Kenneth Esdaile March 28, 2019

Homework 1

CSCI 381/780 Image Processing

Queens College

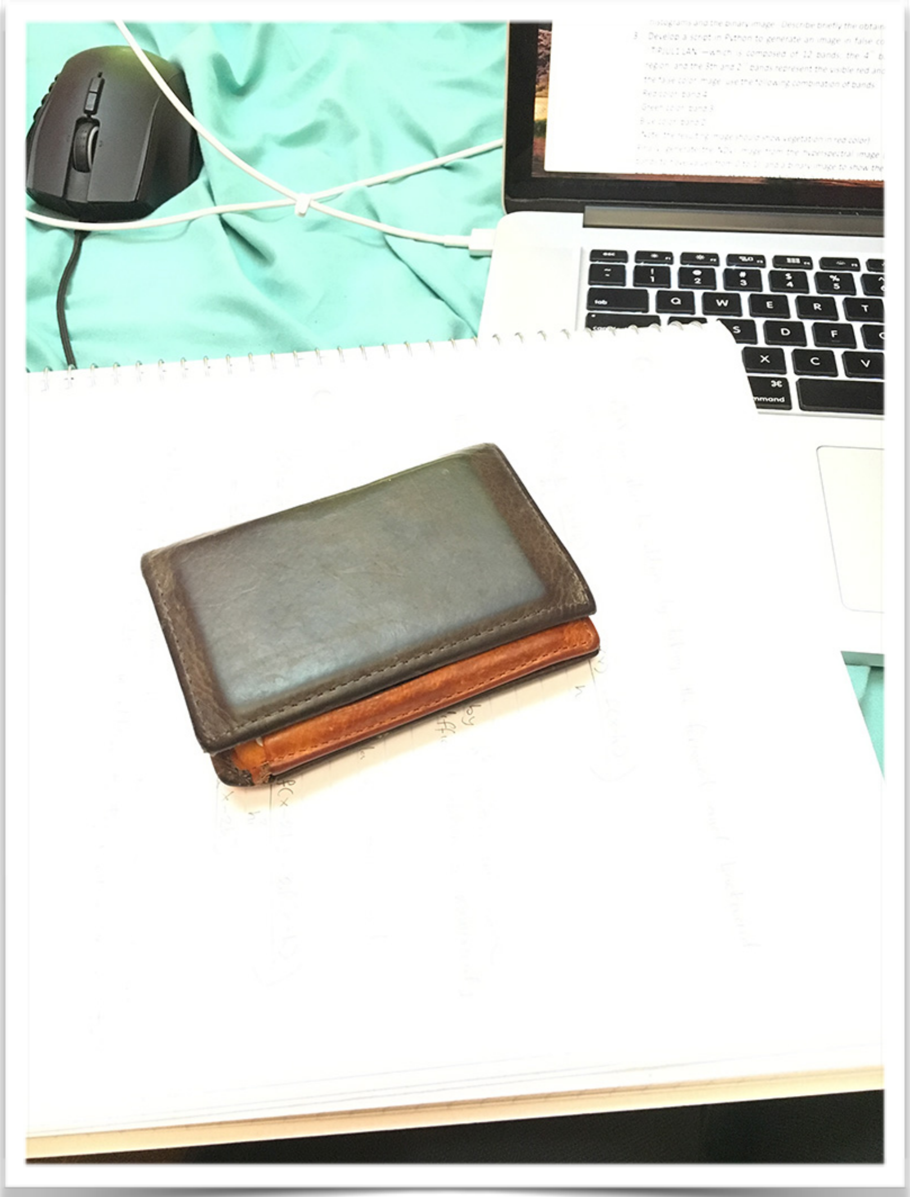
Department of Computer Science

Due Date: March 31st

Kenneth Esdaile March 28, 2019

Question 1: part 1

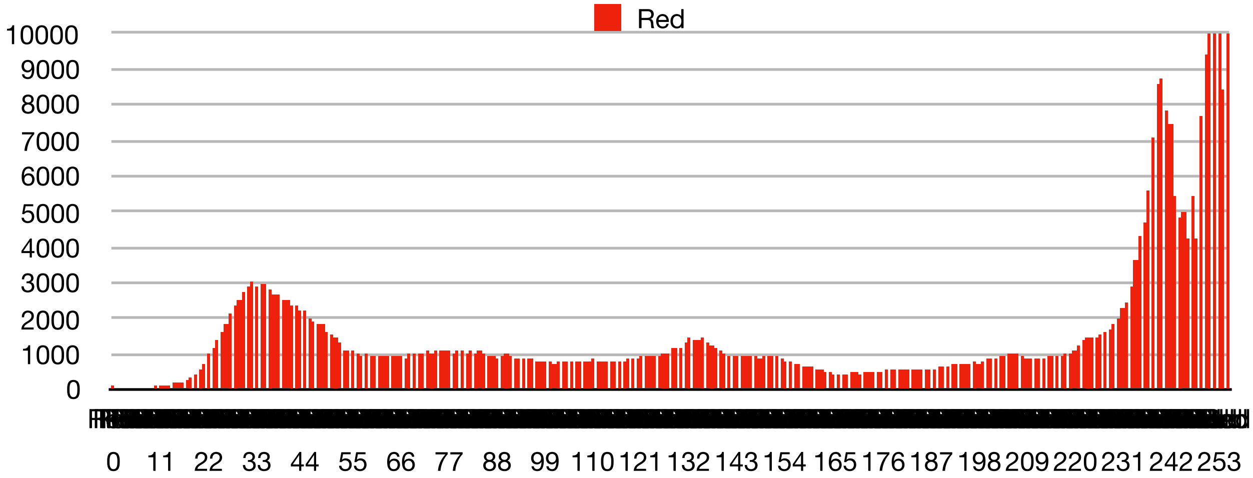
Capture two images (one underexposed, and one overexposed) using your cell phone or digital camera from the same scene. Use the developed program to plot the histograms from the captured images. Describe briefly the differences in the histograms.

