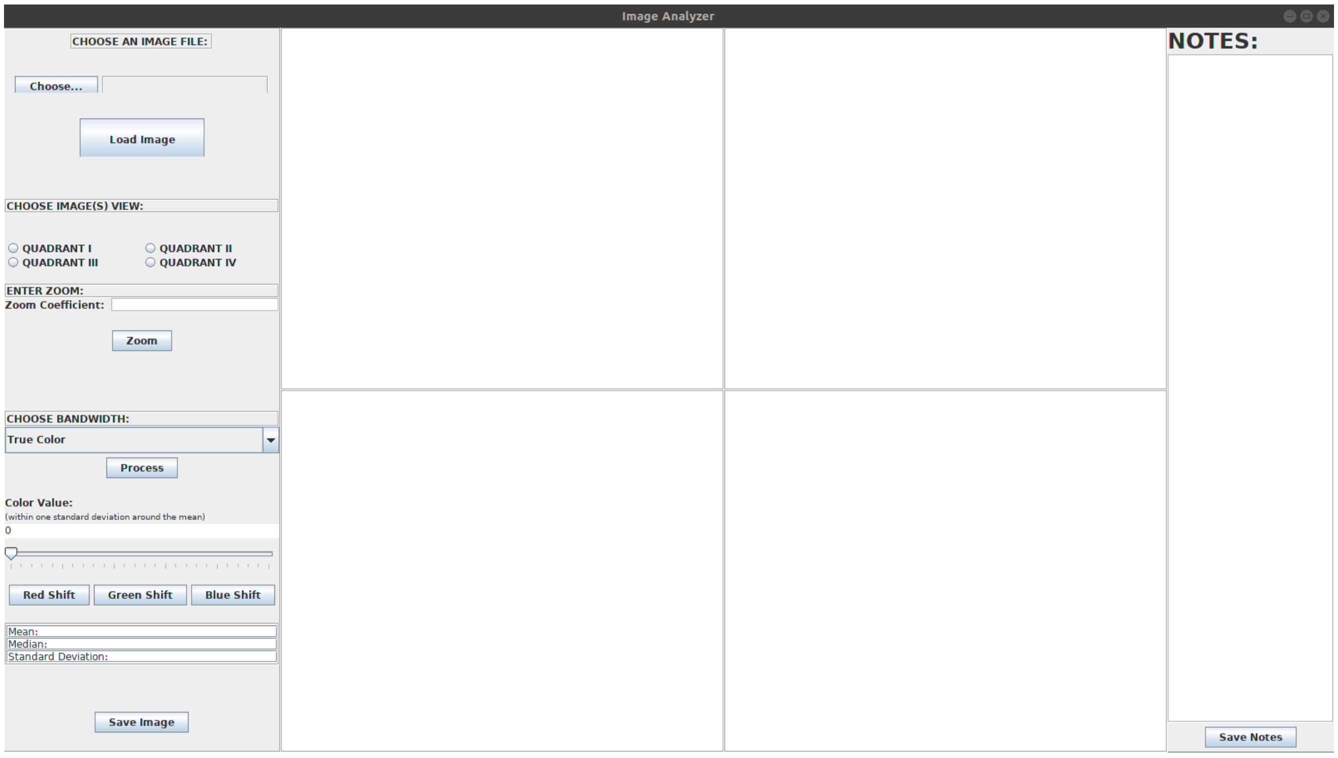
Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

Graduate Project – The Image Analyzer

Introduction:

In an attempt to incorporate some of the information from an Image Processing class at the University of Florida, I built a software program with the purpose of analyzing pixels in an images. Initially I planned for the program to load multispectral and hyperspectral imagery, but because of time constraints the program will only load standard RGB and RGBA images. Additionally, I had to make some modifications from the original proposal on the output. My hope was to be able to classify the pixel alterations, but that also was not possible in the given time. Therefore, I included a notes panel in the east panel (right side) for the user to take notes, and the option to save the notes as a .dat file is incorporated at the bottom.

The overall appearance of the program is clean and well organized. The colors were left unchanged, except for the center panel which was made white. This is to maintain color neutrality for analyzing color pixels on the images to be processed. The left panel contains all the functions allowable. The center pane consists of a four square grid on a white background. This is where the image(s) for analysis will load, and they are referenced in the left panel by quadrant. These four squares are bordered to prevent the images from touching one another. One improvement I would like to investigate is the font styles and sizes in the left control panel. I did make some small adjustments to the font, but nothing notable. I feel highlighting the menu labels would make the program a little more user friendly.



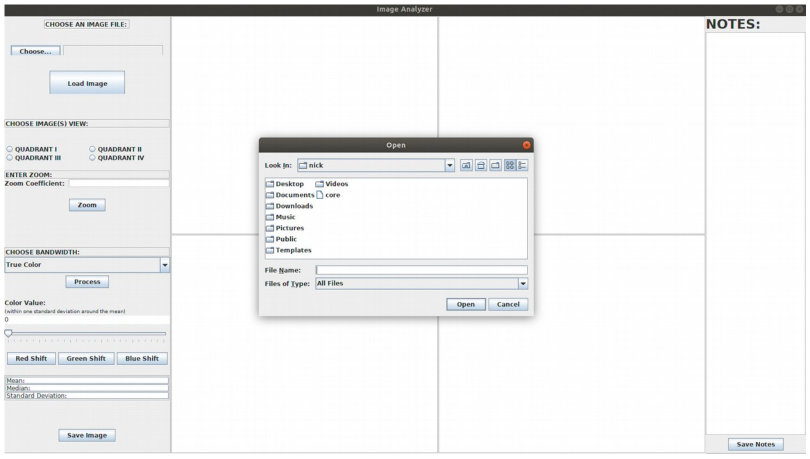
Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

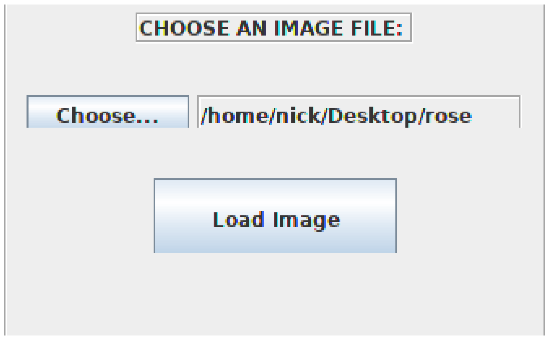
Control Panel:

The left panel of the program is the primary control panel. The functions allow user to choose a file path, load the image in the file path, choose the quadrant within the view pane (center), zoom into the center of the image using a zoom coefficient, choose an individual bandwidth (or reset to true color), make a color shift in a specific band, observe the image pixel array's statistics, and save the image. These are the somewhat simplistic image analysis tools I was able to implement in the program.

Choose an Image File:

The "Choose File" button allows the user to choose a file from the file directory. The button is instantiated in the GUI2 class using an ActionListener. The ActionListener passes the string value of the button to a new Backend object using a feedString method (this method was overloaded four times). The feedString method calls on the method chooseFile to open the file directory, and then place the string value of the directory in the text field next to the 'Choose File' button. The "Load Image" button completes this function by also using an ActionListener to pass the string value to the Backend object one of the feedString methods. The feedString method then calls on the loadImage method, which reads in the image at the file path, saves the image to the BufferedImage instance variable image, and then saves the BufferedImage as the ImageIcon the corresponds with the quadrant selected.



