++) {
                time = Weight[i];
                red = Red[i];
                green = Green[i];
                blue = Blue[i];
                for (int j = 0; j < time; j++) {
                    Redw[counter] = red;
                    Greenw[counter] = green;
                    Bluew[counter] = blue;
                    counter++;
                }

            }

            Arrays.sort(Redw);
            Arrays.sort(Greenw);
            Arrays.sort(Bluew);
            picObj.getPixel(u, v).setColor(new Color(Redw[Redw.length / 2], Greenw[Greenw.length / 2], Bluew[Bluew.length / 2]));
        }
    }
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

24- Red Eye Reduction using thresholding technique:

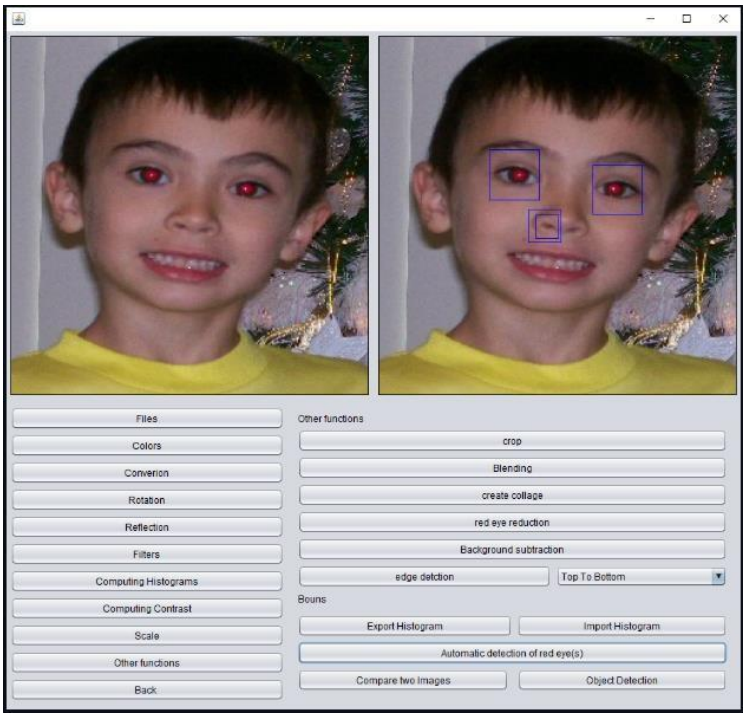


By pressing the "Red Eye Reduction" button, the user can reduce the red eye in the selected picture. The user will then be prompted to select the area of the picture with the red eye and choose a color to replace it. Finally, the user will adjust the thresholding value and the resulting image will be displayed without the red eye.

Red Eye Reduction using thresholding technique Code:

```
private void jButton49ActionPerformed(java.awt.event.ActionEvent evt) {
    // redeye
    if (picObj == null) {
        JOptionPane.showMessageDialog(null, "Select an image please!", "Error", JOptionPane.ERROR_MESSAGE);
    } else {
        JFrame parent = new JFrame();
        JOptionPane.showMessageDialog(parent, "Please click on two points in the image");

        jLabel1.addMouseListener(new MouseAdapter() {
            int numOfClicks = 0;
            int x1;
            int x2;
            int y1;
            int y2;

            public void mouseClicked(MouseEvent e) {
                System.out.println("Clicked!");
                System.out.println(e.getX());
                System.out.println(e.getY());
                numOfClicks++;
                if (numOfClicks == 1) {
                    x1 = e.getX();
                    y1 = e.getY();
                } else if (numOfClicks == 2) {
                    x2 = e.getX();
                    y2 = e.getY();

                    double W = (picObj.getWidth() * 1.00 / jLabel1.getWidth());
                    double H = (picObj.getHeight() * 1.00 / jLabel1.getHeight());

                    x1 = (int) (W * x1);
                    x2 = (int) (W * x2);
                    y1 = (int) (H * y1);
                    y2 = (int) (H * y2);

                    Color newColor = JColorChooser.showDialog(null, "Choose New Color", Color.BLACK);
                    int trashhold = Integer.parseInt(JOptionPane.showInputDialog("TrashHold?"));
                    for (int i = y1; i < y2; i++) {
                        for (int j = x1; j < x2; j++) {
                            Pixel p = picObj.getPixel(j, i);
                            //here we compare because get the different between Red color amd pxl .
                            if (p.colorDistance(Color.RED) < trashhold) {
                                p.setColor(newColor);
                            }
                        }
                    }
                    numOfClicks = 0;
                    jLabel1.removeMouseListener(this);
                    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
                    jLabel2.setText("");
                    jLabel2.setIcon(new ImageIcon(img));
                }
            }
        });
    }
}
```

25- Automatic detection of red eyes:



To detect red eyes in an image, simply press the "Automatic Detection of Red Eyes" button. This feature utilizes OpenCV, an external library, to identify and outline the red eyes in the image with a rectangle. The result is a picture that clearly highlights the red eyes.

Automatic detection of red eyes Code:

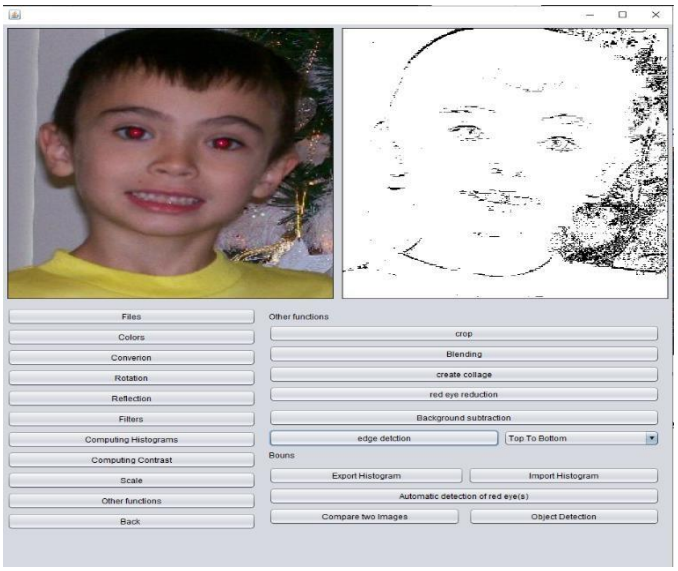
```
private void jButton54ActionPerformed(java.awt.event.ActionEvent evt) {
    System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
    CascadeClassifier faceDetector = new CascadeClassifier("opencv\\sources\\data\\haarcascades\\haarcascade_frontalface_default.xml");
    CascadeClassifier eyeDetector = new CascadeClassifier("opencv\\sources\\data\\haarcascades\\haarcascade_eye.xml");
    String imagePath = pathName;
    Mat image = Imgcodecs.imread(imagePath);
    MatOfRect faceDetections = new MatOfRect();
    faceDetector.detectMultiScale(image, faceDetections);

    for (Rect rect : faceDetections.toArray()) {
        Mat faceROI = new Mat(image, rect);
        MatOfRect eyes = new MatOfRect();
        eyeDetector.detectMultiScale(faceROI, eyes);

        for (Rect eye : eyes.toArray()) {
            Point center = new Point(eye.x + eye.width / 2, eye.y + eye.height / 2);
            Imgproc.rectangle(image, new Point(eye.x + rect.x, eye.y + rect.y),
                    new Point(eye.x + rect.x + eye.width, eye.y + rect.y + eye.height), new Scalar(255, 0, 0));
        }
    }

    Imgcodecs.imwrite("Tmp\\ARE.jpg", image);
    Picture first = new Picture("Tmp\\ARE.jpg");
    Image img = (first.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);

}
```

26- Background Subtraction:

Background Subtraction Code:

```
private void jButton51ActionPerformed(java.awt.event.ActionEvent evt) {
    Pixel currPixel = null;
    Pixel oldPixel = null;
    Pixel newPixel = null;
    JOptionPane.showMessageDialog(null, "Enter the background");
    Picture first = new Picture(FileChooser.pickAFile());
    JOptionPane.showMessageDialog(null, "Enter the forground");
    Picture secend = new Picture(FileChooser.pickAFile());
    String size = JOptionPane.showInputDialog(null, "Please Enter the threshold value: ");
    for (int x = 0; x < first.getWidth(); x++) {
        for (int y = 0; y < first.getHeight(); y++) {

            currPixel = first.getPixel(x, y);
            oldPixel = secend.getPixel(x, y);

            if (currPixel.colorDistance(oldPixel.getColor()) < Integer.valueOf(size)) {
                newPixel = picObj.getPixel(x, y);
                currPixel.setColor(newPixel.getColor());
            }
        }
    }
    picObj = first;
    Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);
}
```

27- Edge Detection methods:



Users can select an image and then choose from one of the three options in the combo box to apply edge detection - top-bottom detection, left-right detection, or merge (which combines the two detections in one picture). Once they have made their selection, they must press the "Edge Detection" button to view the result.