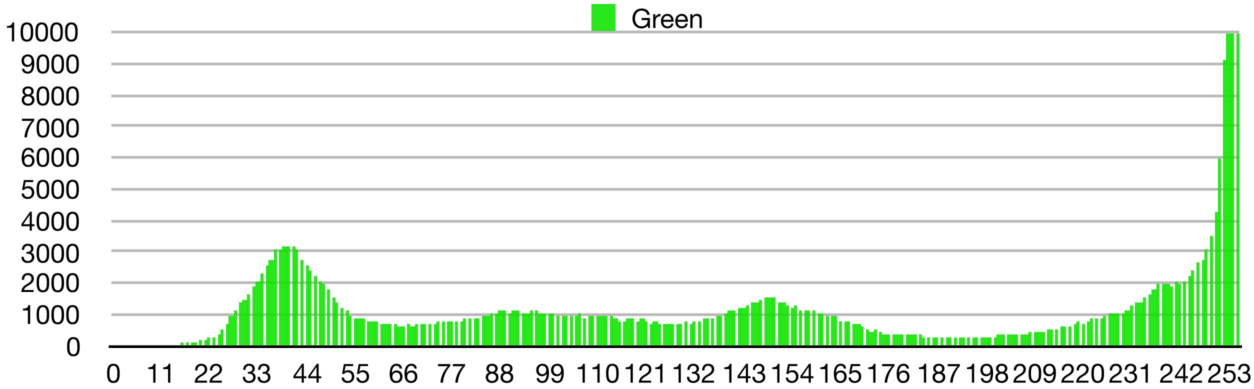


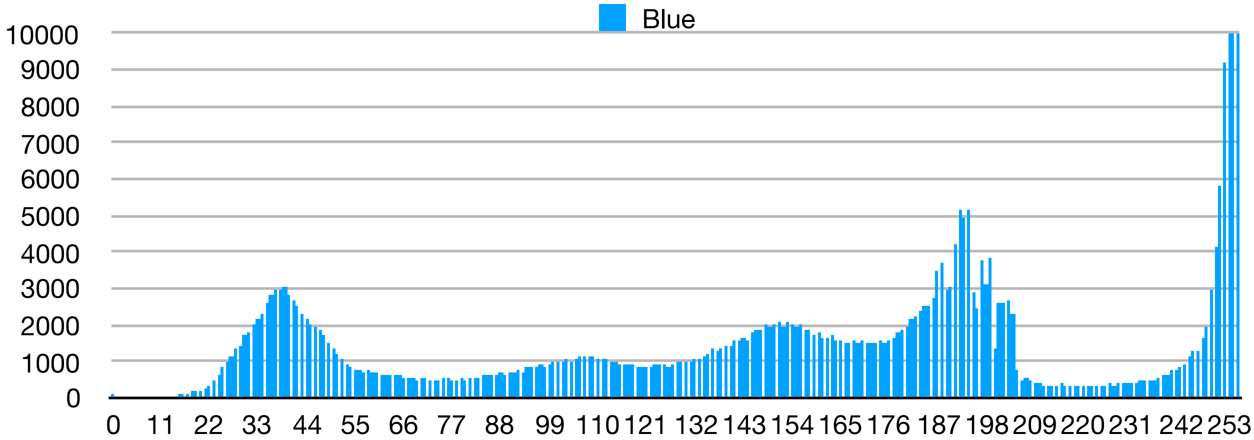
OVER-EXPOSED IMAGE

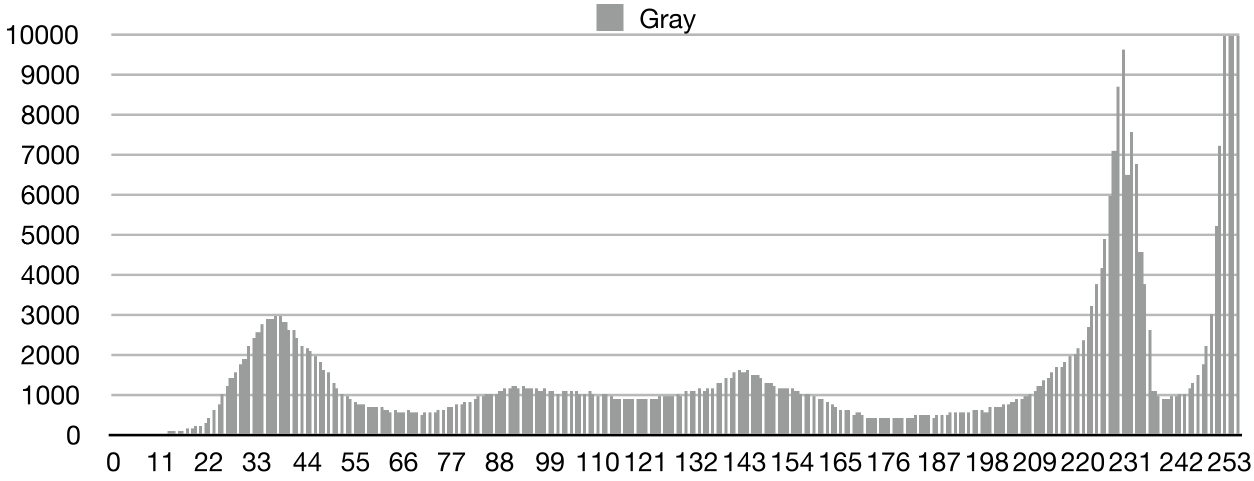
Kenneth Esdaile

March 28, 2019



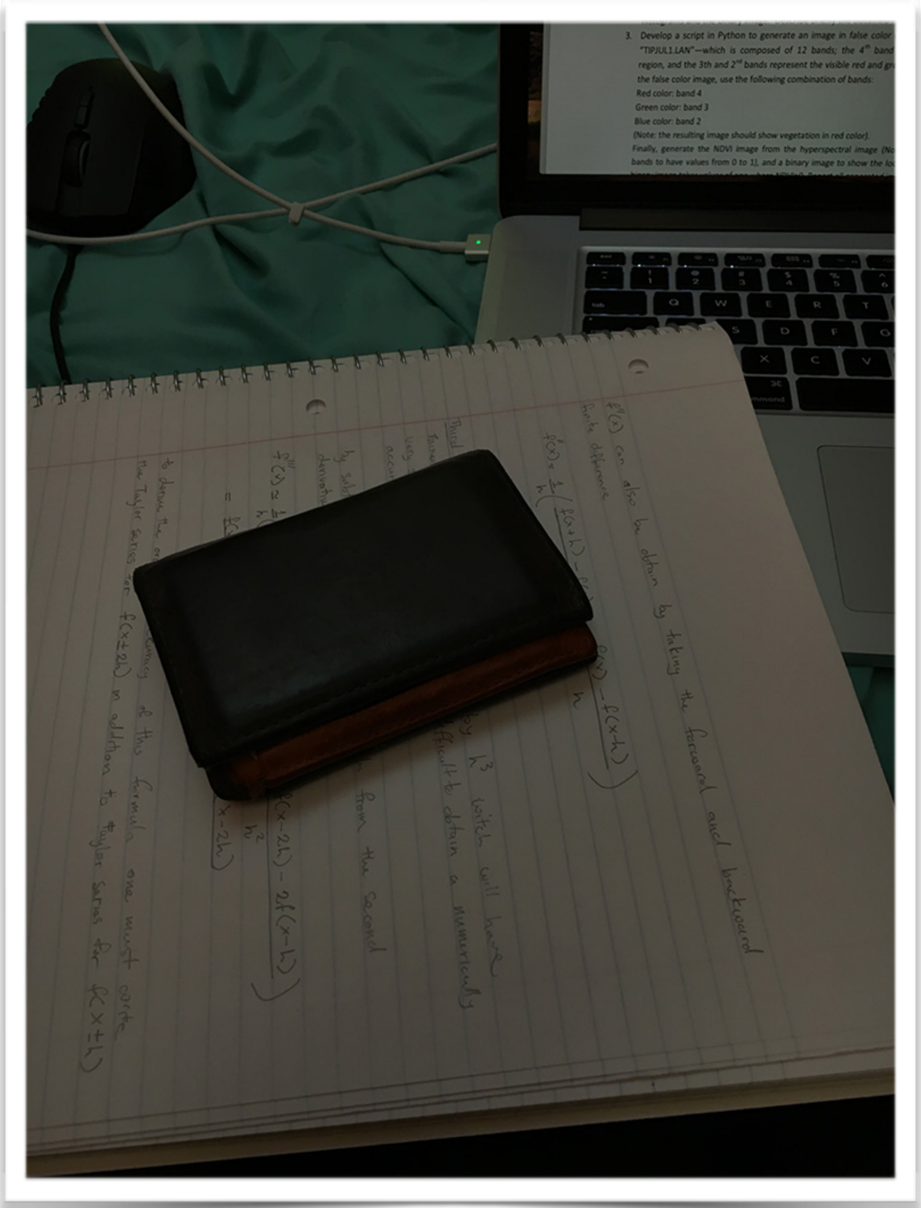






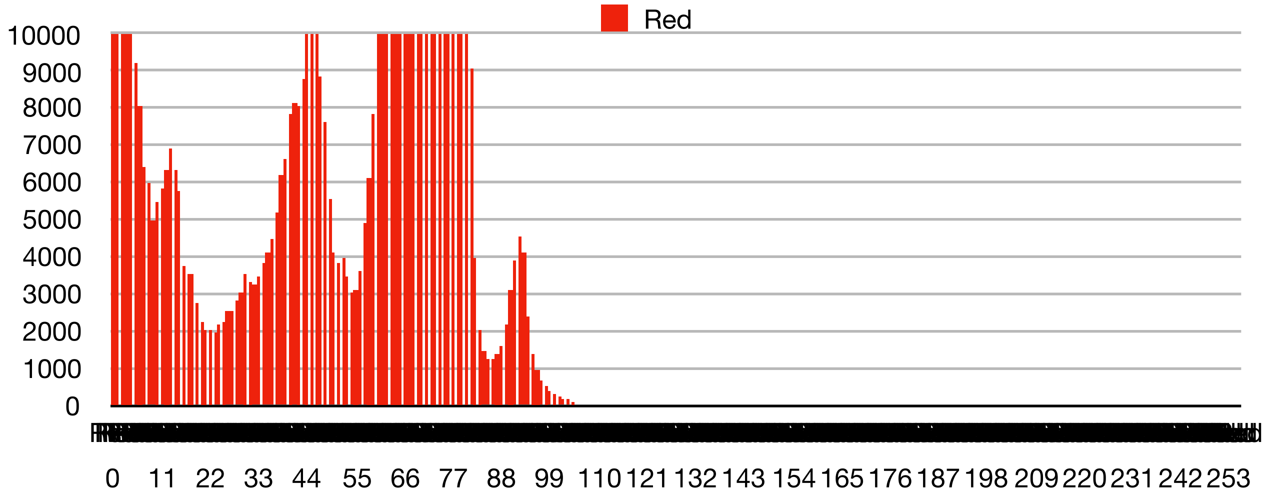
Kenneth Esdaile March 28, 2019

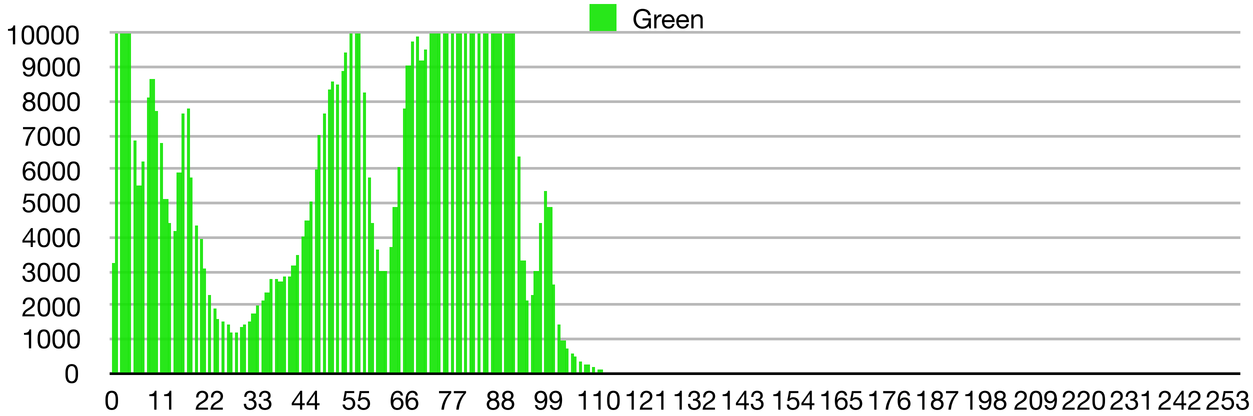
Question 1: part 2

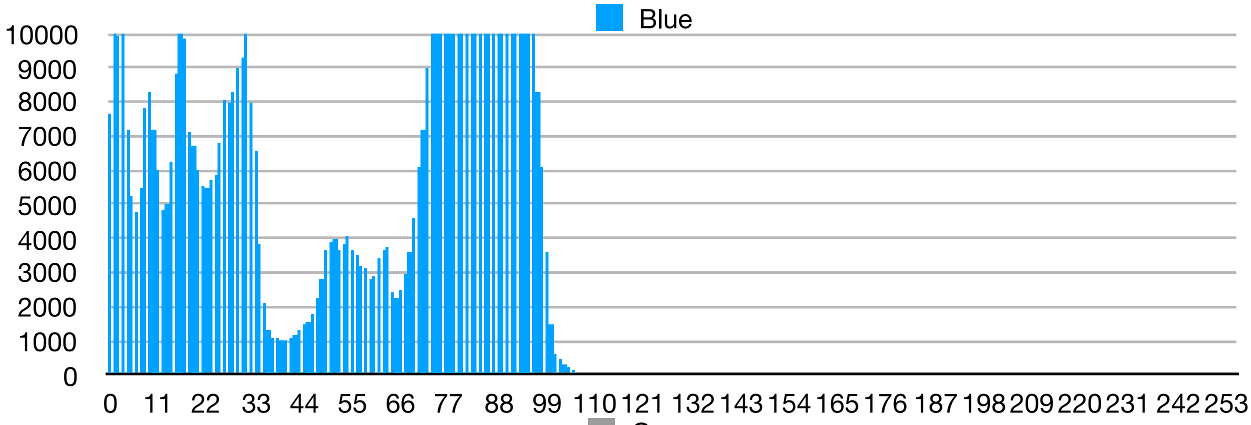

3. Develop a script in Python to generate an image in false color
"TIPJUL1.LAN"—which is composed of 12 bands; the 4th band
region, and the 3th and 2nd bands represent the visible red and gr
the false color image, use the following combination of bands:
Red color: band 4
Green color: band 3
Blue color: band 2
(Note: the resulting image should show vegetation in red color).
Finally, generate the NDVI image from the hyperspectral image (No
bands to have values from 0 to 1), and a binary image to show the lo