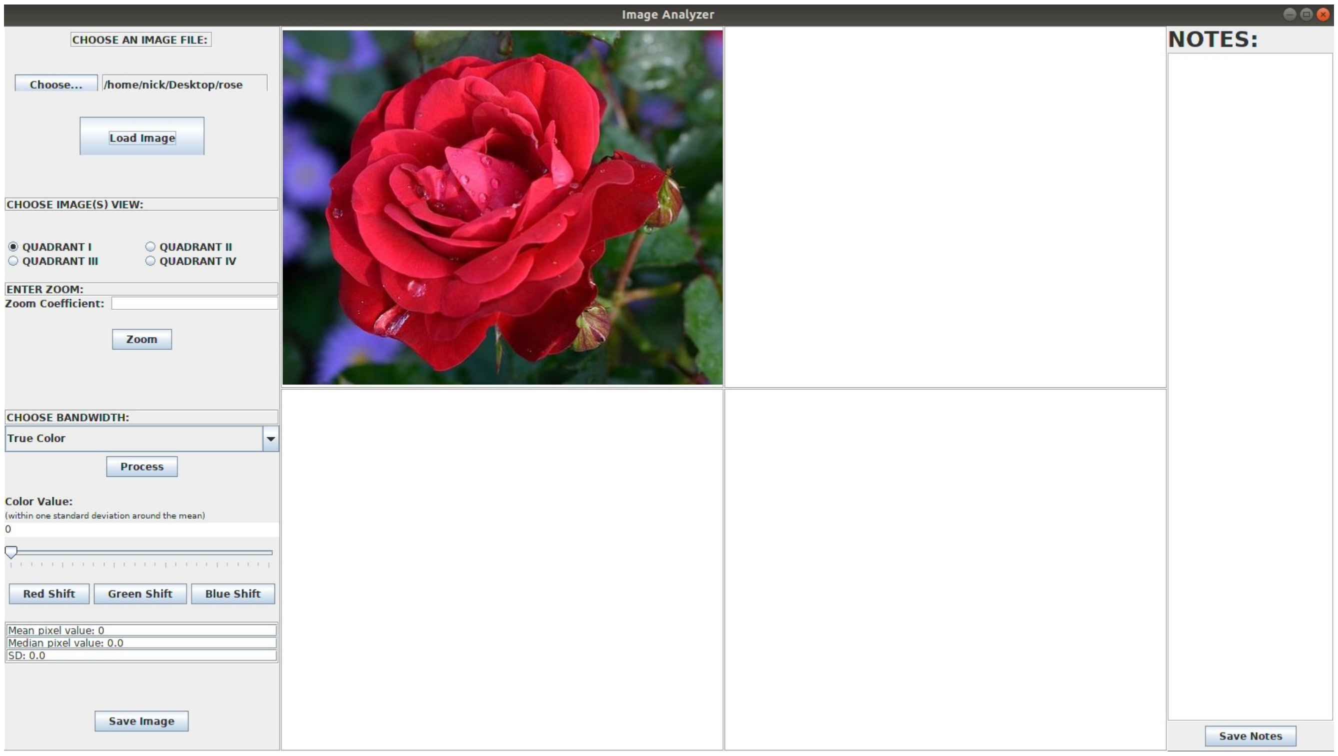


Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

<u>Choose Image(s) View</u>:

In order to load an image, or perform any function on the image, the image view must be chosen. The options are simply Quadrant I – IV. Originally I implemented the quadrants to follow the Cartesian coordinate system, but it was more user friendly and intuitive to follow a left to right, up to down flow. The quadrants are chosen using radio buttons, and any number of quadrants can be selected simultaneously. The radio buttons are instantiated in the GUI2 class. The selection of these radio buttons is used in other methods to determine which ImageIcon to represent on the Jlabel used to create the four-square grid image viewer. One refinement I would consider is passing these radio buttons to the Backend, and then building the methods around which quadrant is selected there. I believe this is the reason my ActionListener became so long winded.

IMAGE LOADED INTO QUADRANT I:



/home/nick/Desktop/rose

[✓ QUADRANT I] QUADRANT II
QUADRANT III QUADRANT IV

Zoom Coefficient:

True Color

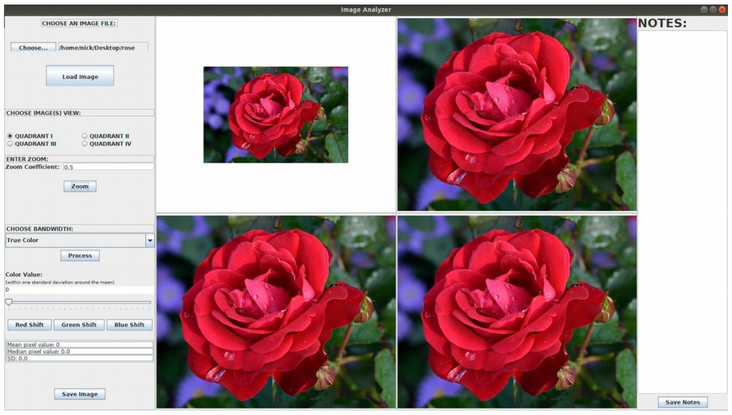
Color Value:
(within one standard deviation around the mean)
0

Mean pixel value: 0
Median pixel value: 0.0
SD: 0.0

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

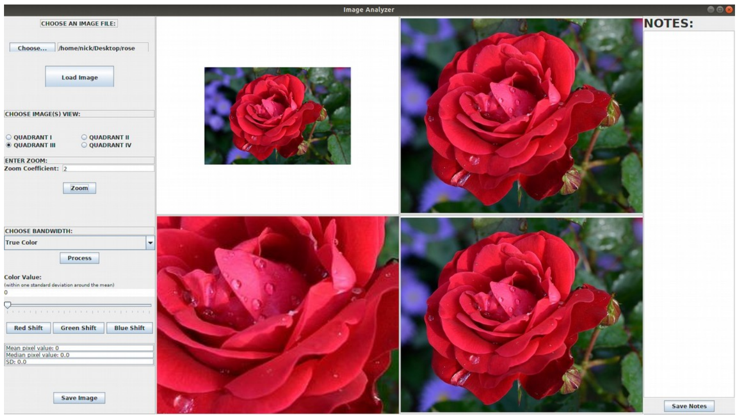
<u>Enter Zoom</u>:

The enter zoom panel allows the user to type in a zoom coefficient and then apply the coefficient to the image quadrant selected. The coefficient is simply a real number that will increase or decrease the size of the image depending on whether the coefficient value is greater than one, less than one, or equal to one. If a user types in a value equal or less than zero the program will not change the image size and remain idle. One feature I did not implement that would go well with this would be to pan the image.



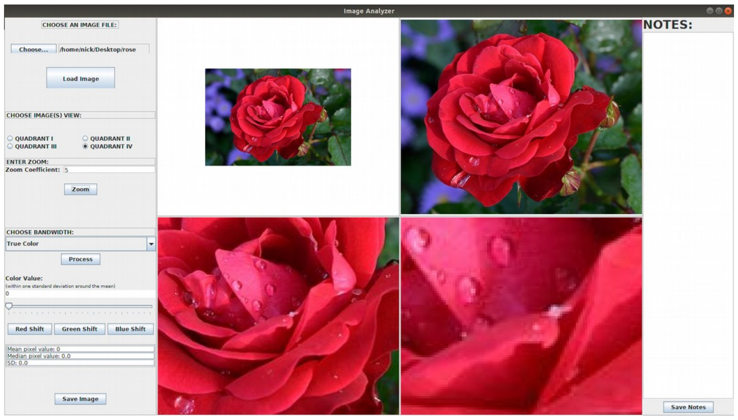
/home/nick/Desktop/rose
[✓ QUADRANT I] QUADRANT II QUADRANT III QUADRANT IV
Zoom Coefficient: 0.5
True Color
0
Mean pixel value: 0
Median pixel value: 0.0
SD: 0.0

ZOOM COEFFICIENT OF 0.5 IN QUADRANT I



/home/nick/Desktop/rose
QUADRANT I QUADRANT II [✓ QUADRANT III] QUADRANT IV
Zoom Coefficient: 2
True Color
0
Mean pixel value: 0
Median pixel value: 0.0
SD: 0.0

ZOOM COEFFICIENT OF 2 IN QUADRANT III



/home/nick/Desktop/rose
QUADRANT I QUADRANT II QUADRANT III [✓ QUADRANT IV]
Zoom Coefficient: 5
True Color
0
Mean pixel value: 0
Median pixel value: 0.0
SD: 0.0

ZOOM COEFFICIENT OF 5 IN QUADRANT IV

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

Choose Bandwidth:

The choose bandwidth panel allows the user to isolate a single band. This is useful for identification purposes of various materials. A few examples of how this might be used would be the identification of vegetation. By isolating specific bandwidths it is possible to isolate healthy from unhealthy vegetation, species of vegetation, aquatic vegetation from terrestrial vegetation, et cetera. Additionally, astrophysicists use spectral bands to identify the materials stars and planets consist of.