Edge Detection methods Code:

```java
private void jButton52ActionPerformed(java.awt.event.ActionEvent evt) {
    // edge detection
    String size = JOptionPane.showInputDialog(null, "Please Enter the threshold value: ");
    try {
        // Top to bottom
        if (jComboBox2.getSelectedIndex() == 0) {
            Pixel topPixel = null;
            Pixel bottomPixel = null;
            double topAverage = 0.0;
            double bottomAverage = 0.0;
            for (int y = 0; y < picObj.getHeight() - 1; y++) {
                for (int x = 0; x < picObj.getWidth(); x++) {
                    topPixel = picObj.getPixel(x, y);
                    bottomPixel = picObj.getPixel(x, y + 1);
                    topAverage = topPixel.getAverage();
                    bottomAverage = bottomPixel.getAverage();
                    if (Math.abs(topAverage - bottomAverage) < Integer.valueOf(size)) {
                        topPixel.setColor(Color.WHITE);
                    } else {
                        topPixel.setColor(Color.BLACK);
                    }
                }
            }
        } // left to right
        else if (jComboBox2.getSelectedIndex() == 1) {
            Pixel LeftPixel = null;
            Pixel RightPixel = null;
            double LeftAverage = 0.0;
            double RightAverage = 0.0;
            for (int y = 0; y < picObj.getHeight(); y++) {
                for (int x = 0; x < picObj.getWidth() - 1; x++) {
                    LeftPixel = picObj.getPixel(x, y);
                    RightPixel = picObj.getPixel(x + 1, y);
                    LeftAverage = LeftPixel.getAverage();
                    RightAverage = RightPixel.getAverage();
                    if (Math.abs(LeftAverage - RightAverage) < Integer.valueOf(size)) {
                        LeftPixel.setColor(Color.WHITE);
                    } else {
                        LeftPixel.setColor(Color.BLACK);
                    }
                }
            }
        } // Merge
        else if (jComboBox2.getSelectedIndex() == 2) {
            Pixel topPixel = null;
            Pixel bottomPixel = null;
            double topAverage = 0.0;
            double bottomAverage = 0.0;
            for (int y = 0; y < picObj.getHeight() - 1; y++) {
                for (int x = 0; x < picObj.getWidth(); x++) {
                    topPixel = picObj.getPixel(x, y);
                    bottomPixel = picObj.getPixel(x, y + 1);
                    topAverage = topPixel.getAverage();
                    bottomAverage = bottomPixel.getAverage();
                    if (Math.abs(topAverage - bottomAverage) < Integer.valueOf(size)) {
                        topPixel.setColor(Color.WHITE);
                    } else {
                        topPixel.setColor(Color.BLACK);
                    }
                }
            }
```

```java
        Image img = (picObj.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
        BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
        Graphics2D bGr = bimage.createGraphics();
        bGr.drawImage(img, 0, 0, null);
        bGr.dispose();
        File outputfile = new File("Tmp\\Tbeds.png");
        ImageIO.write(bimage, "png", outputfile);
        picObj = new Picture(pathName);

        //////
        Pixel LeftPixel = null;
        Pixel RightPixel = null;
        double LeftAverage = 0.0;
        double RightAverage = 0.0;
        for (int y = 0; y < picObj.getHeight(); y++) {
            for (int x = 0; x < picObj.getWidth() - 1; x++) {
                LeftPixel = picObj.getPixel(x, y);
                RightPixel = picObj.getPixel(x + 1, y);
                LeftAverage = LeftPixel.getAverage();
                RightAverage = RightPixel.getAverage();
                if (Math.abs(LeftAverage - RightAverage) < Integer.valueOf(size)) {
                    LeftPixel.setColor(Color.WHITE);
                } else {
                    LeftPixel.setColor(Color.BLACK);
                }
            }
        }

        Image img2 = (picObj.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
        BufferedImage bimage2 = new BufferedImage(img2.getWidth(null), img2.getHeight(null), BufferedImage.TYPE_INT_ARGB);
        Graphics2D bGr2 = bimage2.createGraphics();
        bGr2.drawImage(img2, 0, 0, null);
        bGr2.dispose();
        File outputfile2 = new File("Tmp\\Lbedg.png");
        ImageIO.write(bimage2, "png", outputfile2);
        picObj = new Picture(pathName);

        Picture f = new Picture("Tmp\\Tbedg.png");
        Picture s = new Picture("Tmp\\Lbedg.png");

        Pixel[] pixelsArrayf = f.getPixels();
        Pixel pixelArrayf = null;

        Pixel[] pixelsArrays = s.getPixels();
        Pixel pixelArrays = null;

        Pixel[] pixelsArray = picObj.getPixels();
        Pixel pixelArray = null;

        for (int i = 0; i < pixelsArray.length; i++) {
            pixelArrayf = pixelsArrayf[i];
            pixelArrays = pixelsArrays[i];
            pixelArray = pixelsArray[i];
            if !((pixelArrayf.getRed() == 0) && (pixelArrayf.getGreen() == 0) &&
                    (pixelArrayf.getGreen() == 0)) || ((pixelArrays.getRed() == 0) && (pixelArrays.getGreen() == 0) && (pixelArrays.getGreen() == 0))) {
                pixelArray.setColor(Color.BLACK);
            } else {
                pixelArray.setColor(Color.WHITE);
            }
        }
    }

} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(null, "Please put integer only", "Error", JOptionPane.ERROR_MESSAGE);
} catch (IOException ex) {
    Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
}
Image img = (picObj.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
icon = new ImageIcon(img);
jLabel2.setIcon(icon);
```

28- Change Volume Using Slider:

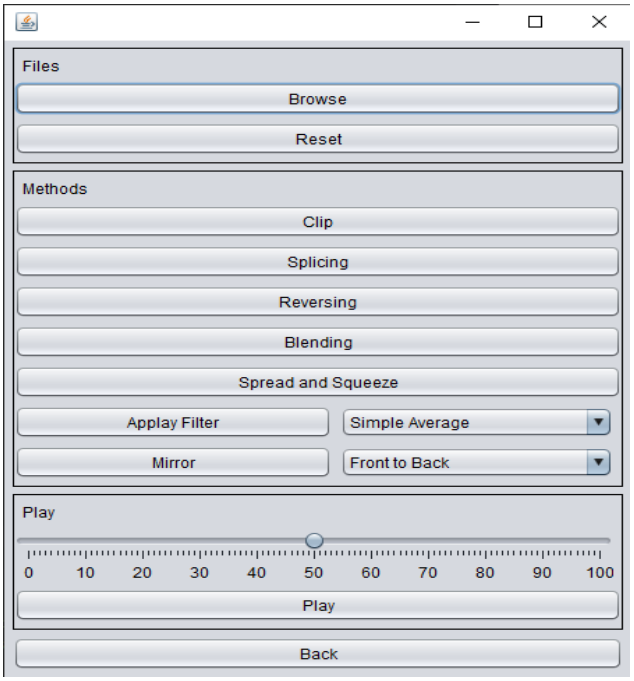
** For listening the sound go to GitHub

Users can adjust the volume of their chosen sound using a slider. Moving the slider to the right increases the volume, while dragging it left decreases the volume. To hear the adjusted sound, the user must press the "Play" button.

Change Volume using a slider code:

```
private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {
    Sound soundl = new Sound(currentSound);
    SoundSample[] sampleArray = soundl.getSamples();
    int jv = jSliderl.getValue();
    for (SoundSample sample : sampleArray) {
        sample.setValue((int) (sample.getValue() * (jv / 100.00)));
    }
    soundl.play();
}
```

29 - Clipping a given sound:



** For listening the sound go to GitHub

Users can clip the sound of their selection by pressing the "Clip" button and then specifying the start and end index. To hear the result, press the "Play" button.

Clip code:

```
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

    String first = JOptionPane.showInputDialog(null, "Enter the start Index the lowest index is 0: ");
    String last = JOptionPane.showInputDialog(null, "Enter the end Index the max value is "
            + (currentSound.getLengthInFrames() - 1) + ": ");
    int start = Integer.parseInt(first);
    int end = Integer.parseInt(last);
    if (start < 0 || end >= currentSound.getLengthInFrames()) {
        JOptionPane.showMessageDialog(null, "wrong input the start must equal 0 or above and the end must be lower than "
                + currentSound.getLengthInFrames());
        first = JOptionPane.showInputDialog(null, "Enter the start Index the lowest index is 0: ");
        last = JOptionPane.showInputDialog(null, "Enter the end Index the max value is " + (currentSound.getLengthInFrames() - 1) + ": ");
    }

    // calculate the number of samples in the clip
    int lengthInSamples = end - start + 1; //s1
    int lengthTarget = lengthInSamples;
    start = Integer.parseInt(first);
    end = Integer.parseInt(last);
    Sound target = new Sound(lengthTarget);
    int value = 0;
    int targetIndex = 0;
    for (int i = start; i <= end; i++, targetIndex++) {
        value = currentSound.getSampleValueAt(i);
        target.setSampleValueAt(targetIndex, value);
    }
    currentSound = target;


}
```

30- Splicing at least two sounds together keeping the silent zone:

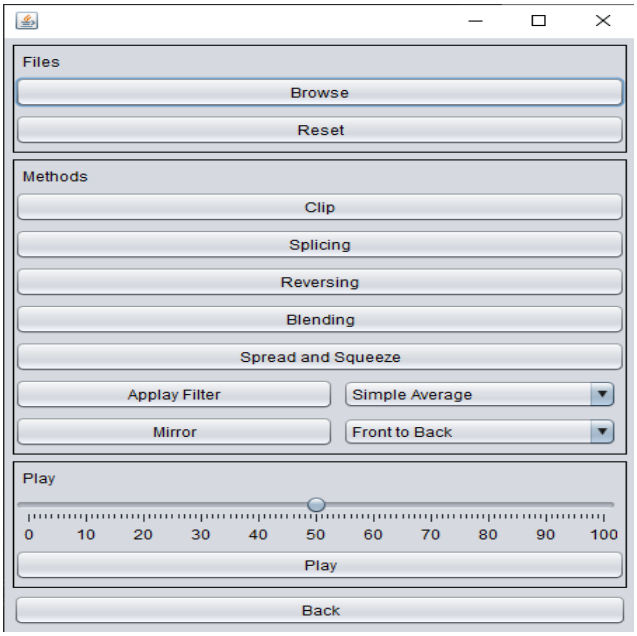
** For listening the sound go to GitHub

To splice two or more sounds together with a silent zone between them, users must first press the "Splicing" button. They will be asked to enter the number of seconds of silence they'd like between the sounds. After inserting the second sound, they will be asked if they'd like to splice another sound. To hear the result, users must press the "Play" button.