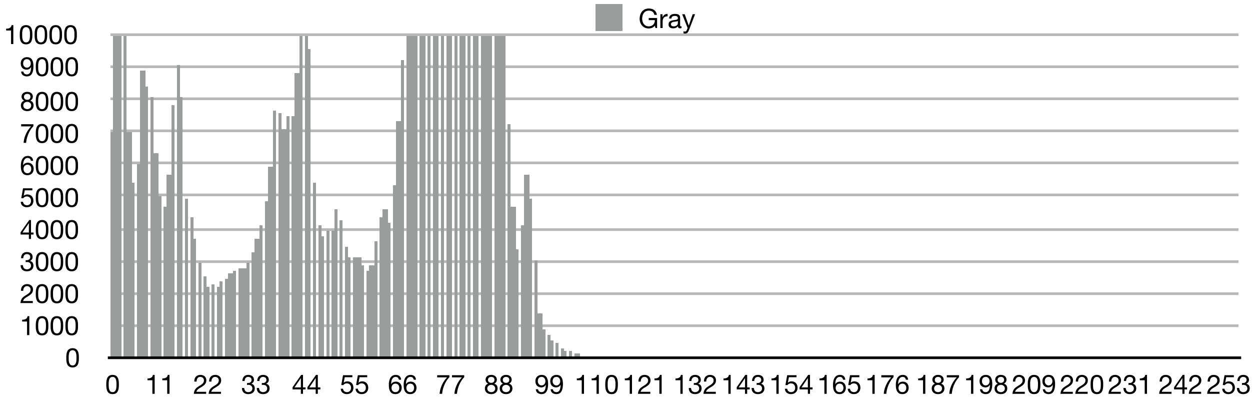
UNDER-EXPOSED IMAGE

Kenneth Esdaile March 28, 2019









Kenneth Esdaile March 28, 2019

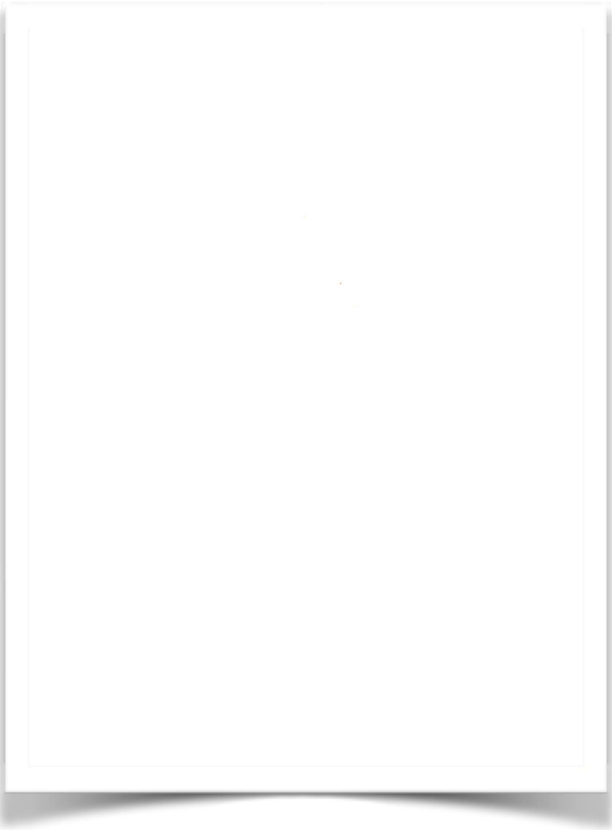
The histogram of this image is binary and not really interesting, since only two shades make the the image, black and white. No in betweens. At shade level 255 there is 151486 hits, which is the total number of pixels that the object of interest occupies in an image of 1024x768 pixels, thus 634946 pixels are black. 634946 pixels are the total number of pixel that make up the background.

Kenneth Esdaile March 28, 2019

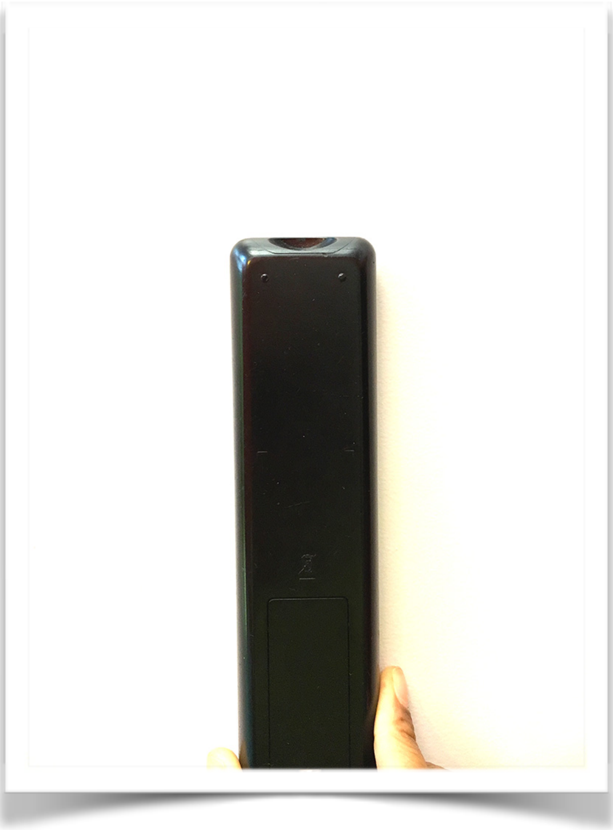
Question 2: part 1

Create a program to generate a binary image that identifies the object from the background--the pixels belonging to the object will take values of one and the pixels from background will take values of zero respectively. Describe briefly the obtained results.