The choose bandwidth combo box gives the user four options: true color, red, green, or blue. The true color band is simply the original image (without the zoom function applied), the red band is the original red pixel values and the blue and green pixels values are averaged, the green is the original green pixel values with red and blue pixel values averaged, and the blue is the original blue pixel values with red and green pixel values averaged.

The combo box buttons are instantiated in the GUI2 class used inside the ActionListner to determine which values to pass to the Backend object feedString method. It is the “Process” button’s string that is passed to the feedString method in the Backend object, but the parameters are different based on which combo box button is chosen, and which quadrant is chosen. The feedString method takes a string and two int values. The string is the “Process” button value.

This is all implemented in the GUI2 class using the processImage method. The processImage method takes the Backend object, the ImageIcon that saves the image from the Backend, the Jlabel that is set to the ImageIcon, and the parameters for the Backend object feedString method to operate.

In the Backend object feedString uses the processImageOne, processImageTwo, processImageThree, or processImageFour to create a new Processor object, and then depending on which combo box button was selected, return the adjusted image using the Processor objects getPixels method to build individual arrays for each bandwidth, count the total pixel values per band, and to count the total number of pixels per image. These values are then returned when the processImage(One, Two, Three, or Four) method calls either Processor object getRed, getGreen, or getBlue. The ImageIcon is then set to the new isolated bandwidth image and passed back to the GUI using the Backend object method getLabel. Also, the Processor object enables the statistics to be performed on the isolated pixel arrays.

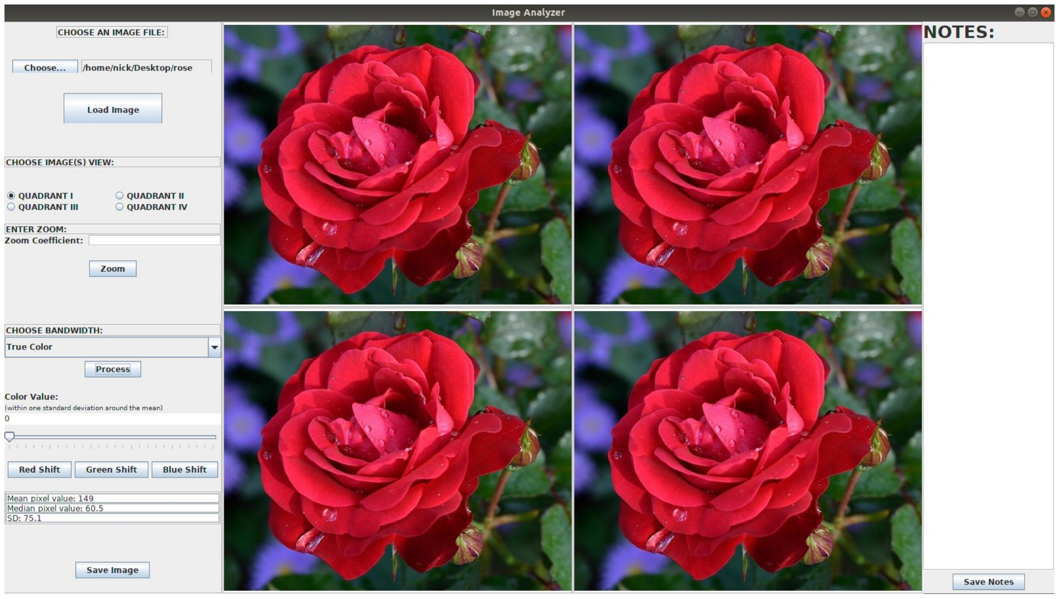
The only deviation from this is the true color pixel array which is simply the reloading of the saved file location that was originally selected. The string instance variables for file locations are saved in the loadImage method when an image is initially loaded. Also, because of this the statistics for the true color are calculated in the Backend using the toArray method, mean method, getMedian method, and getSD method.

Some refinements I would consider here are first to reduce the size of the ActionListener in the GUI2 class by pass the combo buttons to the backend. This is a similar mistake as what I did with the radio buttons. Also, the true color statistics are incorrect when the “Process” button is pressed multiple times. The first selection is correct, but I suspect that because I am reloading the image, but not the

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

toArray method, the calculations become influenced by this change. I think that if I can figure out how to move the true color processing into the Processor object it will resolve this.

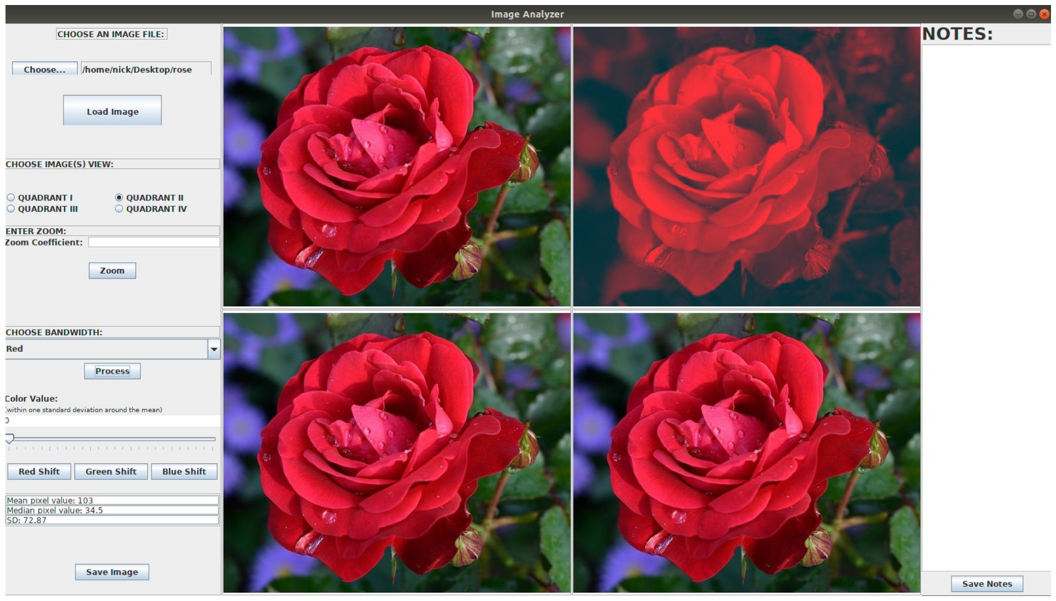
PROCESS TRUE COLOR STATISTICS ON QUADRANT I:



/home/nick/Desktop/rose
[✓ QUADRANT I] QUADRANT II QUADRANT III QUADRANT IV
Zoom Coefficient:
True Color
Color Value:
(within one standard deviation around the mean)
0

```
Mean pixel value: 149
Median pixel value: 60.5
SD: 75.1
```

PROCESS RED STATISTICS ON QUADRANT II:

