IMAGE PROCESSING PRACTICUM

BBM 413 & BBM 415

FUNDAMENTALS OF IMAGE PROCESSING

FUNDAMENTALS OF IMAGE PROCESSING LAB

HACETTEPE UNIVERSITY

Mücahit Veli CUMART, 21605893

Yunus Emre AKGÜN,   21726875

Introduction

The aim of this project is to consolidate the knowledge we have learned in the classes by putting it into practice. We are expected to implement some of image processing (or image enhancements) methods and build simple GUI for these methods. There are totally 17 methods which can be used by a user in our project. This is a simple GUI program for these requirements.

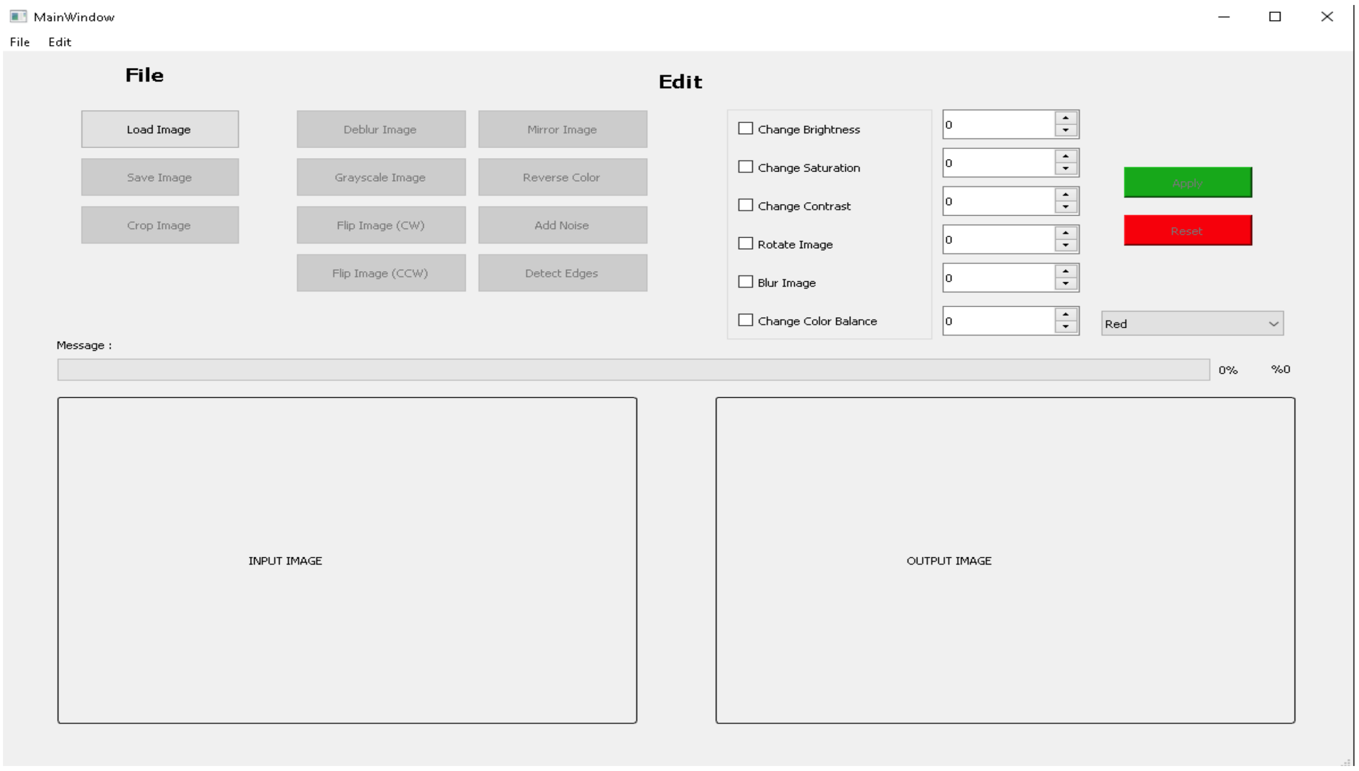
Technical Information

In this project, Python is used for implementation. Some libraries which are for image processing and image enhancement are used in this project. These libraries are Opencv, Numpy, Random, Pillow, Imutils, Scikit-image. Some of the methods in these libraries are used in this project when it is necessary and where they give better results. For implementation of the GUI, the library,PyQt5, is used. Pyqt5 has some advantages like coding flexibility, various UI components.