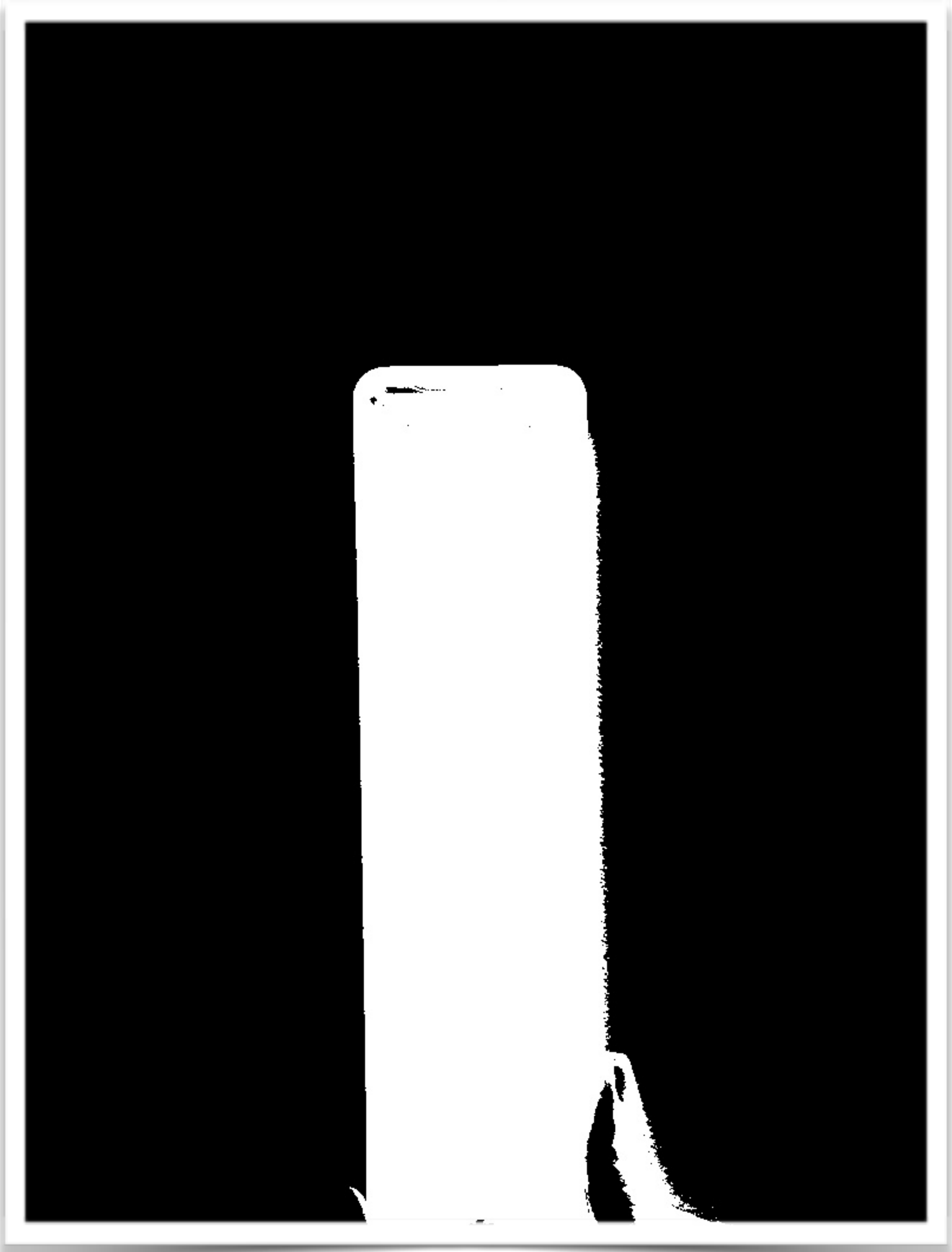
Background image



Foreground image



The resulting binary image from subtracting the Background and Foreground image. Total pixels that belong to the object of interest is: 151486

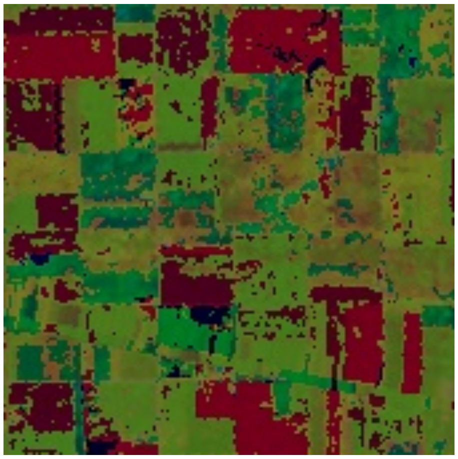


