CSE 107: Lab 02: Simple Image Manipulations in Python.

Cyrus Adriel Arellano
LAB: Th 7:30pm-10:20pm
Haolin Liang
September 26 , 2022

Task 1: Computing the maximum value of an image. Rotating an image.

Figure 1: Rotated 90 Degrees Clockwise Gray Scale Image of Beginnings.jpg

1. What is the maximum pixel value of your grayscale Beginnings image?

The maximum pixel value of the grayscale Beginnings image is 240.

2. What is the maximum pixel value of your clockwise rotated grayscale image?

The maximum pixel value of the clockwise rotated grayscale image is 240.

3. Should these be the same? Why or why not?

These should be the same because we are not changing the pixel values but instead copying pixel values from the 800x533 image and transferring it to the 533x800 matrix so that it is rotated.

4. What was the most difficult part of this task?

The most difficult part of this task was figuring out how to rotate the image and what was needed in the nested for loop. My image would be separated into pieces which was hard to figure out.

Task 2: Writing a function that computes the inverse of a grayscale image.

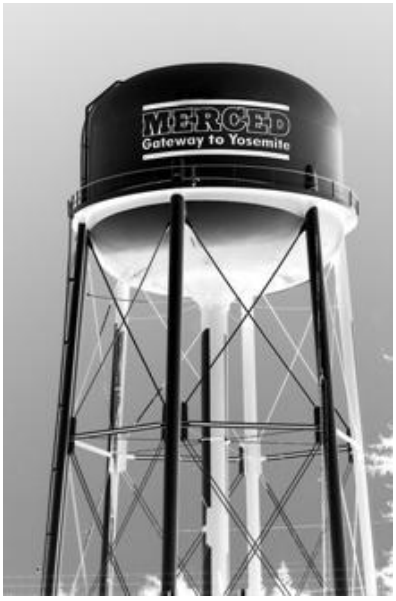

Figure 2: Inverted image of Watertower.tif

1. What is the maximum pixel value of your inverse image?

The maximum pixel value of the inverse image is 255 pixel value.

2. How is this maximum value related to the values of the original image?

This maximum value is related to the values of the original image because wherever the pixel has the value of 255, that means the original image has the pixel value of 0 on the same spot. We know this because the nested for loop contains the equation out_value = 255 - in_value.

3. What was the most difficult part of this task?

The most difficult part of this task was trying to figure out how to make a new numpy matrix for a new image and getting 255 and subtracting it to the original image then putting it into a new matrix.

Task 3: Creating a gradient grayscale image. Computing the image average.

Figure 3: Gradient Image of 0-255 pixel value

1. What is the average pixel value in your gradient image?

When calculating the average pixel value, it would be 127.5 in which it would be the average pixel value.

2. Why did you expect to get this value from the gradient image?

I expected to get this value because we are counting from 0-255 and trying to find the average would be 127.5 which is half of 255.

3. What was the most difficult part of this task?

The most difficult part of this task was the nested for loop and figuring out how to implement each row in one column to have the same pixel value.

Task1.py Code:

```
from PIL import Image, ImageOps
import numpy as np
from numpy import asarray

#Takes in image
im = Image.open('Beginnings.jpg')

#Converts image to gray scale
im_gray = ImageOps.grayscale(im)
#Gets matrix of pixel values
im_pixels = asarray(im_gray)

#Gets dimensions
rows, cols = im_pixels.shape
im_gray_pixels = asarray(im_gray)

#Switches rows and columns so that it can rotate 90 degrees clockwise
rotated_cols = rows
rotated_rows = cols
rotated_pixels = np.zeros(shape=(rotated_rows,rotated_cols ))

maximum_pixel_value=0

# Nested for loop that turns rows into columns and columns into rows. Gets
original matrix and copies values into new numpy matrix but rows are
columns and columns are rows.
# This nested for loop also checks for maximum pixel value
for row in range(rows):
    for col in range(cols):
        rotated_pixels[col, -row-1] = im_gray_pixels[row,col]

        if rotated_pixels[col, -row-1] > maximum_pixel_value:
            maximum_pixel_value = rotated_pixels[col, -row-1]

#Turns pixel values into image format
rotated_img = Image.fromarray(np.uint8(rotated_pixels))
rotated_img.show()

#Saves image
```

```
rotated_img.save("Rotated_Img.jpg")

print("Maximum Pixel Value is: "+str(maximum_pixel_value))
```

Task2.py Code

```
from PIL import Image, ImageOps
import numpy as np
from numpy import asarray
import MyImageFunctions


#Takes in image and gets matrix of pixel values
im = Image.open("Watertower.tif")
im_pixels = asarray(im)

#Function takes in matrix of pixel values
MyImageFunctions.myImageInverse(im_pixels)
```

MyImageFunctions.py Code

```
from PIL import Image, ImageOps
import numpy as np
from numpy import asarray

def myImageInverse(im_pixels):
    # This function helps take in a matrix of a grayscale image's pixel
values.
    # This then outputs an inverted image of the original image by making
each pixel value in the matrix subtract by 255 and creating a new numpy
matrix for the new inverted image.
    # For example, output_value = 255 - input_value
    rows, cols = im_pixels.shape
    inverse_pixels = np.zeros(shape=(rows,cols))
    maximum_pixel_value = 0

    # Nested for loop that goes through the whole original matrix
    # and copies its value into a new matrix but also subtracts 255 to the
original matrix pixel values.
    # Checks for maximum pixel value
```

```
    for x in range(rows):
        for y in range(cols):
            inverse_pixels[x][y] = (255 - im_pixels[x][y])

            if(inverse_pixels[x][y] > maximum_pixel_value):
                maximum_pixel_value = inverse_pixels[x][y]

    #Converts matrix value into image
    inverse_image = Image.fromarray(np.uint8(inverse_pixels))
    inverse_image.show()
    inverse_image.save("Inverse.jpg")
    print("Maximum pixel value is : "+ str(maximum_pixel_value))
```

Task3.py Code

```
from PIL import Image, ImageOps
import numpy as np
from numpy import asarray

# Sets dimensions
rows = 100
cols = 256
gradient_shape = np.zeros(shape=(rows,cols))

# Helps count for gradient pixel values
count=-1

# Helps with calculating average
countSum = 0
average = 0

# Nested for loop that goes that implements each row in one column to have
the same pixel value
for y in range(cols):
    count+=1
    for x in range(rows):
        gradient_shape[x][y] = count
        countSum+=gradient_shape[x][y]
```

```
# Converts matrix into image form
gradient_img = Image.fromarray(np.uint8(gradient_shape))
gradient_img.show()
gradient_img.save("Gradient.jpg")

# Calculates average pixel value
average=countSum/(rows*cols)
print("Average pixel value is: " + str(average))
```