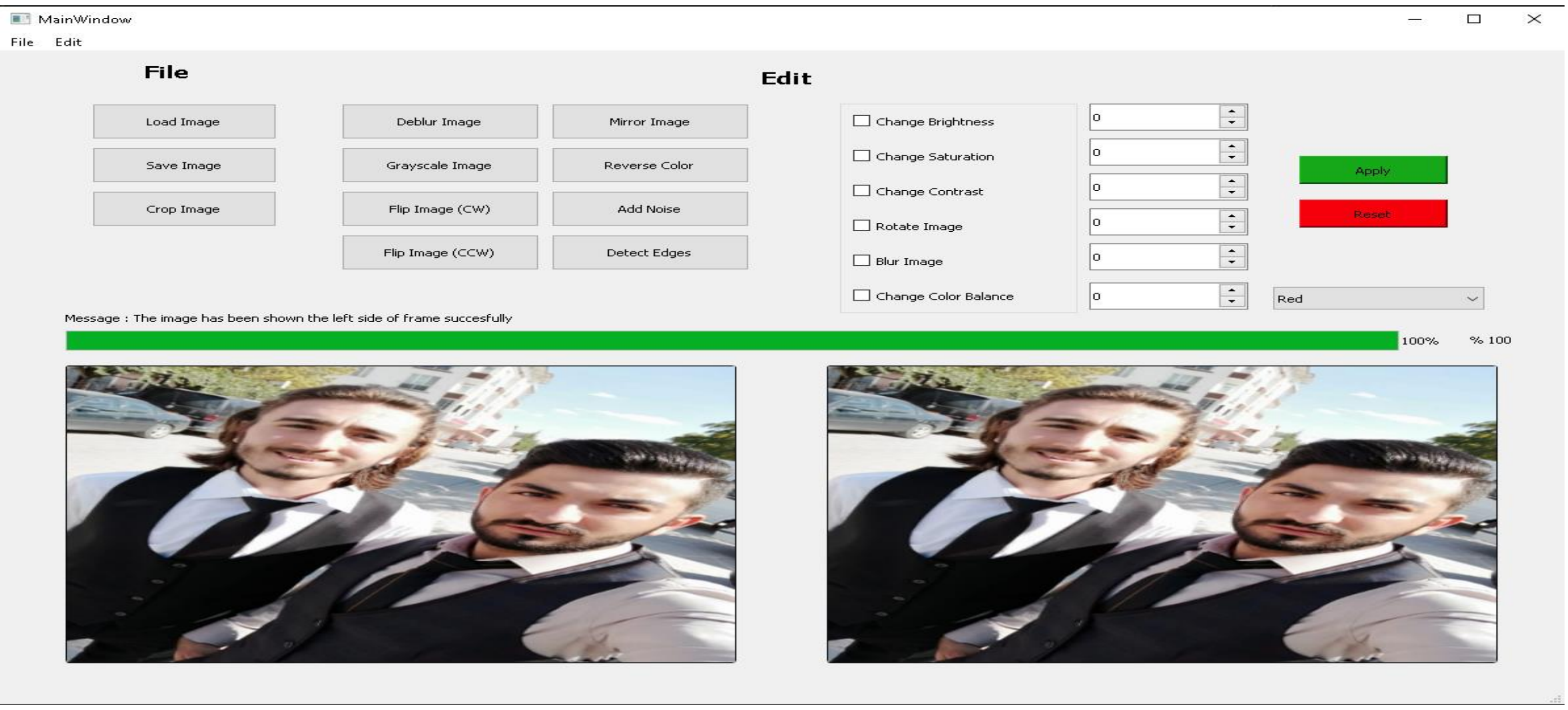
Usage Information

This project has simple features to use as a user. There are buttons, checkboxes, and input fields where necessary. Example usage of this project with screenshots step by step is below:

1) User runs the program

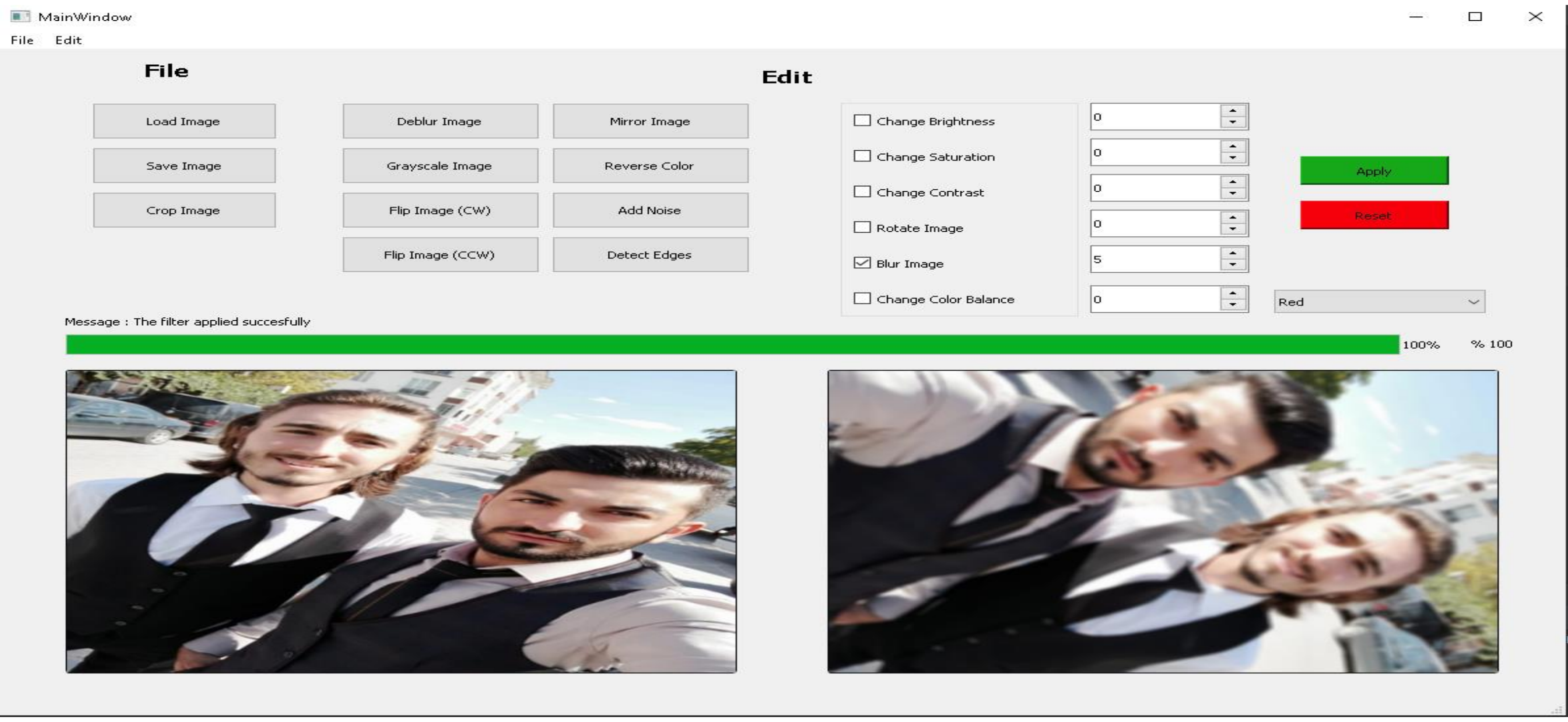


2) When the program runs , Buttons other than the “Load Image” are disabled because there is no image which is loaded to the program. If user pushes to the “Load Image”, he/she can select an image from local storage. After this selection, the images are shown in the frames and all buttons are enabled.



Message : The image has been shown the left side of frame succesfully

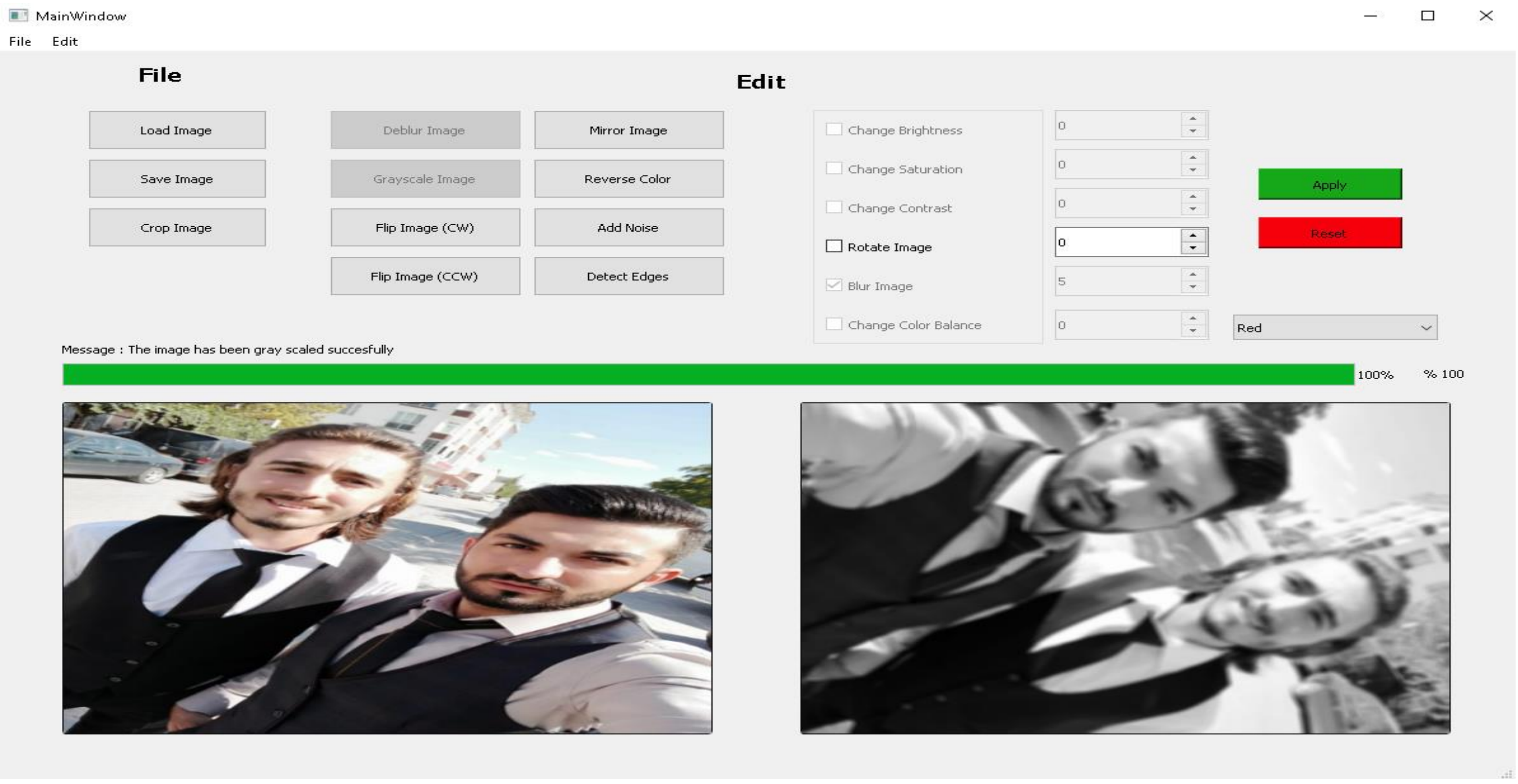
3) After loading image, the user can use all features for manipulating the image. For example, we can assume that the user used “Mirror Image”,then “Flip Image (CW)”, finally “Blur Image(with 5 as an input)” . The output is below:



Blur Image [✓] 5

Message : The filter applied succesfully

4) If the user grayscale the image, some features is disabled for that moment. For example "Change Color Balance" can't be done on a grayscale image. The output when the image grayscaled is blow:

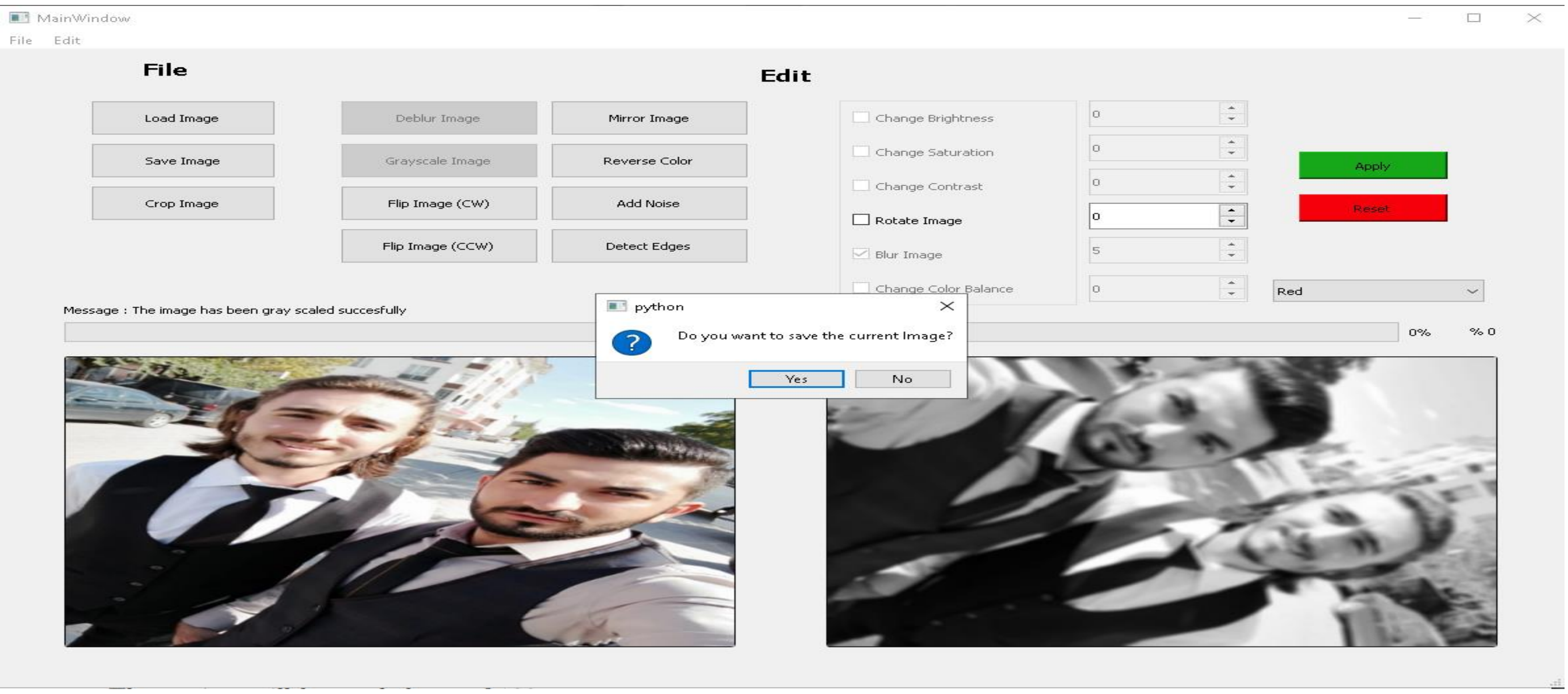


Change Brightness 0
Change Saturation 0
Change Contrast 0
Rotate Image 0
✓ Blur Image 5
Change Color Balance 0 Red

Message : The image has been gray scaled succesfully

100% % 100

5) If the user wants to load an image again, the program alerts an message for saving current manipulated image. If the user want to save this image, he/she must click "yes" and select the path for saving image. After saving, he/she can select new image for loading.