BINARY IMAGE GENERATED FROM THE BACKGROUND AND FOREGROUND IMAGE

Kenneth Esdaile March 28, 2019

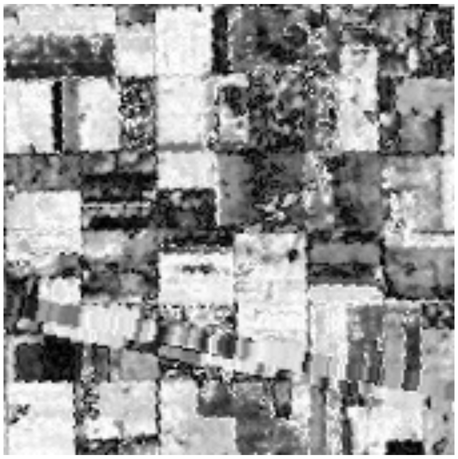
Question 3: part 1

Generate a false color image.



Question 3: part 2

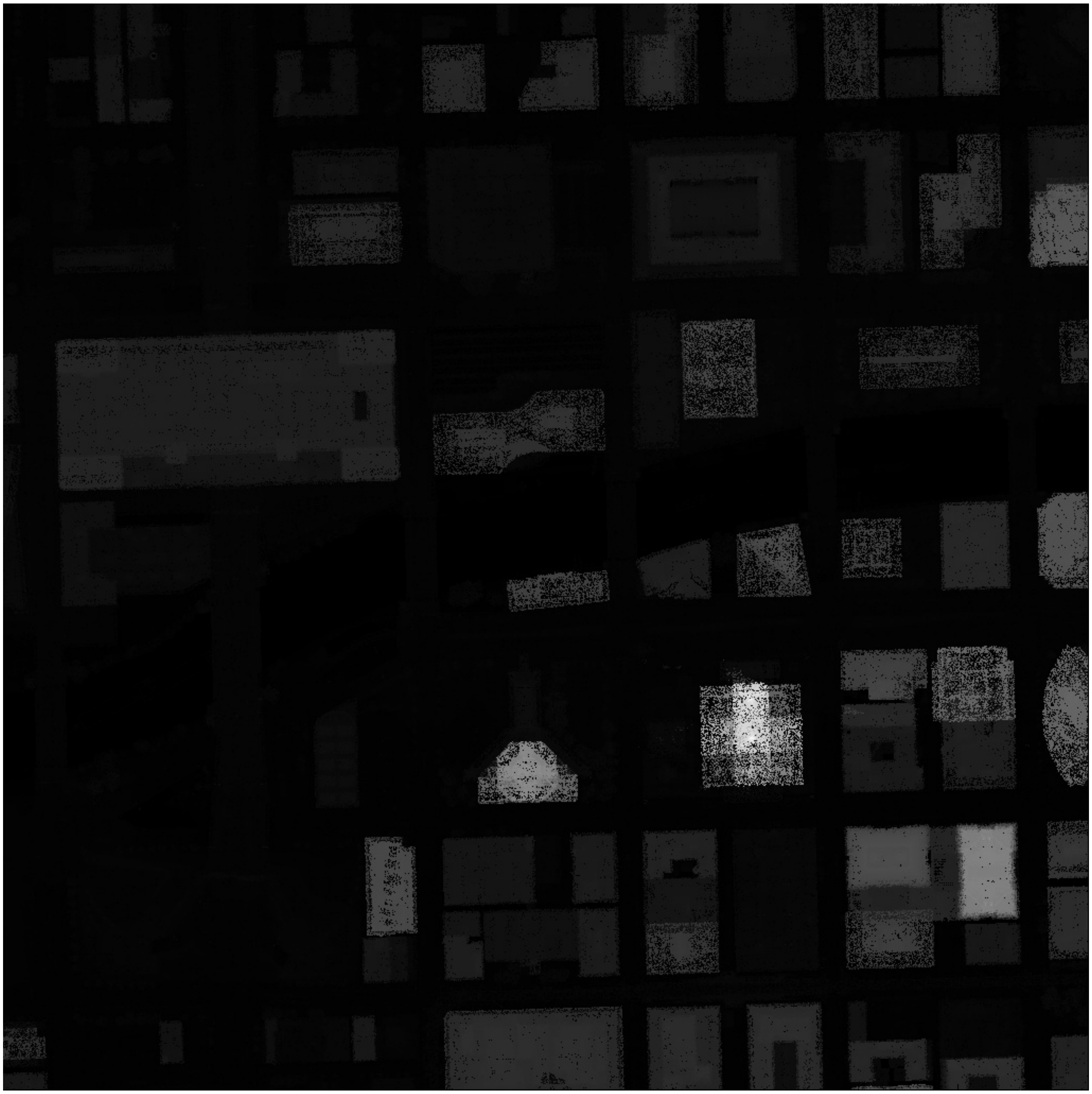
Generate the NDVI image from the hyper-spectral image.