Splice code:

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    String first = JOptionPane.showInputDialog(null, "Enter silence sec:");
    while (true) {
        int zone = Integer.parseInt(first);
        Sound sound1 = new Sound(currentSound);
        Sound sound2 = new Sound(FileChooser.pickAFile());
        int targetIndex = 0;
        Sound s = new Sound((currentSound.getLength() + sound2.getLength()) + (int) (currentSound.getSamplingRate() * (Math.ceil(zone))));
        for (int i = 0; i < sound1.getLength(); i++, targetIndex++) {
            s.setSampleValueAt(targetIndex, currentSound.getSampleValueAt(i));
        }
        for (int i = 0; i < (int) (currentSound.getSamplingRate() * zone); i++, targetIndex++) {
            s.setSampleValueAt(targetIndex, 0);
        }
        for (int i = 0; i < sound2.getLength(); i++, targetIndex++) {
            s.setSampleValueAt(targetIndex, sound2.getSampleValueAt(i));
        }
        currentSound = s;
        int reply = JOptionPane.showConfirmDialog(null, "To splice another sound Press yes", null, JOptionPane.YES_NO_OPTION);
        if (reply == JOptionPane.YES_OPTION) {
            continue;
        } else {
            break;
        }
    }
```

31- Spread & squeeze a sound:



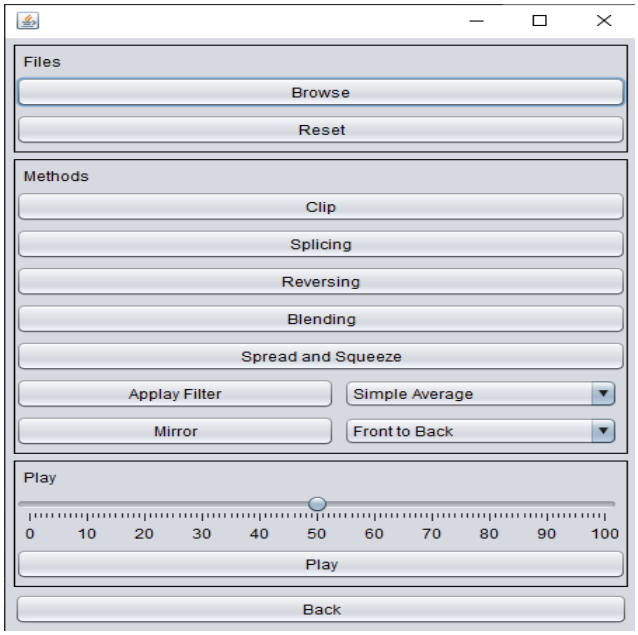
** For listening the sound go to GitHub

By pressing the "Spread and Squeeze" button, the user can manipulate sound, selecting it first. The sound will be spread into a 4 second duration and then squeezed into 1 second. The outcome can be heard by pressing the "Play" button.

Spread & squeeze code:

```
private void jButton11ActionPerformed(java.awt.event.ActionEvent evt) {
    Sound s = new Sound(pathName);
    SoundSample[] sample = currentSound.getSamples();

    for (double sourceIndex = 0, targetIndex = 0; targetIndex < currentSound.getLength(); sourceIndex = sourceIndex + 0.5, targetIndex++) {
        currentSound.setSampleValueAt((int) targetIndex, s.getSampleValueAt((int) sourceIndex));
    }

    s = new Sound(pathName);
    for (int sourceIndex = 0, targetIndex = 0; sourceIndex < currentSound.getLength(); sourceIndex = sourceIndex + 4, targetIndex++) {
        currentSound.setSampleValueAt(targetIndex, s.getSampleValueAt(sourceIndex));
    }
    for (int i = currentSound.getLength() / 4; i < currentSound.getLength(); i++) {
        currentSound.setSampleValueAt(i, 0);
    }


}
```

32- Reversing and Mirroring:



** For listening the sound go to GitHub

Users can reverse the sound they have selected by pressing the "Reversing" button. They can also mirror the sound by selecting the desired option from the combo box and pressing the "Mirror" button.

Once they have made the desired changes, they can press the "Play" button to hear the altered sound.

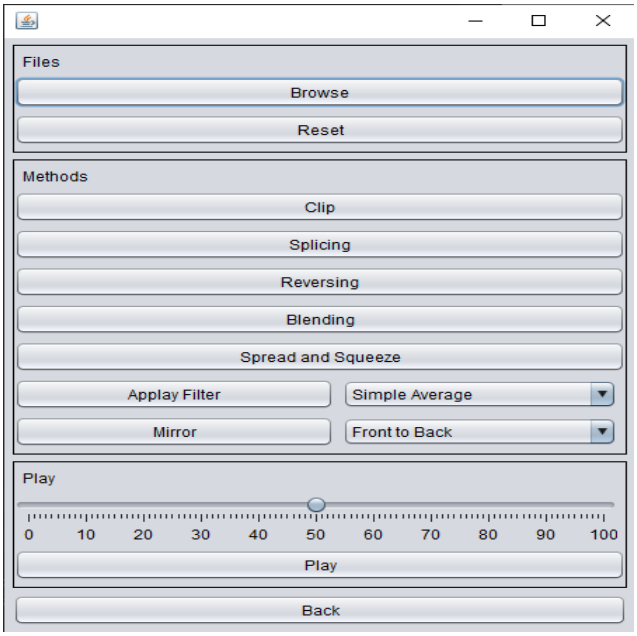
Reversing code:

```
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
    Sound orig = new Sound(pathName);
    int length = currentSound.getLength();
    for (int targetIndex = 0, sourceIndex = length - 1;
            targetIndex < length && sourceIndex > 0;
            targetIndex++, sourceIndex--) {
        currentSound.setSampleValueAt(targetIndex, orig.getSampleValueAt(sourceIndex));
    }
}
```

Mirror code:

```
private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {

    if (jComboBox2.getSelectedIndex() == 0) {
        //FtoB
        int length = currentSound.getLength(); // save the length
        int mirrorPoint = length / 2; // mirror around this
        int value = 0; // hold the current value

        // loop from 0 to mirrorPoint
        for (int i = 0; i < mirrorPoint; i++) {
            value = currentSound.getSampleValueAt(i);
            currentSound.setSampleValueAt(length - 1 - i, value);
        }
    } else if (jComboBox2.getSelectedIndex() == 1) {
        //BtoF
        int length = currentSound.getLength(); // save the length
        int mirrorPoint = length / 2; // mirror around this
        int value = 0; // hold the current value

        // loop from 0 to mirrorPoint
        for (int i = 0; i < mirrorPoint; i++) {
            value = currentSound.getSampleValueAt(length - 1 - i);
            currentSound.setSampleValueAt(i, value);
        }
    }
}
```

33- Blending Two or more sounds:



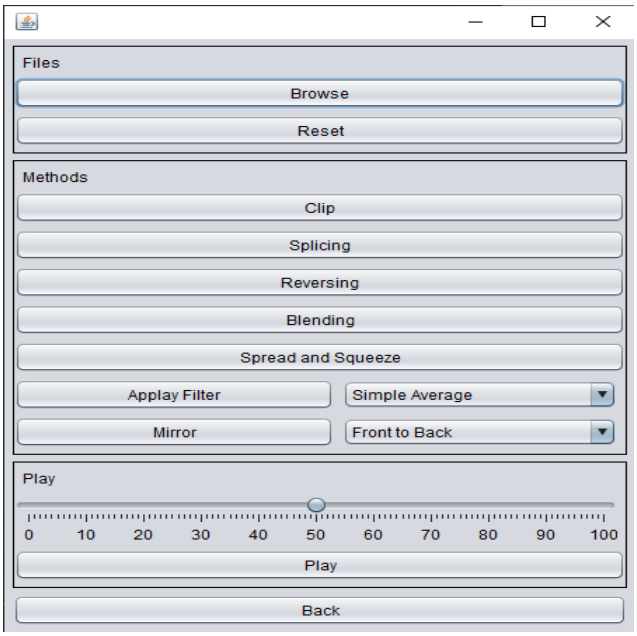
** For listening the sound go to GitHub

Users can blend two or more sounds into one sound by pressing the "Blending" button. Then, they will select the desired sound and blend it to the already selected sound. The program will then ask if they want to add more sounds or not. If yes, they will select another sound. Finally, they must press the "Play" button to hear the blended sound.

Blend code:

```
private void jButton10ActionPerformed(java.awt.event.ActionEvent evt) {
    while (true) {
        Sound sound1 = new Sound(FileChooser.pickAFile());
        if ((currentSound.getLength() - 1) > (sound1.getLength() - 1)) {
            for (int i = 0; i < currentSound.getLength() - 1; i++) {
                if (i >= sound1.getLengthInFrames()) {
                    currentSound.setSampleValueAt(i, currentSound.getSampleValueAt(i));
                } else {
                    currentSound.setSampleValueAt(i, currentSound.getSampleValueAt(i) + sound1.getSampleValueAt(i));
                }
            }
        } else {
            for (int i = 0; i < sound1.getLength() - 1; i++) {
                if (i >= currentSound.getLengthInFrames()) {
                    sound1.setSampleValueAt(i, sound1.getSampleValueAt(i));
                } else {
                    sound1.setSampleValueAt(i, sound1.getSampleValueAt(i) + currentSound.getSampleValueAt(i));
                }
            }
            currentSound = sound1;
        }
        int reply = JOptionPane.showConfirmDialog(null, "To blend another sound Press yes", null, JOptionPane.YES_NO_OPTION);
        if (reply == JOptionPane.YES_OPTION) {
            continue;
        } else {
            break;
        }
    }
}
```

34- Apply simple average, weighted average, max, min and median filters on sounds.



** For listening the sound go to GitHub

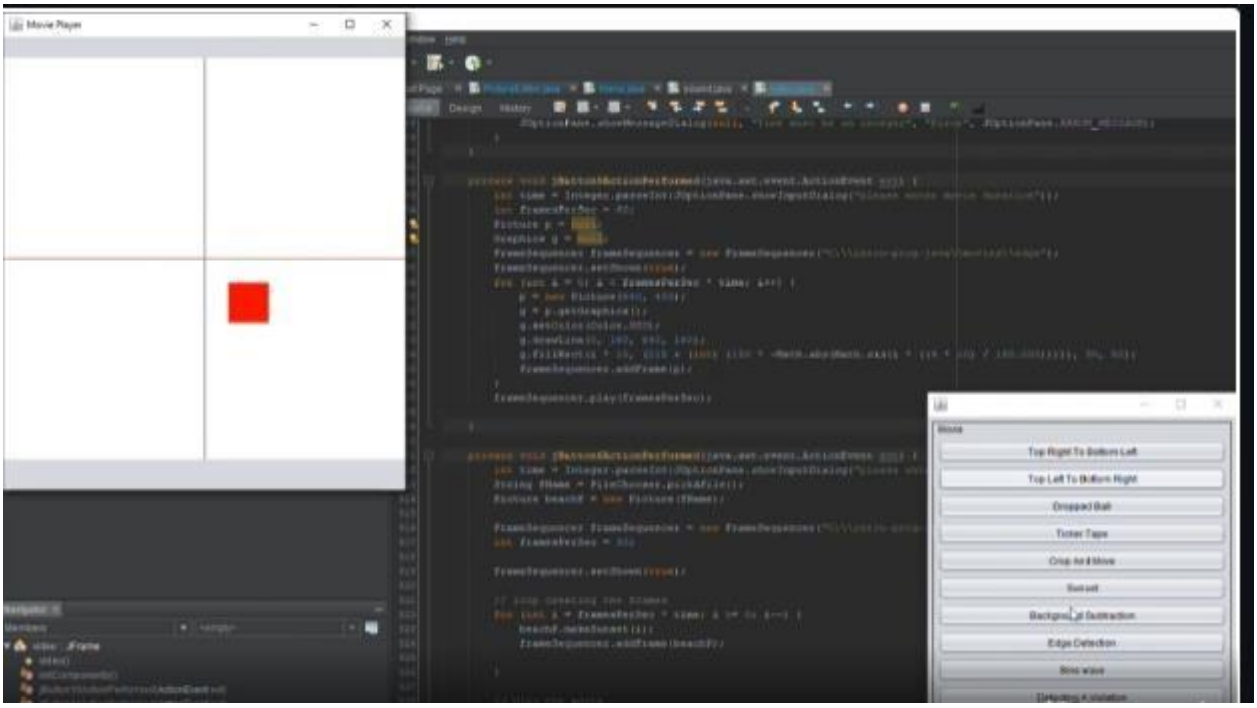
Users can modify the selected sound by applying different filters. To apply the filter they want, they must select it from the combo box and press the "Apply filter" button. To hear the changed sound, they must then press the "Play" button.

SoundFilters code:

```java
private void jButton12ActionPerformed(java.awt.event.ActionEvent evt) {

    SoundSample[] sampleArray = currentSound.getSamples();
    //simAvg
    if (jComboBox1.getSelectedIndex() == 0) {
        int startIndex = 0;
        int endIndex = 3;

        while (endIndex <= sampleArray.length) {
            int sum = 0;
            for (int i = startIndex; i < endIndex; i++) {
                sum += sampleArray[i].getValue();
            }

            int average = sum / 3;
            for (int i = startIndex; i < endIndex; i++) {
                sampleArray[i].setValue(average);
            }
            startIndex += 3;
            endIndex += 3;
        }

        for (int i = 0; i < sampleArray.length; i++) {
            System.out.println(sampleArray[i].getValue());
        }
        ////
    } else if (jComboBox1.getSelectedIndex() == 1) {
        //wightAvg
        int[] weighted = new int[]{2, 3, 2};
        int startIndex = 0;
        int endIndex = 3;

        while (endIndex <= sampleArray.length) {
            int sum = 0;
            int tmpIndex = 0;
            for (int i = startIndex; i < endIndex; i++) {
                sum += (sampleArray[i].getValue() * weighted[tmpIndex]);
                tmpIndex++;
            }

            int average = sum / 3;
            for (int i = startIndex; i < endIndex; i++) {
                sampleArray[i].setValue(average);
            }
            startIndex += 3;
            endIndex += 3;
        }

        for (int i = 0; i < sampleArray.length; i++) {
            System.out.println(sampleArray[i].getValue());
        }

    } else if (jComboBox1.getSelectedIndex() == 2) {
//max
        int startIndex = 0;
        int endIndex = 3;

        while (endIndex <= sampleArray.length) {
            int max = Integer.MIN_VALUE;

            for (int i = startIndex; i < endIndex; i++) {
                int tmp = sampleArray[i].getValue();
                if (tmp > max) {
                    max = tmp;
                }
            }

            for (int i = startIndex; i < endIndex; i++) {
                sampleArray[i].setValue(max);
            }
            startIndex += 3;
            endIndex += 3;
        }
```

```java
        for (int i = 0; i < sampleArray.length; i++) {
            System.out.println(sampleArray[i].getValue());
        }

    } else if (jComboBox1.getSelectedIndex() == 3) {
//min
        int startIndex = 0;
        int endIndex = 3;

        while (endIndex <= sampleArray.length) {
            int min = Integer.MAX_VALUE;

            for (int i = startIndex; i < endIndex; i++) {
                int tmp = sampleArray[i].getValue();
                if (tmp < min) {
                    min = tmp;
                }
            }

            for (int i = startIndex; i < endIndex; i++) {
                sampleArray[i].setValue(min);
            }
            startIndex += 3;
            endIndex += 3;
        }
        for (int i = 0; i < sampleArray.length; i++) {
            System.out.println(sampleArray[i].getValue());
        }

    } else if (jComboBox1.getSelectedIndex() == 4) {
//midin
        int[] medianArray = new int[3];
        int startIndex = 0;
        int endIndex = 3;
        while (endIndex <= sampleArray.length) {
            int tmpIndx = 0;

            for (int i = startIndex; i < endIndex; i++) {
                medianArray[tmpIndx] = sampleArray[i].getValue();
                tmpIndx++;
            }
            Arrays.sort(medianArray);
            for (int i = startIndex; i < endIndex; i++) {
                sampleArray[i].setValue(medianArray[1]);
            }
            startIndex += 3;
            endIndex += 3;
        }
        for (int i = 0; i < sampleArray.length; i++) {
            System.out.println(sampleArray[i].getValue());
        }
    }


}
```

36- Create Rectangle Movie method, Show it is moving from top right corner to bottom left, curved trajectory:



** For watching the video go to GitHub

Press the "Top Right To Bottom Left" button to watch the RecatangleMovie move from the top right corner to the bottom left corner. If you'd like to watch the rectangle move from the top left corner to the bottom right corner, press the "Top Left To Bottom Right" button. After pressing these buttons it will ask the user to enter the movie duration in seconds to show the seconds that he inserted.

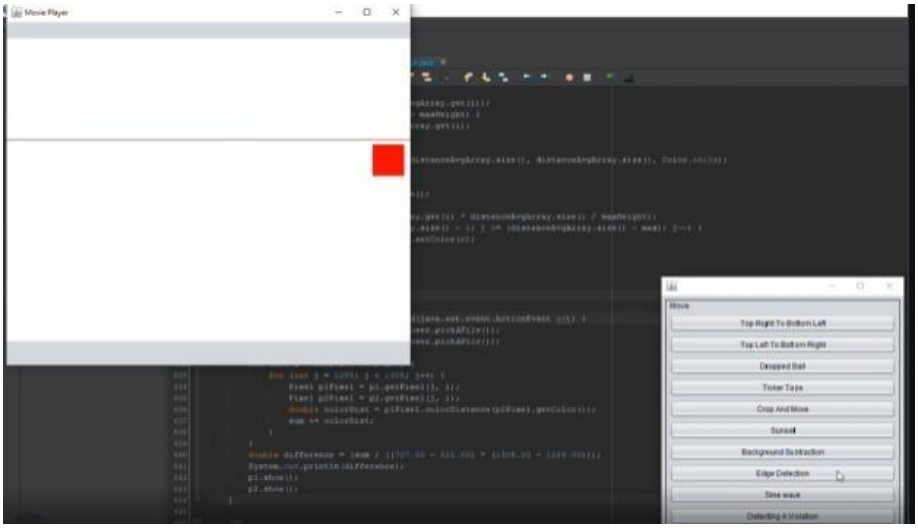
Top right to bottom left code:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    int Sec = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration in seconds"));
    int framesPerSec = 30;
    Picture p = null;
    Graphics g = null;
    FrameSequencer frameSequencer = new FrameSequencer("re");
    frameSequencer.setShown(true);
    int x = 450;
    int y = 0;
    for (int i = 0; i < framesPerSec * Sec; i++) {
        System.out.println(x + " " + y);
        x = x - 10;
        y = y + 10;

        if (x == -10 && y == 460) {
            break;
        }
        p = new Picture(500, 500);
        g = p.getGraphics();
        g.setColor(Color.RED);
        g.drawLine(0, 250, 500, 250);
        g.drawLine(250, 0, 250, 500);
        // x y w h
        g.fillRect(x, y, 50, 50);
        g.setColor(Color.CYAN);
        frameSequencer.addFrame(p);

    }
}
```