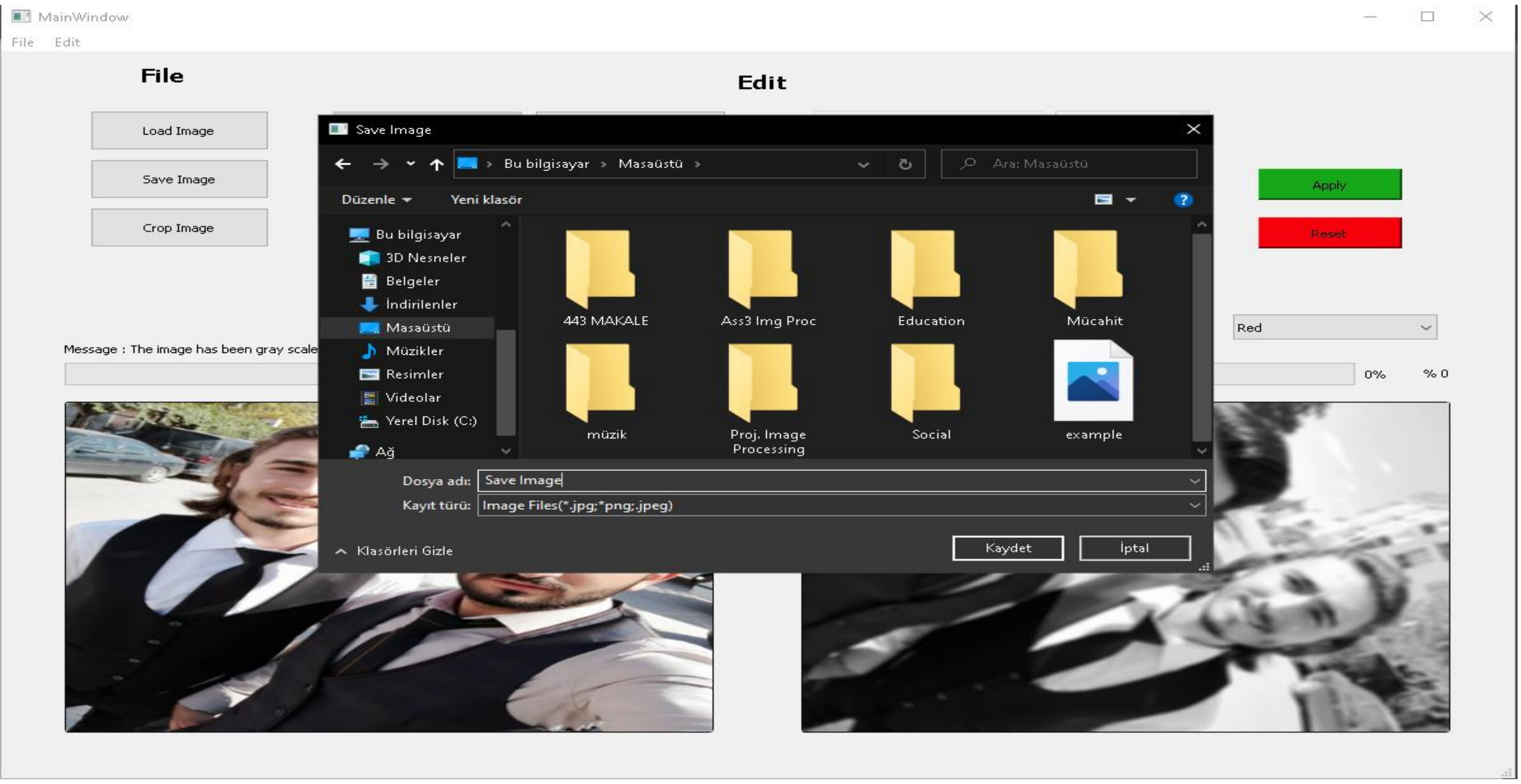
Change Brightness 0
Change Saturation 0
Change Contrast 0
Rotate Image 0
✓ Blur Image 5
Change Color Balance 0 Red

Message : The image has been gray scaled succesfully

0% % 0

Do you want to save the current Image?

6) If the user wants to save the image, he/she must push the "Save Image". After pushing this button, a window is opened for selecting saving path. Also, there is a shortcut for saving image. It is "CTRL + S ".