Kenneth Esdaile March 28, 2019

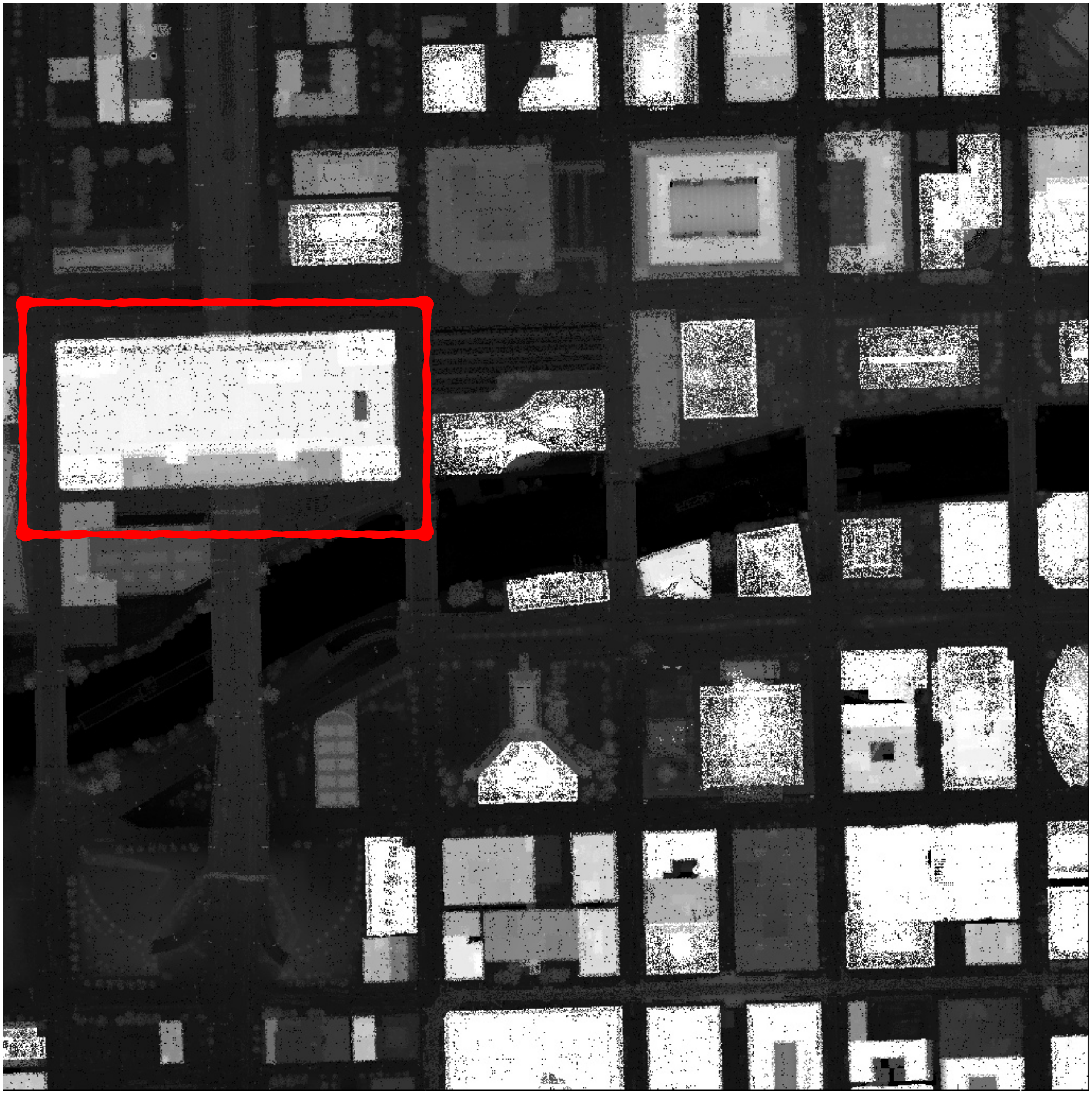
Question 4: part 1

Create from the Lidar point cloud “17258975.las” its corresponding raster



THE LIDAR IMAGE GENERATED FROM THE 17258975.LAS FILE

THE RESULTING IMAGE GENERATED FROM INCREASING THE EXPOSURE OF THE LIDAR IMAGE



Kenneth Esdaile March 28, 2019

Question 4: part 2

Applying logical operators, use a binary image to select one building from the raster (masking). Finally, develop an algorithm to calculate the area and altitude of the selected building.