Top Left to bottom Right code:

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    int Sec = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration in seconds"));
    int framesPerSec = 30;
    Picture p = null;
    Graphics g = null;
    FrameSequencer frameSequencer = new FrameSequencer("re");
    frameSequencer.setShown(true);
    int x = 0;
    int y = 0;
    for (int i = 0; i < framesPerSec * Sec; i++) {
        System.out.println(x + " " + y);
        x = x + 10;
        y = y + 10;
        if (x == 460 && y == 460) {
            break;
        }
        // draw a filled rectangle
        p = new Picture(500, 500);
        g = p.getGraphics();
        g.setColor(Color.RED);
        g.drawLine(0, 250, 500, 250);
        g.drawLine(250, 0, 250, 500);

        // x y w h
        g.fillRect(x, y, 50, 50);
        g.setColor(Color.CYAN);
        frameSequencer.addFrame(p);

    }
}
```

37- Create Rectangle Movie method, Show the trajectory of motion followed by a Sine function:



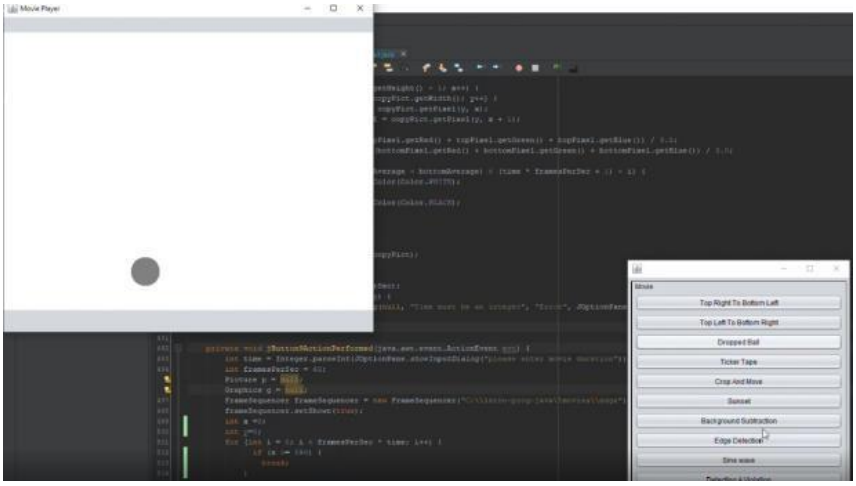
** For watching the video go to GitHub

By pressing the "Sine wave" button, the user can insert the desired number of seconds for the video to continue. This will cause a rectangular shape to move in a sine wave motion.

Sine wave code:

```
private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {
    int time = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration"));
    int framesPerSec = 60;
    Picture p = null;
    Graphics g = null;
    FrameSequencer frameSequencer = new FrameSequencer("C:\\intro-prog-java\\movies\\edge");
    frameSequencer.setShown(true);
    for (int i = 0; i < framesPerSec * time; i++) {
        p = new Picture(640, 480);
        g = p.getGraphics();
        g.setColor(Color.RED);
        g.drawLine(0, 160, 640, 160);
        g.fillRect(i * 10, (215 + (int) (150 * -Math.abs(Math.sin(i * ((4 * 10) / 180.000))))), 50, 50);
        frameSequencer.addFrame(p);
    }
    frameSequencer.play(framesPerSec);

}
```

38- Create Rectangle Movie method Show the trajectory of motion, when a ball is dropped on the floor:



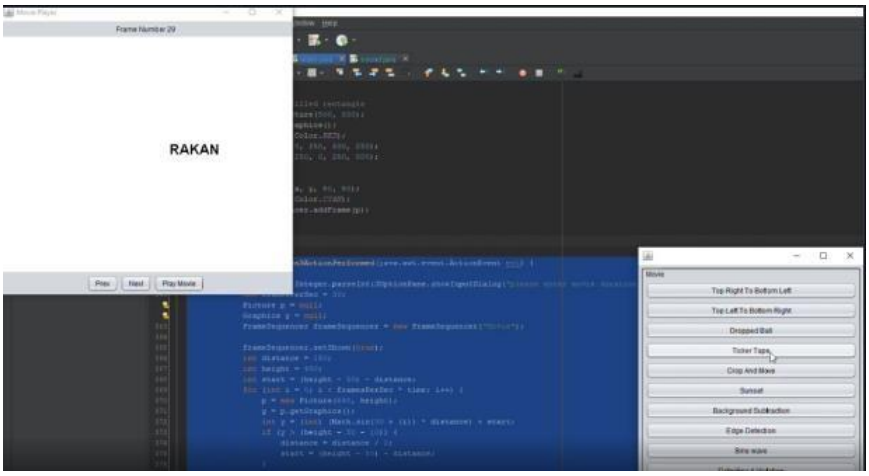
** For watching the video go to GitHub

By pressing the "Dropped Ball" button, the user can insert the desired number of seconds for the video to continue. This will cause a circle to move in a drop motion.

Dropp ball code:

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int time = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration in seconds"));
        int framesPerSec = 30;
        Picture p = null;
        Graphics g = null;
        FrameSequencer frameSequencer = new FrameSequencer("Movie");

        frameSequencer.setShown(true);
        int distance = 150;
        int height = 480;
        int start = (height - 50) - distance;
        for (int i = 0; i < framesPerSec * time; i++) {
            p = new Picture(640, height);
            g = p.getGraphics();
            int y = (int) (Math.sin(30 + (i)) * distance) + start;
            if (y > (height - 50 - 10)) {
                distance = distance / 2;
                start = (height - 50) - distance;
            }
            y = (int) (Math.sin(30 + (i)) * distance) + start;
            g.setColor(Color.GRAY);
            g.fillOval(i * 20, y, 50, 50);
            // add frame to sequencer
            frameSequencer.addFrame(p);
        }
        // play the movie
        frameSequencer.play(framesPerSec);
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(null, "Time must be an integer", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

39- CreateTickerTapeMovie – A string of text moves back and forth and does not disappear from screen.



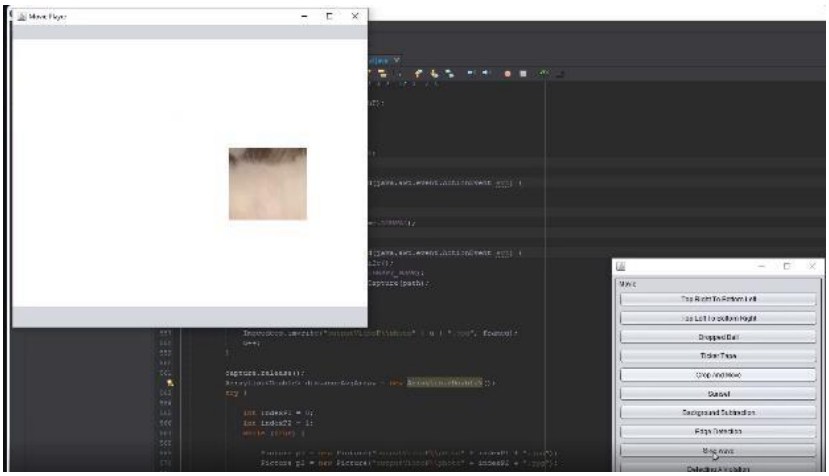
** For watching the video go to GitHub

By pressing the "Tricker Tape" button, the user can enter the number of seconds they wish for the video to run and the text they want to move. This will cause the selected text to move along the screen according to the number of seconds specified

TickerTapeMovie code:

```
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int time = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration"));
        String message = JOptionPane.showInputDialog("please enter your message");
        int framesPerSec = 30;
        Picture p = null;
        Graphics g = null;
        FrameSequencer frameSequencer = new FrameSequencer("Movie");
        Font font = new Font("Arial", Font.BOLD, 24);
        boolean right = true;
        // loop for 2 seconds of animation
        for (int j = 0, k = 0; j < framesPerSec * time; j++) {
            // draw the string
            p = new Picture(500, 400);
            g = p.getGraphics();
            g.setColor(Color.BLACK);
            g.setFont(font);
            if (right) {
                if (k * 10 > p.getWidth() - message.length() * 12) {
                    right = false;
                }
            } else {
                if (k < 1) {
                    right = true;
                }
            }
            if (right) {
                g.drawString(message, k++ * 10, 200);
            } else {
                g.drawString(message, k-- * 10, 200);
            }
            // add frame to sequencer
            frameSequencer.addFrame(p);
        }