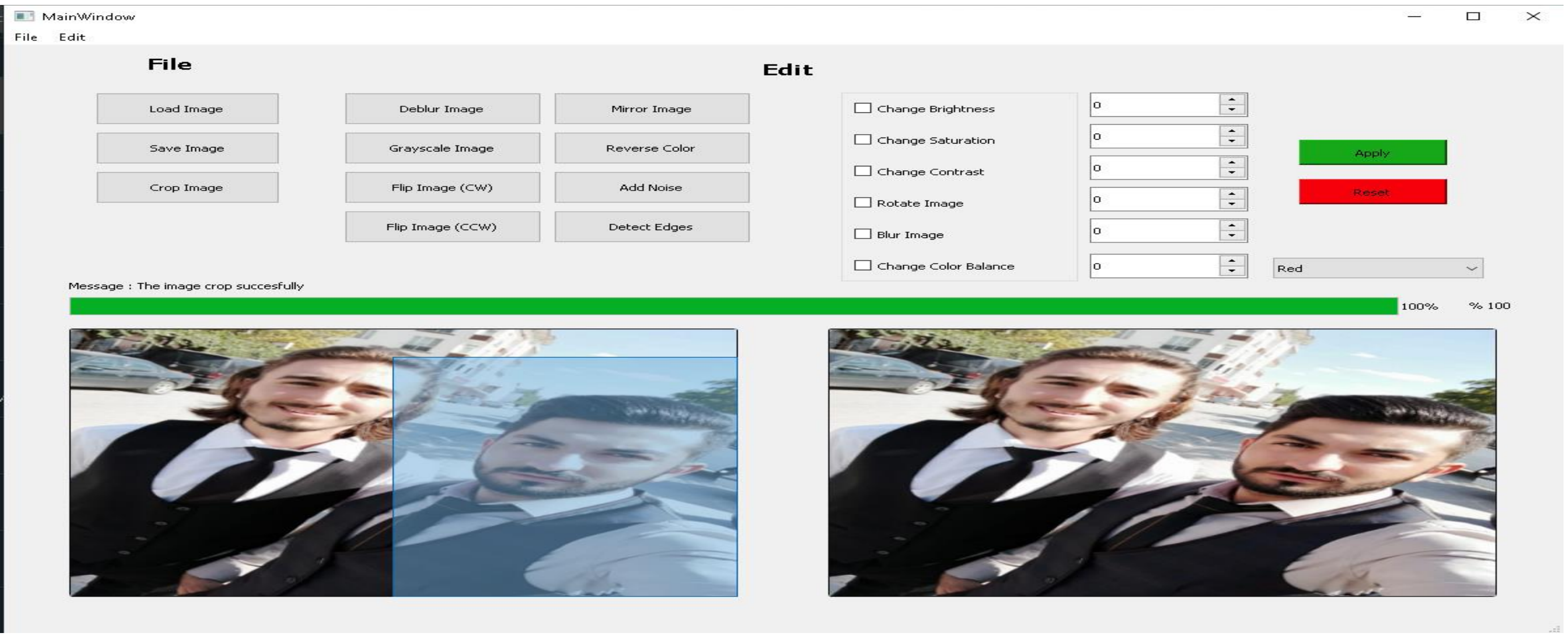


Message : The image has been gray scale

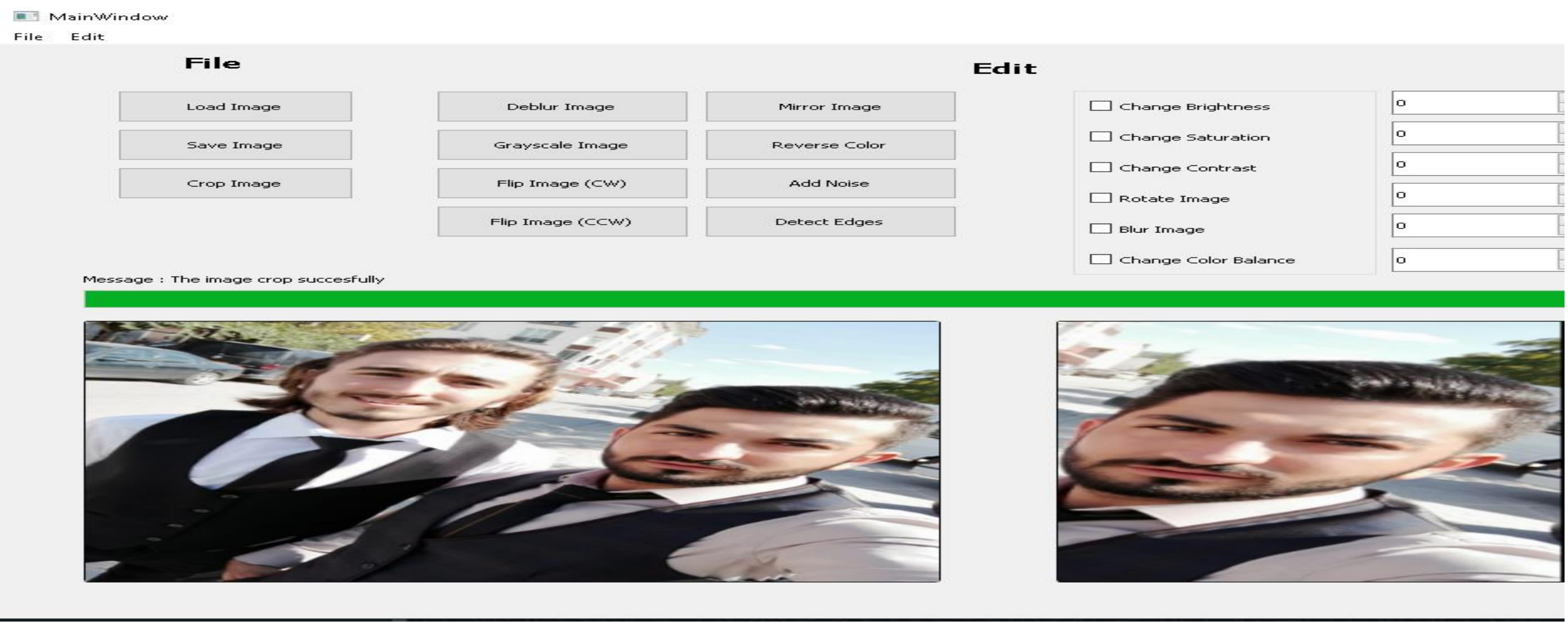
Dosya adı: Save Image

Kayıt türü: Image Files(*.jpg;*png;.jpeg)

7) If you want to crop image, Firstly, you should push the "Crop Image", Then, you should press the original image once.Then,you should move the mouse from down to up while you are clicking the mouse. When you release the clicking, you can see the cropped image in output frame.