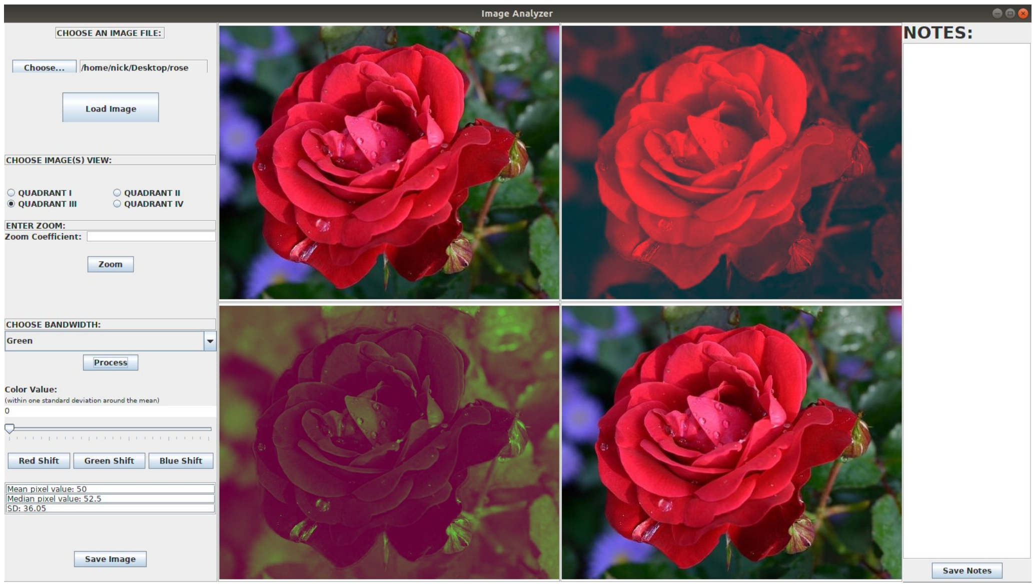


/home/nick/Desktop/rose
QUADRANT I [✓ QUADRANT II] QUADRANT III QUADRANT IV
Zoom Coefficient:
Red
Color Value:
(within one standard deviation around the mean)
0

```
Mean pixel value: 103
Median pixel value: 34.5
SD: 72.87
```

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

PROCESS GREEN STATISTICS ON QUADRANT III:



/home/nick/Desktop/rose

CHOOSE IMAGE(S) VIEW:
QUADRANT I
QUADRANT II
[✓ QUADRANT III]
QUADRANT IV

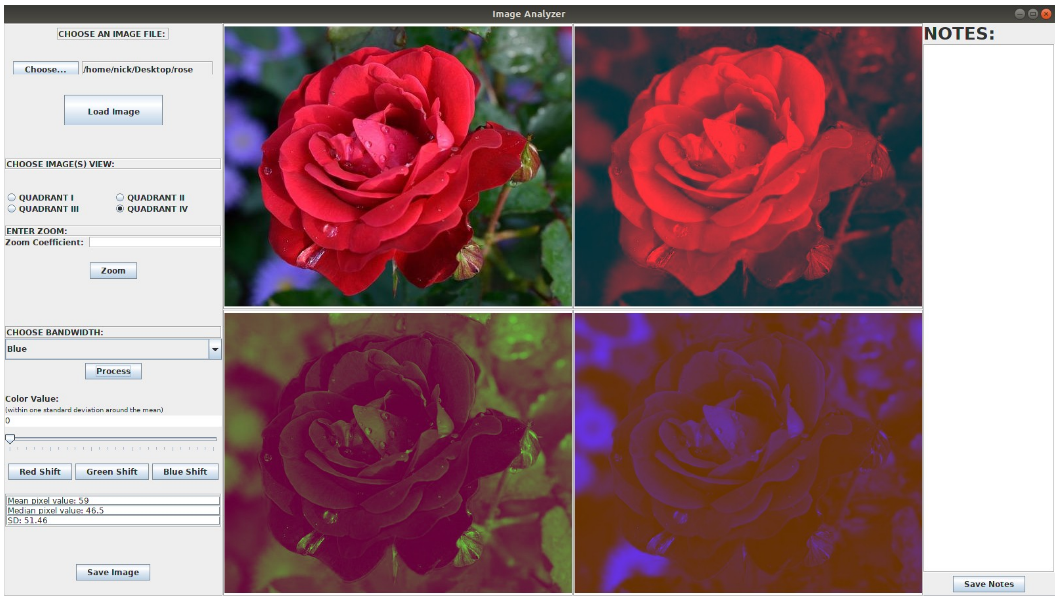
ENTER ZOOM:
Zoom Coefficient:

CHOOSE BANDWIDTH:
Green

Color Value:
(within one standard deviation around the mean)
0

Mean pixel value: 50
Median pixel value: 52.5
SD: 36.05

PROCESS BLUE STATISTICS ON QUADRANT IV:



/home/nick/Desktop/rose

CHOOSE IMAGE(S) VIEW:
QUADRANT I
QUADRANT II
QUADRANT III
[✓ QUADRANT IV]

ENTER ZOOM:
Zoom Coefficient:

CHOOSE BANDWIDTH:
Blue

Color Value:
(within one standard deviation around the mean)
0

Mean pixel value: 59
Median pixel value: 46.5
SD: 51.46

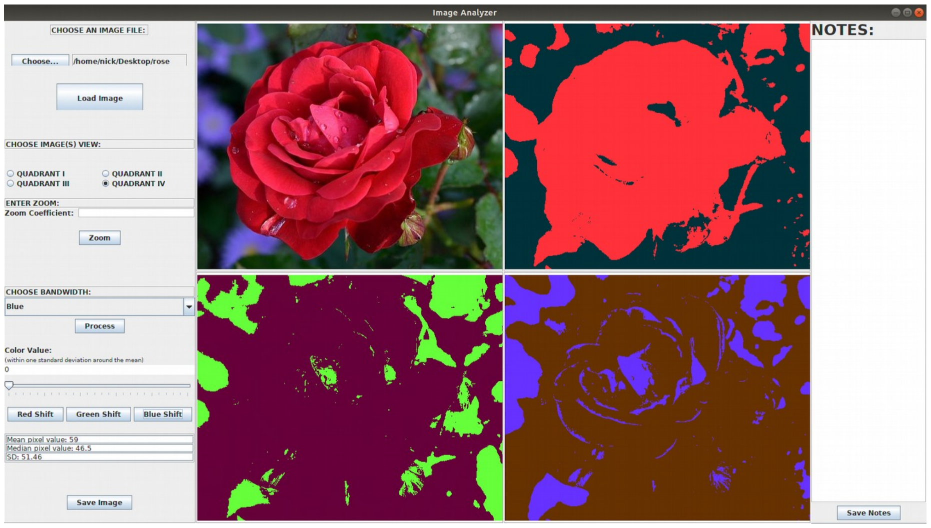
Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

<u>Color Shift and Color Value Slider</u>:

The color shift tool allows the user to shift the pixel values that are greater than or less than one standard deviation to the extreme values. If the pixel value is less than one standard deviation (half the standard deviation from the mean) the pixel value is set equal to 0. If the pixel value is greater than one standard deviation (half the standard deviation from the mean) the pixel value is set to 255. The slider allows the user to adjust the value of all of the pixels within one standard deviation. This enables the user to view the extreme bandwidth values, to flood the band width values, or to see anything in between.

The shift buttons and the slider are instantiated in the GUI2 class and passed to the Backend object with the ActionListener and using the shiftImage method found in GUI2 class. The shiftImage method passes a string value used to identify the button, an int used to identify the image quadrant, and a double that is the value selected by the slider to the feedString method in the Backend. Inside the Backend object the feedString method calls either redSeparation, greenSeparation, or blueSeparation depending on the string value of the button selected. Each of the three separation buttons takes two parameters, an int value the corresponds to the image quadrant and a double value that is the shift value selected by the slider. These methods set the ImageIcon that corresponds to the int value for the quadrant to a new ImageIcon that takes either a redMeanSeperation, greenMeanSeperation, or blueMeanSeperation method, depending on the button value chosen. The MeanSeperation methods iterate through the image pixel array and pull out the pixels, set them to red, green, or blue, and then perform the necessary operations to separate the extreme values from the values within one standard deviation. These MeanSeperation methods return a BufferedImage of the separated value which is in turn converted to an ImageIcon in the Separation methods mentioned above.