        // play the movie
        frameSequencer.play(framesPerSec);
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(null, "Time must be an integer", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

40- Create a movie, cropping a part of given image that appears randomly on screen:



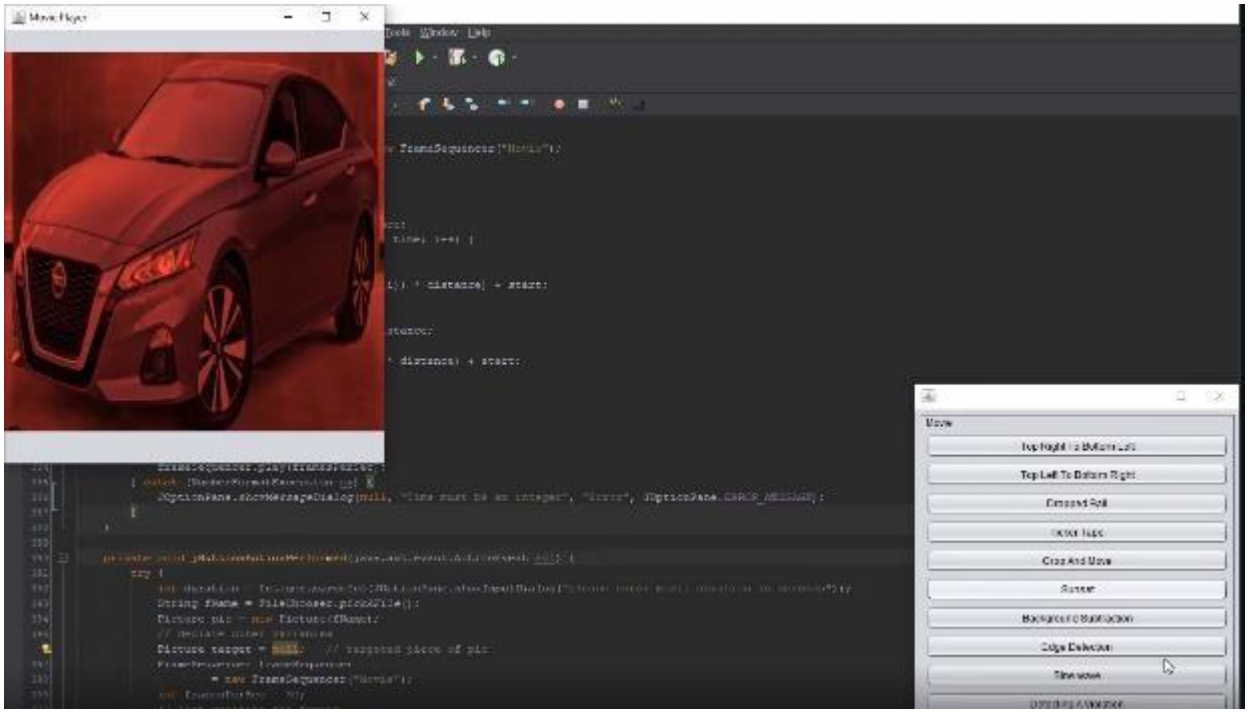
** For watching the video go to GitHub

In this method, the users must specify the duration of the video and the picture that they want to crop. Afterwards, the method will randomly select a portion of the image to be moved.

Cropping and move in a video code:

```
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int duration = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration in seconds"));
        String fName = FileChooser.pickAFile();
        Picture pic = new Picture(fName);
        // declare other variables
        Picture target = null;   // targeted piece of pic
        FrameSequencer frameSequencer
                = new FrameSequencer("Movie");
        int framesPerSec = 30;
        // loop creating the frames
        for (int i = 0; i < framesPerSec * duration; i++) {
            target = new Picture(640, 480);
            target.copy(pic, 250, 170, 390, 300, i * 10, i * 5);
            frameSequencer.addFrame(target);
        }
        // play the movie
        frameSequencer.play(framesPerSec);

    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(null, "Time must be an integer", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

41- Create Sunset movie:



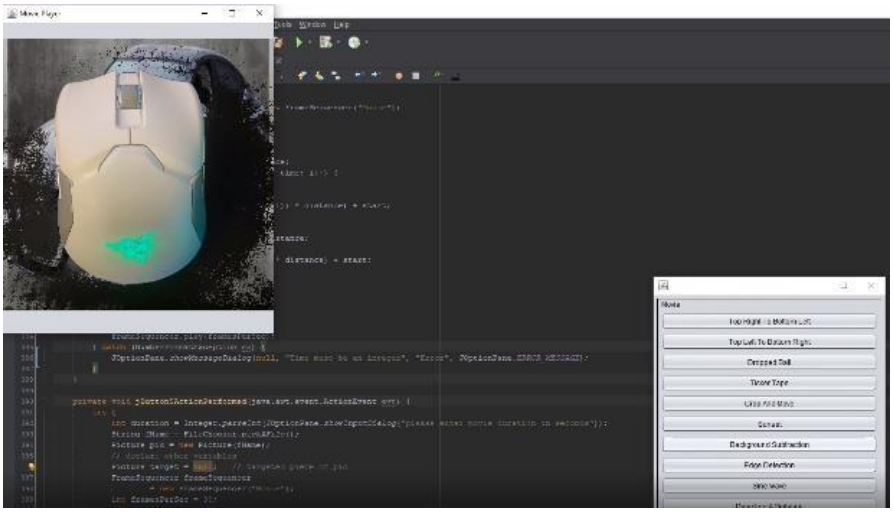
** For watching the video go to GitHub

By pressing the "Sunset" button, the user can enter the number of seconds they wish for the video to run and the the image they want to make sunset effect. This will cause sunset effect to the photo by time the user selected according to the number of seconds specified.

Create Sunset movie Code:

```
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
    int time = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration"));
    String fName = FileChooser.pickAFile();
    Picture beachP = new Picture(fName);

    FrameSequencer frameSequencer = new FrameSequencer("C:\\intro-prog-java\\movies\\edge");
    int framesPerSec = 30;

    frameSequencer.setShown(true);

    // loop creating the frames
    for (int i = framesPerSec * time; i >= 0; i--) {
        beachP.makeSunset(i);
        frameSequencer.addFrame(beachP);

    }

    // play the movie
    frameSequencer.play(framesPerSec);
}
```

42- Back Ground Subtraction Movie:



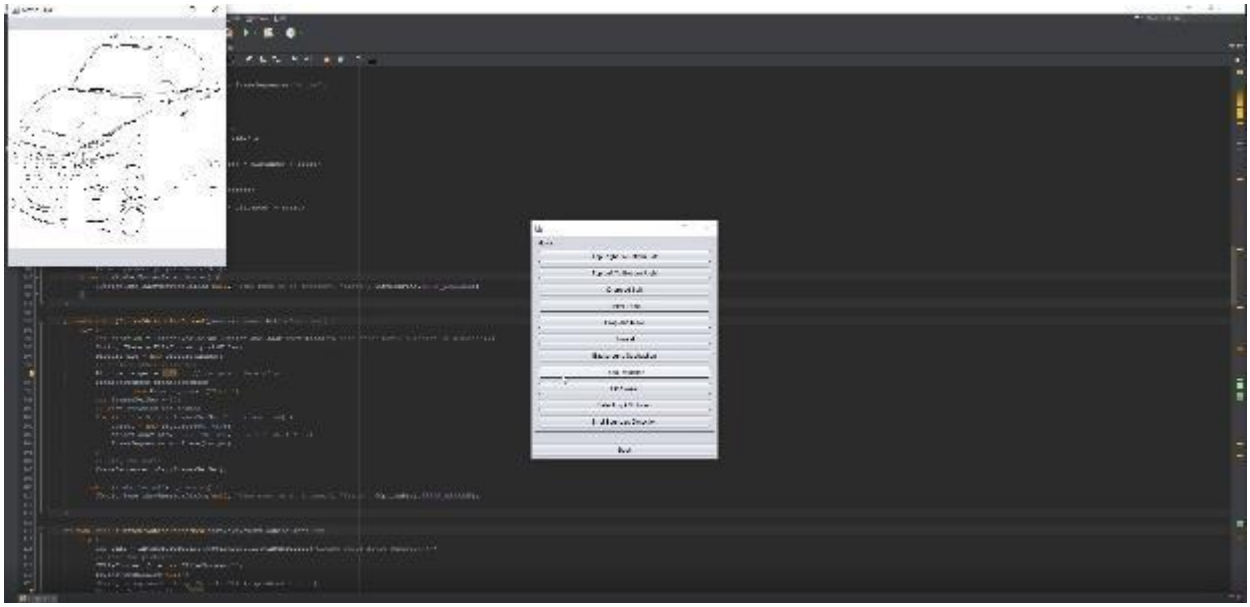
** For watching the video go to GitHub

This method allows the user to create a new video background by entering the duration of the video and three pictures. The first picture will be used as the new background, the second will be used as the foreground, and the third will be cut out and removed from the foreground. After these steps are completed, the first picture will be used as the new background of the video.

Back Ground Subtraction Movie Code:

```
private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int time = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration"));
        // load the pictures
        JFileChooser fc = new JFileChooser("");
        fc.showOpenDialog(null);
        String foreground = fc.getSelectedFile().getAbsolutePath();
        Picture foregroundPic = null;

        fc.showOpenDialog(null);
        String oldBG = fc.getSelectedFile().getAbsolutePath();
        Picture oldBGPic = new Picture(oldBG);

        fc.showOpenDialog(null);
        String newBG = fc.getSelectedFile().getAbsolutePath();
        Picture newBGPic = new Picture(newBG);

        // declare other variables
        FrameSequencer frameSequencer = new FrameSequencer("Movie_Edit");
        int framesPerSec = 30;
        frameSequencer.setShown(true);
        // loop creating the frames
        for (int i = 0; i < framesPerSec * time; i++) {
            foregroundPic = new Picture(foreground);
            foregroundPic.swapBackground(oldBGPic, newBGPic, i);
            frameSequencer.addFrame(foregroundPic);
        }

        // play the movie
        frameSequencer.play(framesPerSec);
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(null, "Time must be an integer", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

43- Edge Detection Movie:



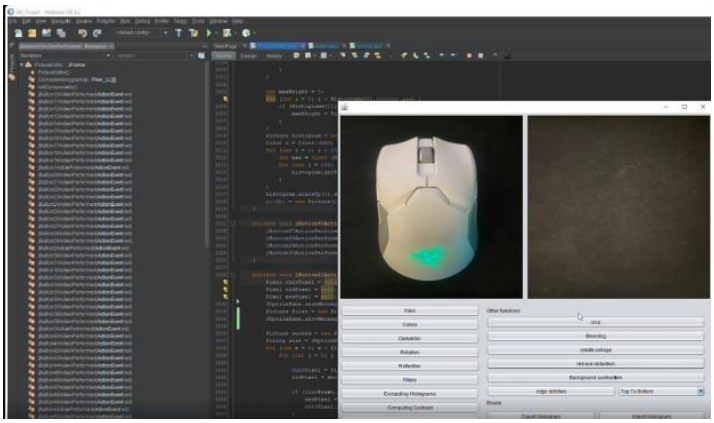
** For watching the video go to GitHub

In this method, the user needs to specify the duration of the video and upload an image to detect the edges. The method will then output a video displaying the detected edges.

Edge Detection Movie Code:

```
private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int time = Integer.parseInt(JOptionPane.showInputDialog("please enter movie duration"));
        JFileChooser fc = new JFileChooser("");
        fc.showOpenDialog(null);
        String fName = fc.getSelectedFile().getAbsolutePath();
        Picture pic = new Picture(fName);

        Picture copyPict = null;

        FrameSequencer frameSequencer = new FrameSequencer("movie_Edit");
        int framesPerSec = 30;

        for (int i = 0; i < framesPerSec * time; i++) {
            copyPict = new Picture(pic);
            double topAverage = 0.0;
            double bottomAverage = 0.0;

            for (int x = 0; x < pic.getHeight() - 1; x++) {
                for (int y = 0; y < copyPict.getWidth(); y++) {
                    Pixel topPixel = copyPict.getPixel(y, x);
                    Pixel bottomPixel = copyPict.getPixel(y, x + 1);

                    topAverage = (topPixel.getRed() + topPixel.getGreen() + topPixel.getBlue()) / 3.0;
                    bottomAverage = (bottomPixel.getRed() + bottomPixel.getGreen() + bottomPixel.getBlue()) / 3.0;

                    if (Math.abs(topAverage - bottomAverage) < (time * framesPerSec + 1) - i) {
                        topPixel.setColor(Color.WHITE);
                    } else {
                        topPixel.setColor(Color.BLACK);
                    }
                }
            }

            frameSequencer.addFrame(copyPict);
        }

        frameSequencer.play(framesPerSec);
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(null, "Time must be an integer", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

44- Compare two Images:

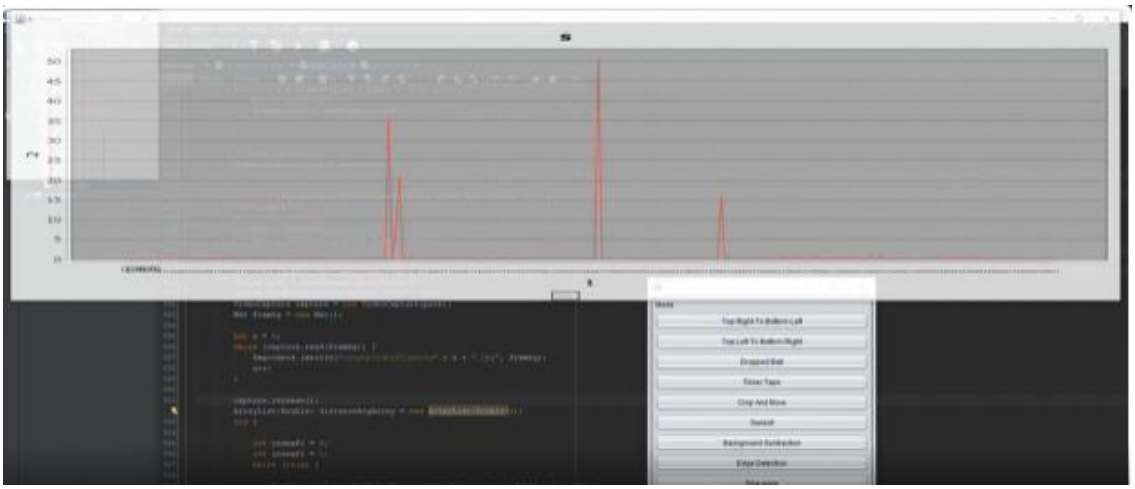


In this method, the user will insert two pictures which will then be compared to determine how much difference there is between them. If the difference is less than 50, it indicates a small difference between the two images. If the difference is greater than 50, it indicates a significant difference.

Compare two Images code:

```
private void jButton55ActionPerformed(java.awt.event.ActionEvent evt) {

    try {
        Picture p1 = picObj;
        Picture p2 = new Picture(FileChooser.pickAFile());
        Image img = (p2.getImage()).getScaledInstance(jLabel1.getWidth(), jLabel1.getHeight(), Image.SCALE_SMOOTH);
        BufferedImage bimage = new BufferedImage(img.getWidth(null), img.getHeight(null), BufferedImage.TYPE_INT_ARGB);
        Graphics2D bGr = bimage.createGraphics();
        bGr.drawImage(img, 0, 0, null);
        bGr.dispose();
        File outputfile = new File("Tmp\\p2.png");
        ImageIO.write(bimage, "png", outputfile);
        String pathNameP2 = "Tmp\\p2.png";
        p2 = new Picture(pathNameP2);
        img = (p2.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
        icon = new ImageIcon(img);
        jLabel2.setIcon(icon);

        double distanceAvg = 0;
        Pixel p1Pixel = null;
        Pixel p2Pixel = null;

        for (int x = 0; x < p1.getWidth(); x++) {
            for (int y = 0; y < p2.getHeight(); y++) {
                p1Pixel = p1.getPixel(x, y);
                p2Pixel = p2.getPixel(x, y);

                double distance = p1Pixel.colorDistance(p2Pixel.getColor());
                distanceAvg += distance;
            }
        }
        JOptionPane.showMessageDialog(null, "the difference between is" + (distanceAvg / p2.getPixels().length), "File",
                JOptionPane.OK_OPTION);
    } catch (IOException ex) {
        Logger.getLogger(PictureEditor.class.getName()).log(Level.SEVERE, null, ex);
    }
```

45- Shot boundary detection:



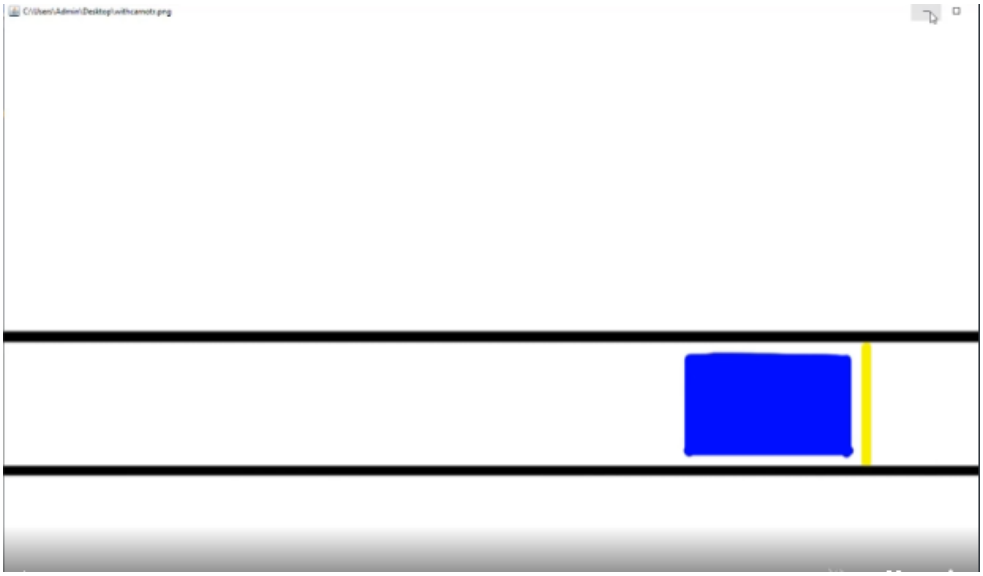
** For watching the video go to GitHub

In this method, the user can input a video and view a chart that shows the differences between the frames of that video. If the video has significant differences between the frames, the chart will show a high value, whereas if there are not many differences, the chart will remain low.

Shot boundary detection code:

```
private void jButton11ActionPerformed(java.awt.event.ActionEvent evt) {
    String path = FileChooser.pickAFile();
    System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
    VideoCapture capture = new VideoCapture(path);
    Mat frameg = new Mat();

    int u = 0;
    while (capture.read(frameg)) {
        Imgcodecs.imwrite("outputVidtoP\\photo" + u + ".jpg", frameg);
        u++;
    }

    capture.release();
    ArrayList<Double> distanceAvgArray = new ArrayList<Double>();
    try {

        int indexP1 = 0;
        int indexP2 = 1;
        while (true) {

            Picture p1 = new Picture("outputVidtoP\\photo" + indexP1 + ".jpg");
            Picture p2 = new Picture("outputVidtoP\\photo" + indexP2 + ".jpg");

            double distanceAvg = 0;
            Pixel p1Pixel = null;
            Pixel p2Pixel = null;

            for (int x = 0; x < p1.getWidth(); x++) {
                for (int y = 0; y < p2.getHeight(); y++) {
                    p1Pixel = p1.getPixel(x, y);
                    p2Pixel = p2.getPixel(x, y);

                    double distance = p1Pixel.colorDistance(p2Pixel.getColor());
                    distanceAvg += distance;
                }
            }
            distanceAvgArray.add(((distanceAvg / p1.getPixels().length)));
            indexP1++;
            indexP2++;
        }
    } catch (Exception e) {

    }

    DefaultCategoryDataset dataset = new DefaultCategoryDataset();
    for (int i = 0; i < distanceAvgArray.size(); i++) {
        dataset.setValue(distanceAvgArray.get(i), "", String.valueOf(i));
    }
    JFreeChart chart = ChartFactory.createLineChart("s", "1", "2", dataset);

    CategoryPlot catPlot = chart.getCategoryPlot();
    catPlot.setRangeGridlinePaint(Color.BLACK);
    ChartFrame frame = new ChartFrame("b", chart);
    frame.setSize(1900, 500);
    frame.setVisible(true);

    double maxHeight = 0.0;
    for (int i = 0;
            i < distanceAvgArray.size();
            i++) {
        System.out.println(distanceAvgArray.get(i));
        if (distanceAvgArray.get(i) > maxHeight) {
            maxHeight = distanceAvgArray.get(i);
        }
    }
    Picture histogram = new Picture(distanceAvgArray.size(), distanceAvgArray.size(), Color.white);
    Color c = Color.RED;
    for (int i = 0;
            i < distanceAvgArray.size();
            i++) {
        double max = (distanceAvgArray.get(i) * distanceAvgArray.size() / maxHeight);
        for (int j = distanceAvgArray.size() - 1; j >= (distanceAvgArray.size() - max); j--) {
            histogram.getPixel(i, j).setColor(c);
        }
    }

    histogram.show();
}
```

47- Detecting a violation of traffic rule:



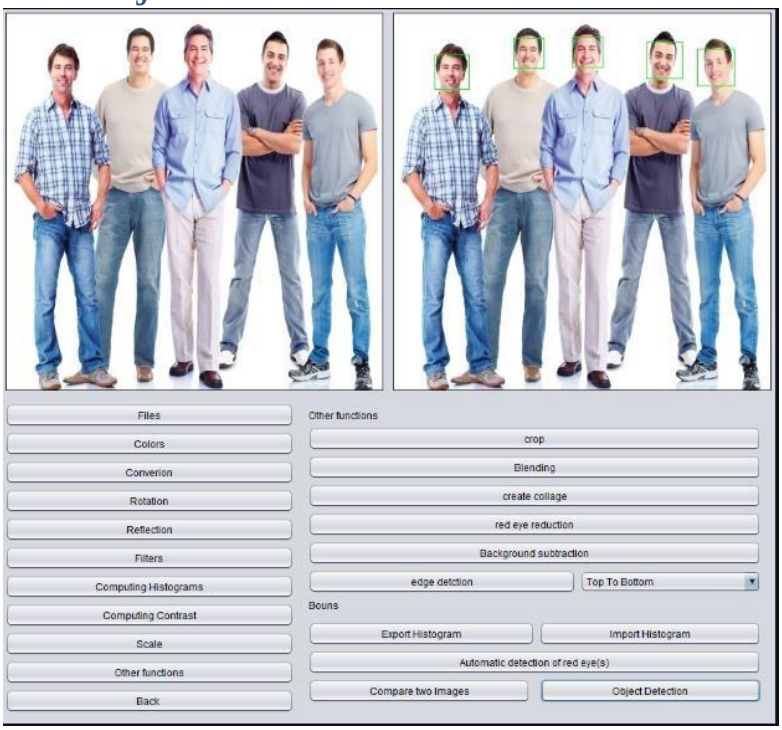
** For watching the video go to GitHub

This method allows users to upload two photos to check for any traffic rule violations. If a violation is detected, such as a vehicle passing or touching the yellow line, the user will be notified of the violation. If no violation is found, the user will be informed that there is no violation.

Detecting a violation of traffic rule code

```
private void jButton10ActionPerformed(java.awt.event.ActionEvent evt) {
    Picture p1 = new Picture(FileChooser.pickAFile());
    Picture p2 = new Picture(FileChooser.pickAFile());
    double sum = 0;
    for (int i = 521; i < 707; i++) {
        for (int j = 1288; j < 1305; j++) {
            Pixel p1Pixel = p1.getPixel(j, i);
            Pixel p2Pixel = p2.getPixel(j, i);
            double colorDist = p1Pixel.colorDistance(p2Pixel.getColor());
            sum += colorDist;
        }
    }
    double difference = (sum / ((707.00 - 521.00) * (1305.00 - 1288.00)));
    if (difference > 15) {
        JOptionPane.showMessageDialog(null, "there are traffic violation");
    } else {
        JOptionPane.showMessageDialog(null, "there are no traffic violation");
    }
    System.out.println(difference);
    p1.show();
    p2.show();

}
```

48- Object Detection:



** For watching the video go to GitHub

This method can be used to detect objects, such as heads, and draw a rectangle around them. To increase accuracy, it needs to be trained further to recognize more objects.

Object detection code:

```
private void jButton56ActionPerformed(java.awt.event.ActionEvent evt) {
    // loading the OpenCV core library
    System.loadLibrary(Core.NATIVE_LIBRARY_NAME);

    // loading the image
    String imagePath = pathName;
    Mat newImage = Imgcodecs.imread(imagePath);

    // loading the classifier
    String classifierPath = "opencv\\sources\\data\\lbpcascades\\lbpcascade_frontalface.xml";
    CascadeClassifier classifier = new CascadeClassifier(classifierPath);

    // detect the objects
    MatOfRect objects = new MatOfRect();
    classifier.detectMultiScale(newImage, objects);

    // draw the rectangle
    Rect[] objectsArray = objects.toArray();
    for (int i = 0; i < objectsArray.length; i++) {
        Imgproc.rectangle(newImage, new Point(objectsArray[i].x, objectsArray[i].y),
                new Point(objectsArray[i].x + objectsArray[i].width, objectsArray[i].y + objectsArray[i].height),
                new Scalar(0, 255, 0));
    }

    // save the image
    String outputPath = "Tmp\\ODF.jpg";
    Imgcodecs.imwrite(outputPath, newImage);
    Picture first = new Picture("Tmp\\ODF.jpg");
    Image img = (first.getImage()).getScaledInstance(jLabel2.getWidth(), jLabel2.getHeight(), Image.SCALE_SMOOTH);
    icon = new ImageIcon(img);
    jLabel2.setIcon(icon);

}
```

Problems that we faced while doing the project:

One of the most difficult challenges we faced during this project was dealing with the size of pictures that the user would insert. Many methods did not accept differences in size, resulting in an array out of bounds error. Additionally, uploading and pushing the project onto GitHub was a significant problem because of its file size restrictions. We also encountered difficulties with OpenCV, both in terms of implementation and usage, as it did not support our Netbeans version. Additionally, a large amount of effort was put into making the detailed documentation for this project. Lastly, we faced a lack of codes and information, such as shot boundary, which was necessary for the project's completion.

References:

[1] B. Ericson, "Picture class," in IntroCS, Princeton University, https://introcs.cs.princeton.edu/java/stdlib/javadoc/Picture.html, Accessed: Month Day, Year.

[2] B. Ericson, "SimplePicture class," in UIC, University of Illinois at Chicago, https://www2.cs.uic.edu/~i101/doc/SimplePicture.html, Accessed: Month Day, Year.

[3] "RGB to HSV color conversion," RapidTables, https://www.rapidtables.com/convert/color/rgb-to-hsv.html, Accessed: Month Day, Year.

[4] OpenCV Development Team, "OpenCV library," in OpenCV, Accessed: Month Day, Year.

[5] B. Ericson, "Sound class," in Duke University, https://www2.cs.duke.edu/courses/spring07/cps001/doc/mediaClasses/Sound.html, Accessed: Month Day, Year.

[6] B. Ericson, "MovieMaker class," in Georgia Institute of Technology, Accessed: Month Day, Year.

[7] B. Ericson, "FrameSequencer class," in UIC, University of Illinois at Chicago, https://www2.cs.uic.edu/~i101/doc/FrameSequencer.html, Accessed: Month Day, Year.