Message : The image crop succesfully



Message : The image crop succesfully

8) If you want to reset all changes. You can press the “Reset” button. After you pressed this button, all changes will be undone.

Method Informations

There are two “.py” files in this project. Main.py is for manipulation calculations(Backend), Gui.py is for GUI.

loadImage(path): This function gets a parameter,path, for selected path of original image. In this method, cv2.imdecode() function from Opencv is used. After opening the original image,returns opened image. In the GUI, Load Image button connected to function, and after necessary checks , the “loadImage(path)” method is called.

resizeImage(image, x, y): This function gets three parameters,image, x(the height of image) , y(the width of image). In this method, cv.resize() function from Opencv is used. Returns resized image. In the GUI, this function called to sync image with frames.

saveImage(image, path): This function gets two parameters. Image is the image which will be saved,and path is where the image will be saved. In this method, cv2.imencode() function from Opencv is used,and also cv2.tofile is used for saving image.

blurImage(image, val): This function gets two parameters. Image is the image which will be manipulated, and val is the blur level. In this method, cv2.blur() function from Opencv is used.Returns blurred image.

deblurImage(image,kernel_size=(5,5), sigma=1.0, amount = 1.0, thershold = 0): This function gets image parameter and also the other parameters which are static. In this method, cv2.GaussianBlur() function from Opencv is used. The maximum and minimum of sharpened image is calculated, and the values are rounded.Returns deblurred image(sharpened).

grayScaleImage(image): This function gets only image parameter which will be manipulated image. In this method, cv2.cvtColor(image,cv2.COLOR_BGR2GRAY) function from Opencv is used. Returns grayscaled image.

cropSelectedPartOfImage(image,startWidth,stopWidth,startHeight,stopHeight): This function gets image parameter and four location points. These points are selected part from image. With these points, the image is changed. Returns new image.

flipImageClockWise(image): This function gets image parameter. In this method, cv2.rotate() from Opencv function is used. The image is rotated clockwise 90 degree.Returns rotated image.

flipImageClockWise(image): This function gets image parameter. In this method, cv2.rotate() from Opencv function is used. The image is rotated counter clockwise 90 degree.Returns rotated image.

mirrorImage(image): This function gets image parameter. In this method, cv2.flip() from Opencv function is used. The y-symmetry of the image is taken.Returns new image.

rotateImage(image,degree): This function gets image and degree parameters. In this method, imutils.rotate() function from Imutils library is used. The image is rotated by the given degree.

reverseColor(image): This function gets image parameter. The values of pixels in this image are subtracted from 255. Returns reversed image.

changeColorBalance(image,balance,r_g_b): This function gets three parameters, image,balance, and r_g_b. method, np.zeros_like() function from Numpy library is used. Firstly, A numpy array is created with the same size as the given image with zeros. Then, the values in the array are changed with given balance and r_g_b parameter.If the r_g_b parameter is “RED”, the red values of color array are changed.Same for green and blue. The changes is protectted with 0 and 255. Finally, the color array is added to the picture with cv2.add() function from Opencv. Returns new image.

addjustBrightnessOfImage(image, brightness): This function gets two parameters, image and brightness value.In this method,cv2.addWeighted() function from Opencv is used. First, the alpha and gamma values are calculated according to the brightness value.Then, with the function cv2.addWeighted, the alpha and gamma values are added to the image. Returns new image.

adjustSaturationOfImage(image, saturation): This function gets two parameters, image and saturation value. In this method, cv2.cvtColor(cv2.COLOR_BGR2HLS) and cv2.cvtColor(cv2.COLOR_HLS2BGR) functions from Opencv are used. Firstly, the image is converted to float32,and the calculation is done. Then, the saturation value are added to the HLS image. If the pixel values are greater than 1, the values are equal 1. Finally, the HLS image is converted to BGR and multiplied with 255. Returns new image.

adjustContrastOfImage(image, contrasat): This function gets two parameters, image and contrast value. In this method, cv2.addWeighted() function from Opencv is used. Firstly,the alpha and gamma values are calculated. After this calculation, the alpha and gamma values are added to the image. Returns new image.

addNoiseToImage(image,isGray): This function gets two parameters, image and isGray. In this method,np.random.normal(), np.array(), np.reshape() functions from Numpy library are used. Also, random_noise() function from Scikit Image library. If the image is not grayscale, firstly, a random numpy array is created and the pixel values which is multiplied with the random array is added to the original image. If the image is grayscale, random_noise() function is used. Returns noised image.

detectEdgesOfImage(image): This function gets image parameter. In this method, cv2.GaussianBlur() and cv2.Canny functions from Opencv are used. Firstly, the image is blurred with GaussianBlur,then the edges detected with Canny function. Returns detected edges image.