TRUE COLOR IMAGE QUADRANT I, RED SHIFT ON RED BANDWIDTH QUADRANT II, GREEN SHIFT ON GREEN BANDWIDTH QUADRANT III, BLUE SHIFT ON BLUE BANDWIDTH QUADRANT IV



/home/nick/Desktop/rose

QUADRANT I QUADRANT II
QUADRANT III [✓ QUADRANT IV]

Zoom Coefficient:

Blue

Color Value:
(within one standard deviation around the mean)
0

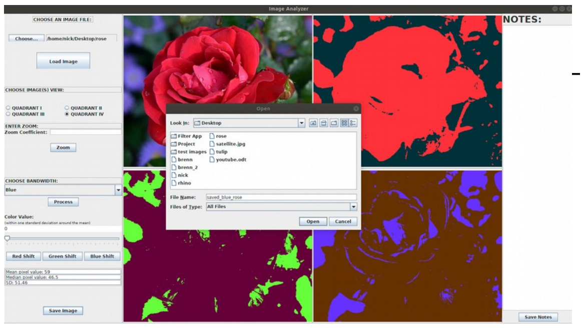
Mean pixel value: 59
Median pixel value: 46.5
SD: 51.46

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

Save Image:

The save image button will save the image(s) selected. The user can save the file(s) by the name they like, and in whichever directory they choose from the directory chooser that opens.

This function is implemented similarly to the load image button. The button is instantiated in the GUI2 class and passed to the Backend object feedString method with the ActionListner. The feedString method calls on the saveImage method, and the image is saved based on the int parameter passed which identifies the image quadrant.



/home/nick/Desktop/rose

Blue

0

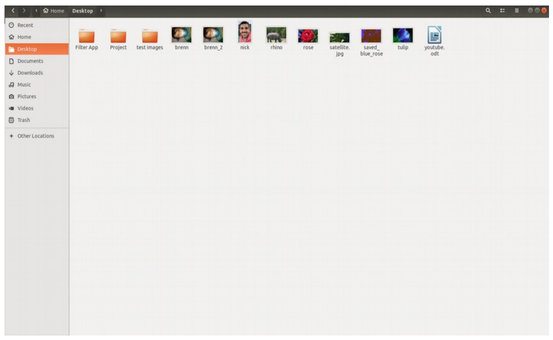
Mean pixel value: 59
Median pixel value: 46.5
SD: 51.46

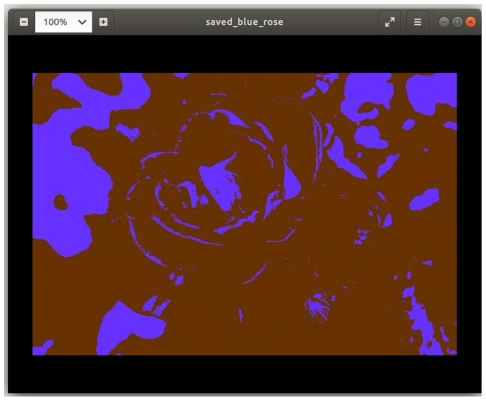
Look In: Desktop

File Name: saved_blue_rose

Files of Type: All Files

-





Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

Notes:

The notes section was added because it seemed to be a good tool for someone to have while analyzing an image, and because this program is not at the point where it can print out details for specified pixel values. There is a save notes button that allows the user to save notes as a .dat file.

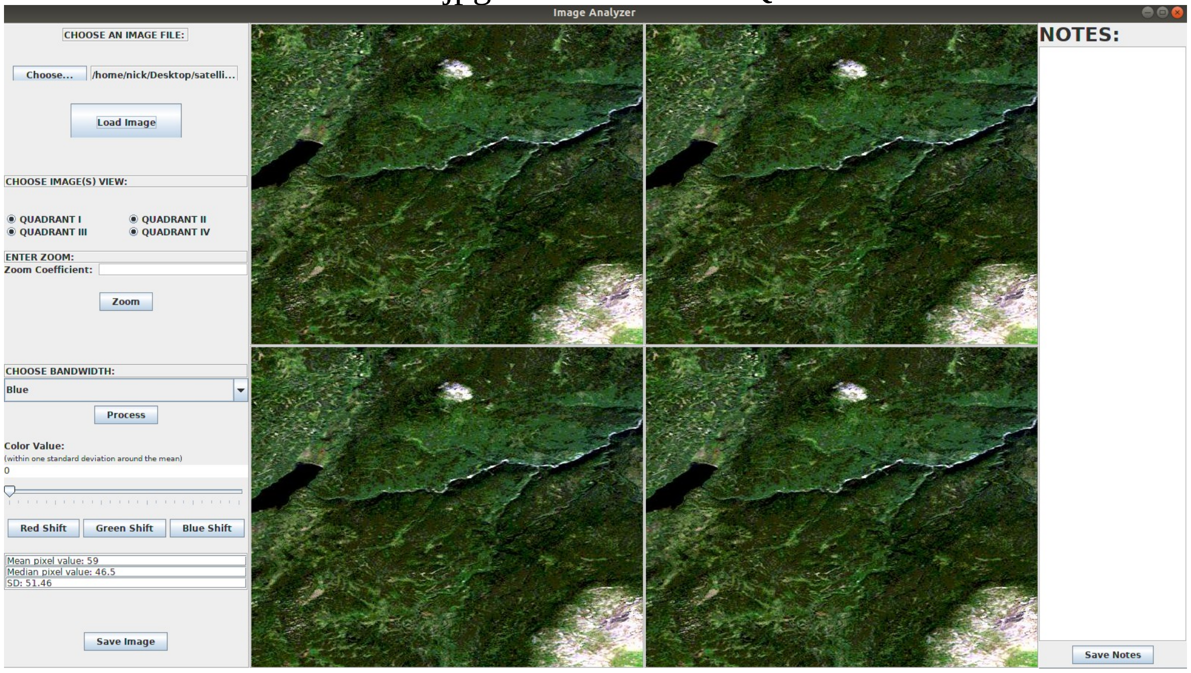
This function is instantiated in the GUI2 class and uses the DocumentListener and ActionListener to pass the string value of the notes typed to the feedString method inside the Backend object. The feedString method calls a saveNotes method that opens th file chooser and allows the user to choose where to save the notes. The method then saves the string in the chosen location using a BufferedWriter.

The zoom function works similarly by using the DocumentListner to create an instance variable, and then passing it to the Backend object feedString method.

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

Quick Example:

LOAD satellite.jpg TO ALL FOUR QUADRANTS:



/home/nick/Desktop/satelli...

[✓ QUADRANT I] [✓ QUADRANT II]
[✓ QUADRANT III] [✓ QUADRANT IV]

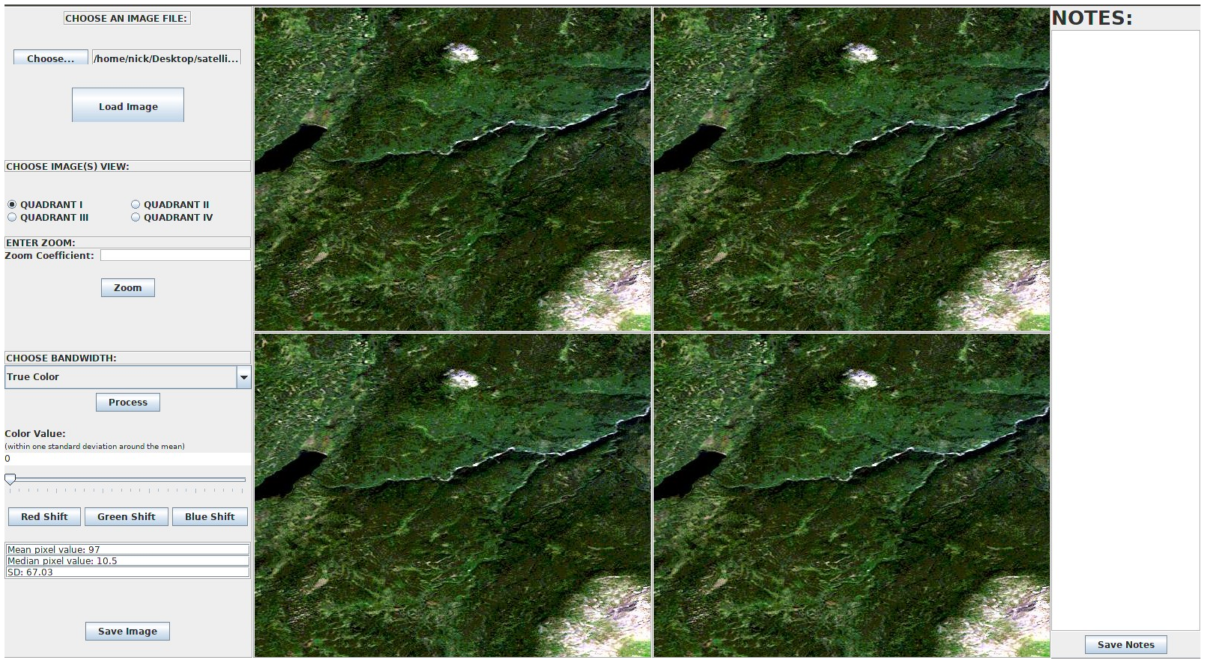
Zoom Coefficient:

Blue

0

Mean pixel value: 59
Median pixel value: 46.5
SD: 51.46

PROCESS THE TRUE COLOR STATISTICS FOR QUADRANT I:



/home/nick/Desktop/satelli...

[✓ QUADRANT I] QUADRANT II
QUADRANT III QUADRANT IV

Zoom Coefficient:

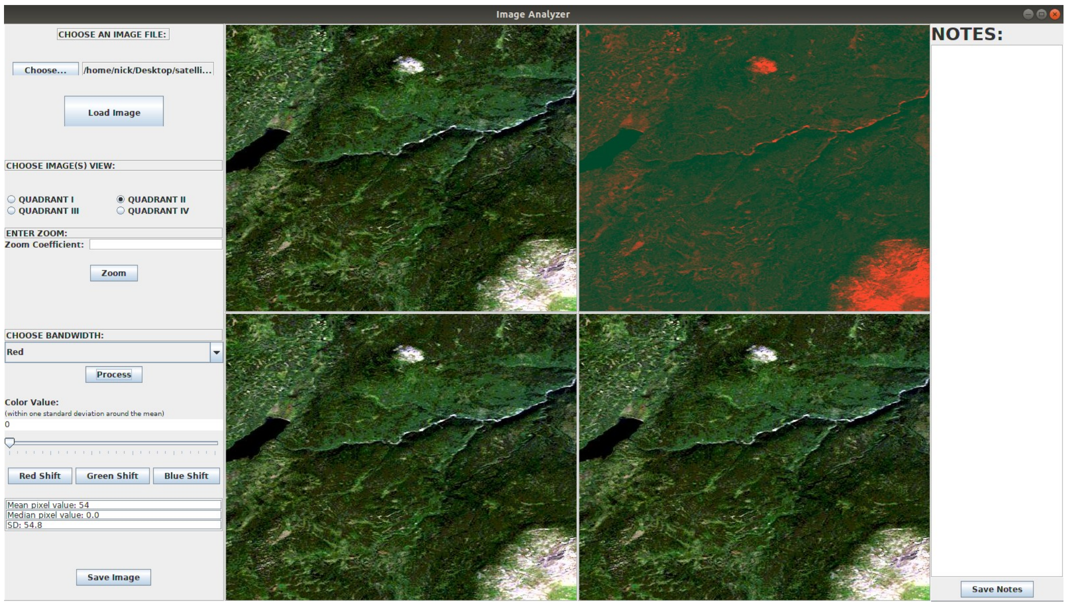
True Color

0

Mean pixel value: 97
Median pixel value: 10.5
SD: 67.03

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

PROCESS RED STATISTICS AND RED BANDWIDTH – QUADRANT II



/home/nick/Desktop/satelli...

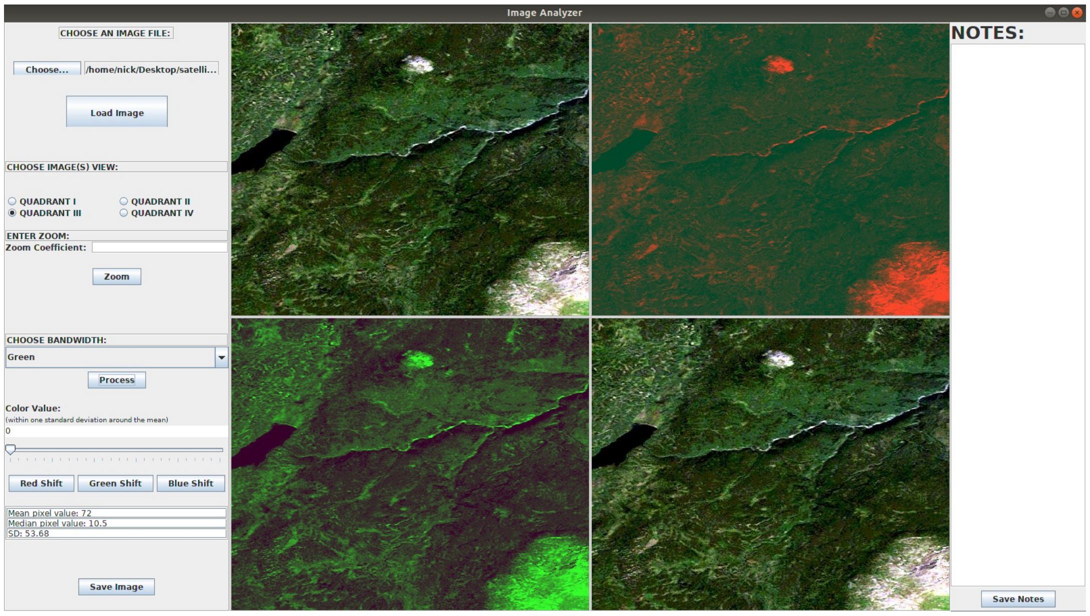
( ) QUADRANT I (●) QUADRANT II
( ) QUADRANT III ( ) QUADRANT IV

Bandwidth: Red

Color Value: 0

Mean pixel value: 54
Median pixel value: 0.0
SD: 54.8

PROCESS GREEN STATISTICS AND RED BANDWIDTH – QUADRANT III



/home/nick/Desktop/satelli...