Code Samples

```python
def adjustBrightnessOfImage(image, brightness):
    brightness = mapValue(brightness, -255, 255, -255, 255)
    if brightness != 0:
        if brightness > 0:
            shadow = brightness
            highlight = 255
        else:
            shadow = 0
            highlight = 255 + brightness
        alpha_b = (highlight - shadow) / 255
        gamma_b = shadow
        buf = cv2.addWeighted(image, alpha_b, image, 0, gamma_b)
    else:
        buf = image.copy()

    return buf
```

```
def changeColorBalance(image, balance, r_g_b):
    color = np.zeros_like(image)
    if balance > 255:
        balance = 255
    elif balance < 0:
        balance = 0
    if r_g_b == "Red":
        color[:, :] = [0, 0, balance]
    elif r_g_b == "Green":
        color[:, :] = [0, balance, 0]
    elif r_g_b == "Blue":
        color[:, :] = [balance, 0, 0]
    image = cv2.add(image, color)
    return image
```

From GUI:

```
def show_image(self):
    if self.outputImage is None:

self.outputFrame.setPixmap(QtGui.QPixmap.fromImage(self.inputImage))
    else:

self.outputFrame.setPixmap(QtGui.QPixmap.fromImage(self.outputImage))
    self.messageBox.setText("Message : The image has been shown the left
side of frame succesfully")

    for self.button in self.findChildren(QtWidgets.QPushButton):
        if self.grayscaled:
            if self.button.objectName() == "deBlurBtn" or
self.button.objectName() == "grayScaleBtn":
                self.button.setDisabled(True)
            else:
                self.button.setDisabled(False)
        else:
            self.button.setDisabled(False)
    for valBox in self.findChildren(QtWidgets.QSpinBox):
        if self.grayscaled:

            if valBox.objectName() in ["colorBalanceValue",
"saturationValue", "contrastValue", "blurValue",
                                       "brightnessValue"]:
                valBox.setDisabled(True)
        else:
            valBox.setDisabled(False)

    for checkBox in self.groupBox.children():
        if self.grayscaled:
            if checkBox.objectName() in ["colorBalance", "saturation",
"contrast", "blur",
                                         "brightness"]:
                checkBox.setDisabled(True)
        else:
            checkBox.setDisabled(False)
```

```python
def reset(self):
    qm = QtWidgets.QMessageBox
    ret = qm.question(self, 'Reset Changes', "Do you want to reset all
changes?", qm.Yes | qm.No)
    if ret == qm.Yes:
        for checkBox in self.groupBox.children():
            checkBox.setChecked(False)
        for box in self.findChildren(QtWidgets.QSpinBox):
            box.setValue(0)
        self.grayscaled = False
        self.saved = False
        self.image = main.loadImage(self.filepath[0])
        self.width, self.height = self.image.shape[0], self.image.shape[1]
        self.image = main.resizeImage(self.image, 400, 400)
        self.outputImage = None
        self.inputImage = QtGui.QImage(self.image.data,
self.image.shape[1], self.image.shape[0],

QtGui.QImage.Format_RGB888).rgbSwapped()
        self.messageBox.setText("Message : The image has been uploaded
succesfully")

        self.inputFrame.setPixmap(QtGui.QPixmap.fromImage(self.inputImage))
        self.rubberband =
QtWidgets.QRubberBand(QtWidgets.QRubberBand.Rectangle, self.inputFrame)
        self.setMouseTracking(True)
        self.show_image()
def load_image(self):
    if not self.saved:
        qm = QtWidgets.QMessageBox
        ret = qm.question(self, 'Save', "Do you want to save the current
Image?", qm.Yes | qm.No)
        if ret == qm.Yes:
            self.save()
    self.filepath = QtWidgets.QFileDialog.getOpenFileName(self, 'Hey!
Select an Image',
                                                          filter="Image
Files(*.jpeg;*.jpg;*.png)")
    if len(self.filepath[0])==0:
        self.messageBox.setText("Message : image can not uploaded")
        self.progressBar.setValue(100)


    if not len(self.filepath[0]) == 0:
        self.grayscaled = False
        self.saved = False
        self.image = main.loadImage(self.filepath[0])
        self.width, self.height = self.image.shape[0], self.image.shape[1]
        self.image = main.resizeImage(self.image, 400, 400)
        self.outputImage = None
        self.inputImage = QtGui.QImage(self.image.data,
self.image.shape[1], self.image.shape[0],

QtGui.QImage.Format_RGB888).rgbSwapped()
        self.messageBox.setText("Message : The image has been uploaded
succesfully")

        self.inputFrame.setPixmap(QtGui.QPixmap.fromImage(self.inputImage))
        self.rubberband =
QtWidgets.QRubberBand(QtWidgets.QRubberBand.Rectangle, self.inputFrame)
```

```
        self.setMouseTracking(True)
        self.show_image()
```

Conclusion

With this project, we put into practice the subjects we saw in both our theoretical and applied courses. We have used our knowledge on image processing and image development in a way that will contribute to our business life. We have gained experience in the operations we perform on images. We had the chance to see the areas we lacked on the subjects we saw in the lessons. The fact that this project is a group assignment contributed to us working as a team.