( ) QUADRANT I ( ) QUADRANT II
(●) QUADRANT III ( ) QUADRANT IV

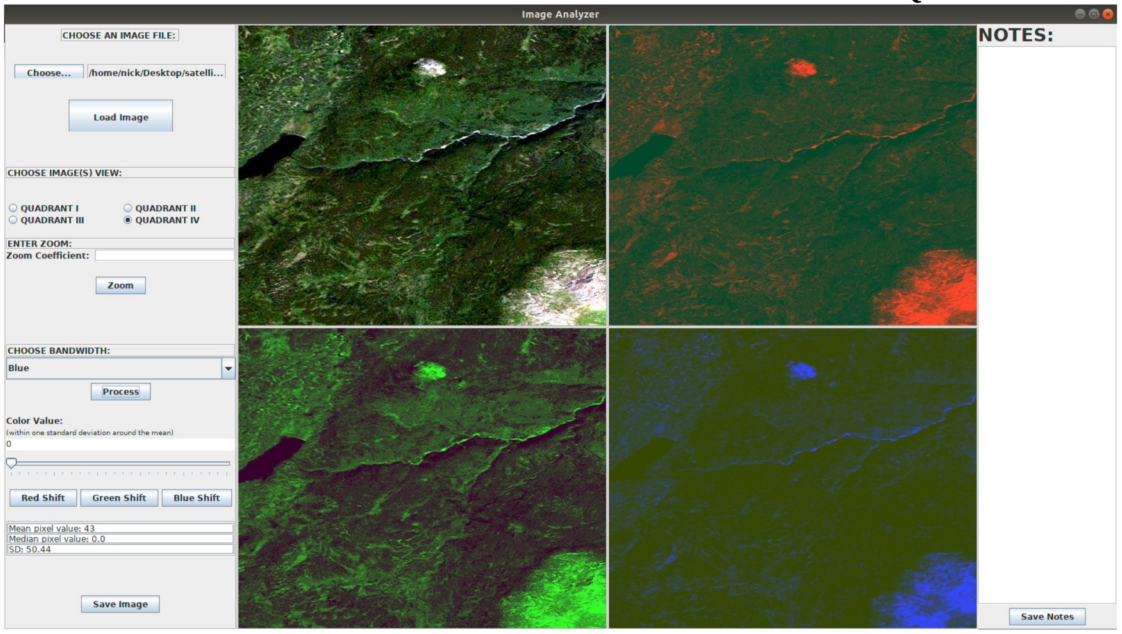
Bandwidth: Green

Color Value: 0

Mean pixel value: 72
Median pixel value: 10.5
SD: 53.68

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

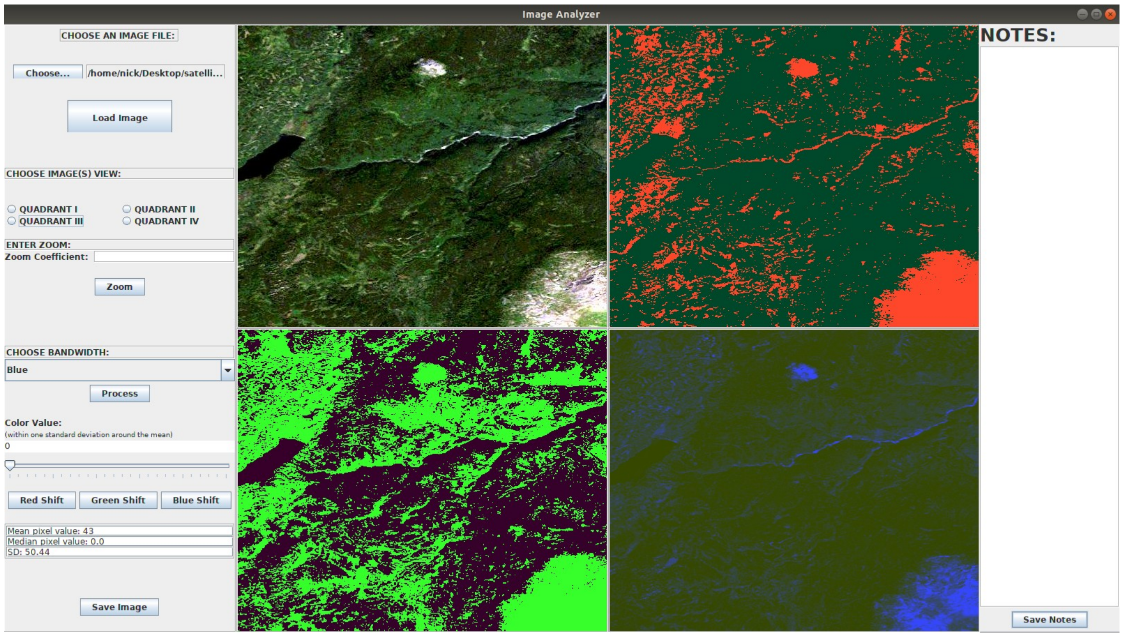
PROCESS BLUE STATISTICS AND RED BANDWIDTH – QUADRANT IV



/home/nick/Desktop/satelli...
[✓ QUADRANT IV]
Blue
0
Mean pixel value: 43
Median pixel value: 0.0
SD: 50.44

Because vegetation reflects green bandwidths and absorbs red bandwidths we will first look at the extreme red and green values:

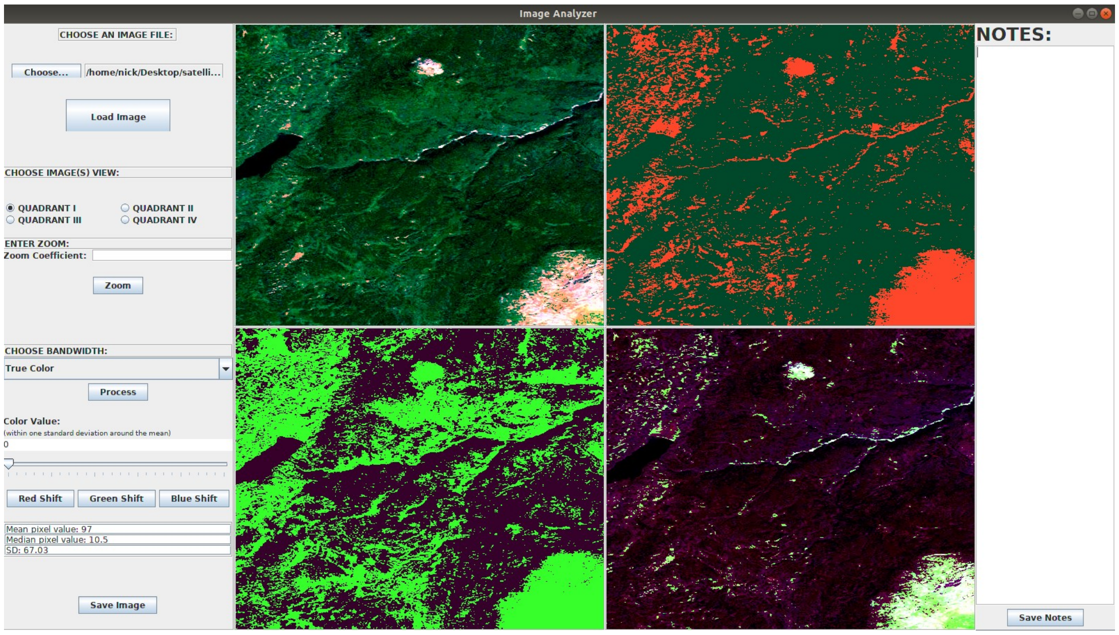
-



/home/nick/Desktop/satelli...
Blue
0
Mean pixel value: 43
Median pixel value: 0.0
SD: 50.44

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

Now we apply a Red Shift and a Green Shift to the True Color Image in Quadrants I and IV



/home/nick/Desktop/satelli...

[✓ QUADRANT I] QUADRANT II QUADRANT III QUADRANT IV

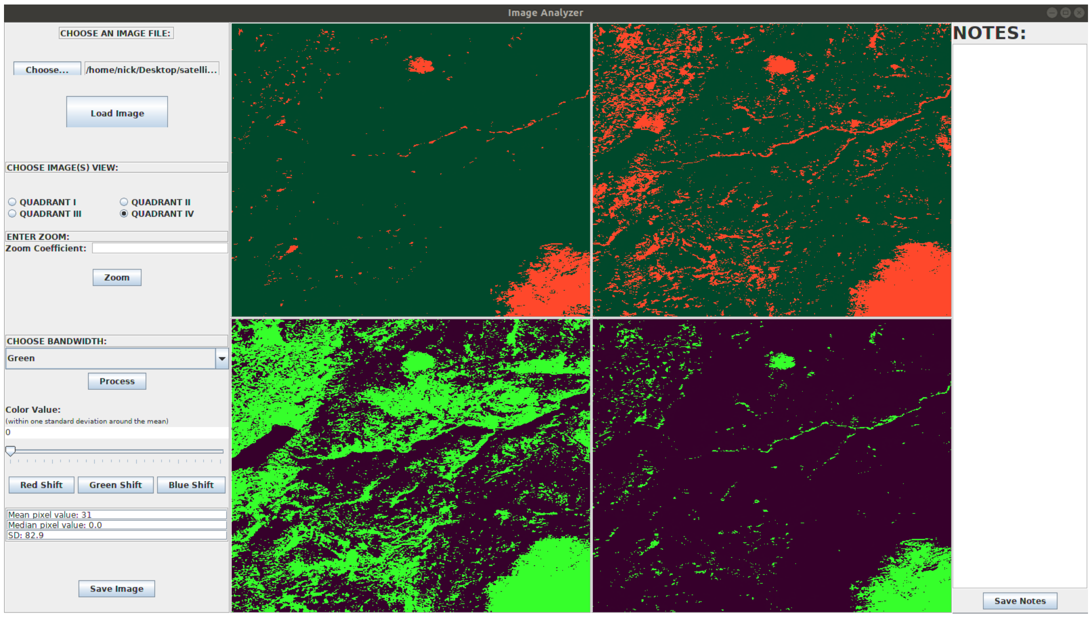
Zoom Coefficient:

True Color

Color Value:
(within one standard deviation around the mean)
0

Mean pixel value: 97
Median pixel value: 10.5
SD: 67.03

Next we choose only the red bandwidth for quadrant I and only the green bandwidth for quadrant IV – this eliminates all bands other than the extreme red in I and green in IV



/home/nick/Desktop/satelli...

QUADRANT I QUADRANT II QUADRANT III [✓ QUADRANT IV]

Zoom Coefficient:

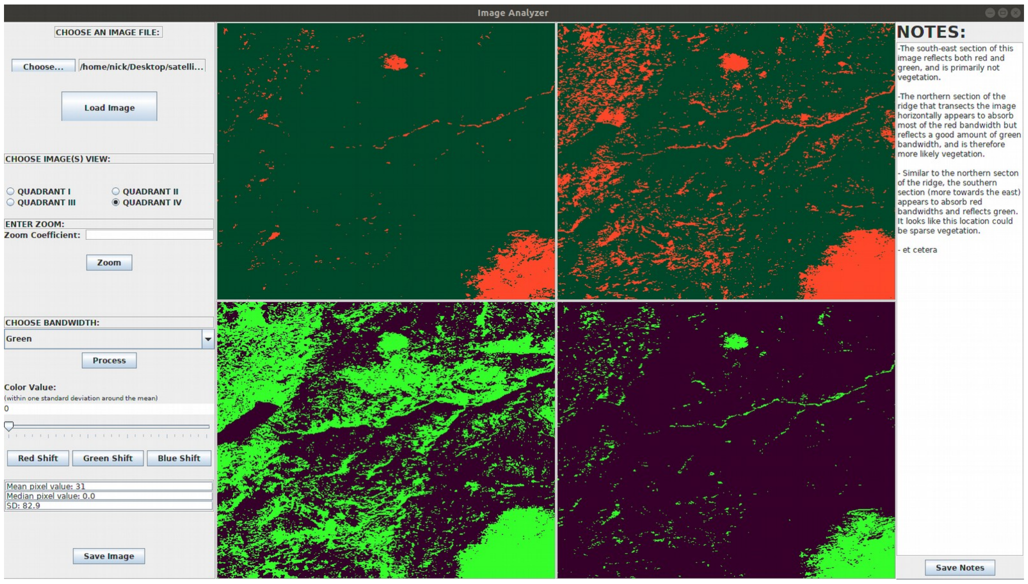
Green

Color Value:
(within one standard deviation around the mean)
0

Mean pixel value: 31
Median pixel value: 0.0
SD: 82.9

Nicholas A. Grokhowsky
CSCI E-10b
Professor: Dr. Henry H. Leitner
TF: Christopher Morris

Take notes on features analyzed



/home/nick/Desktop/satelli...

QUADRANT I QUADRANT II
QUADRANT III [✓ QUADRANT IV]

Zoom Coefficient:

Green

Color Value:
(within one standard deviation around the mean)
0

Mean pixel value: 31
Median pixel value: 0.0
SD: 82.9

NOTES:

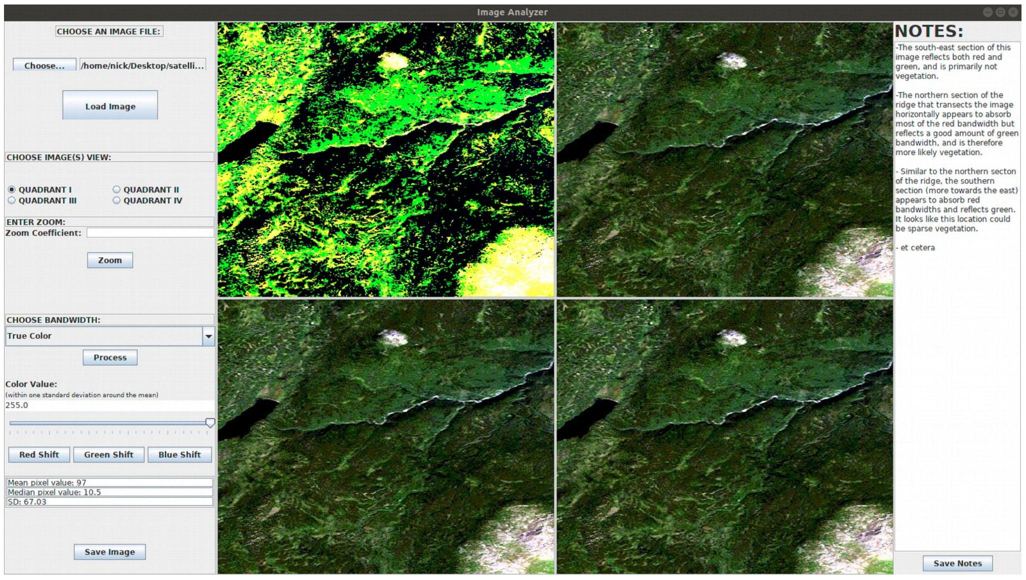
-The south-east section of this image reflects both red and green, and is primarily not vegetation.

-The northern section of the ridge that transects the image horizontally appears to absorb most of the red bandwidth but reflects a good amount of green bandwidth, and is therefore more likely vegetation.

- Similar to the northern secton of the ridge, the southern section (more towards the east) appears to absorb red bandwidths and reflects green. It looks like this location could be sparse vegetation.

- et cetera

Combine extreme red shift with extreme green shift – quadrant I



/home/nick/Desktop/satelli...

[✓ QUADRANT I] QUADRANT II
QUADRANT III QUADRANT IV

Zoom Coefficient:

True Color

Color Value:
(within one standard deviation around the mean)
255.0

Mean pixel value: 97
Median pixel value: 10.5
SD: 67.03

NOTES:

-The south-east section of this image reflects both red and green, and is primarily not vegetation.

-The northern section of the ridge that transects the image horizontally appears to absorb most of the red bandwidth but reflects a good amount of green bandwidth, and is therefore more likely vegetation.

- Similar to the northern secton of the ridge, the southern section (more towards the east) appears to absorb red bandwidths and reflects green. It looks like this location could be sparse vegetation.

- et cetera

CONTINUE WITH MORE IMAGE MANIPULATION AND ANALYSIS.