Building Information	
altitude	294.8864117647060
length	4780.699842104950
width	1168.0902336181900
area	5584288.79542281
pixels	43916

KAE_CS381_HW1.py file

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Fri Apr  5 11:25:02 2019

@author: kanthonye
"""
import numpy as np
def GenerateHistogram(img):
    height = img.shape[1]
    width  = img.shape[0]

    r = 0;
    g = 1;
    b = 2;
    G = 3;
    histo = np.zeros([4, 256], int);

    for h in range(height):
        for w in range(width):
            gray = int((int(img[w][h][r]) + int(img[w][h][g]) + int(img[w][h][b])) / 3);
            histo[r][ img[w][h][r] ] += 1
            histo[g][ img[w][h][g] ] += 1
            histo[b][ img[w][h][b] ] += 1
            histo[G][ gray ] += 1

    return histo;

def GenerateFalseColorImage(img):
    height = img.shape[1]
    width  = img.shape[0]

    simg = np.zeros([width, height, 3], int)

    for h in range(height):
        for w in range(width):
            #print(img[w,h,1])
            nir = img[w,h,4]
            red = img[w,h,3]
            grn = img[w,h,2]

            if nir < 0:
                nir = -nir
            if red < 0:
                red = -red
            if grn < 0:
                grn = -grn

            simg[w,h,0] = nir
            simg[w,h,1] = red
            simg[w,h,2] = grn
    return simg



def GenerateBinaryImage(bgimg, fgimg, threshold):
    bg_height = bgimg.shape[1]
    bg_width  = bgimg.shape[0]
    fg_height = fgimg.shape[1]
    fg_width  = fgimg.shape[0]
    height = 1
    width  = 1

    if fg_height < bg_height:
        height = fg_height
    else:
        height = bg_height

    if fg_width < bg_width:
        width = fg_width
    else:
        width = bg_width

    r = 0;
    g = 1;
    b = 2;
    count = 0
    bimg = np.zeros([width, height, 3], int)

    for h in range(height):
        for w in range(width):

            R = int(fgimg[w][h][r]) - int(bgimg[w][h][r])
            G = int(fgimg[w][h][g]) - int(bgimg[w][h][g])
            B = int(fgimg[w][h][b]) - int(bgimg[w][h][b])
            u = int((R + G + B) / 3)

            if u <= threshold:
                bimg[w][h][r] = 0;
                bimg[w][h][g] = 0;
                bimg[w][h][b] = 0;
            else:
                bimg[w][h][r] = 255;
                bimg[w][h][g] = 255;
                bimg[w][h][b] = 255;
                count += 1
    return bimg, count
```

```
def GenerateNDVIImage(img):
    height = img.shape[1]
    width  = img.shape[0]

    simg = np.zeros([width, height, 3], int)

    for h in range(height):
        for w in range(width):
            nir = img[w,h,4]
            red = img[w,h,3]

            if nir < 0:
                nir = -nir
            if red < 0:
                red = -red

            g = (nir + red)

            if g != 0:
                g = (nir - red) / g

            g = (1.0 + g) * 0.5

            simg[w,h,0] = int(g * 255)
            simg[w,h,1] = int(g * 255)
            simg[w,h,2] = int(g * 255)
    return simg


def GenerateLidarImage(las, width, height):
    mmin = las.header.min
    mmax = las.header.max

    longitude = mmax[0] - mmin[0]
    latitude = mmax[1] - mmin[1]
    altitude = mmax[2] - mmin[2]

    simg = np.zeros([width+1, height+1, 3], int)

    for x, y, z, ite, c, nr, rn in np.nditer([las.x, las.y, las.z, las.Intensity, las.Classification,
las.num_returns, las.return_num]):
        lx = x - mmin[0]
        ly = y - mmin[1]
        nx = lx / longitude
        ny = ly / latitude
        nz = ((z - mmin[2]) / altitude)
        mx = int(nx * width)
        my = int(ny * height)
        simg[mx,my] = nz * 255;
    return simg
```

Kenneth Esdaile March 28, 2019

Question 1: part 1

```
from KAE_CS381_HW1 import GenerateHistogram
import cv2

img = cv2.imread('over-expose.jpg')
histo = GenerateHistogram(img)

for i in range(256):
    print(str(histo[3][i]))
```

Question 1: part 1

```
from KAE_CS381_HW1 import GenerateHistogram
import cv2

img = cv2.imread('under-expose.jpg')
histo = GenerateHistogram(img)

for i in range(256):
    print(str(histo[3][i]))
```

Question 2: part 1

```
from KAE_CS381_HW1 import GenerateBinaryImage
import cv2
import matplotlib.pyplot as plt

bgimg = cv2.imread('bgimg.jpg')
fgimg = cv2.imread('fgimg.jpg')

bimg, count = GenerateBinaryImage(fgimg, bgimg, 30)
print('pixels that belong to the object of interes -> ', count)
plt.imshow(bimg)

cv2.imwrite(‘binary-image.jpg',bimg)
```

Question 3: part 1

```
from KAE_CS381_HW1 import GenerateFalseColorImage
from spectral import open_image
import cv2

img = open_image('TIPJUL1.LAN')
print (img)

simg = GenerateFalseColorImage(img)
cv2.imwrite(‘false-img.jpg’,simg)
```

Question 3: part 2

```
from KAE_CS381_HW1 import GenerateNDVIImage
from spectral import open_image
import cv2

img = open_image('TIPJUL1.LAN')
print (img)

simg = GenerateNDVIImage(img)
cv2.imwrite(‘NDVI.jpg',simg)
```

Question 4: part 1

```
from KAE_CS381_HW1 import GenerateLidarImage
from laspy.file import File
import cv2

las = File('17258975.las', mode="r")

height = 1023
width  = 1023
simg   = GenerateLidarImage(las, 1023, 1023)

cv2.imwrite('lidar-image.jpg', simg)
```

Question 4: part 2

```
import cv2
import numpy as np
from laspy.file import File

img = cv2.imread('lidar-image.jpg')
las = File('17258975.las', mode="r")
mmin = las.header.min
mmax = las.header.max

print("\n\n")
print("lidar-min ", mmin)
print("lidar-max ", mmax)
```

```
longitude = mmax[0] - mmin[0]
latitude = mmax[1] - mmin[1]
altitude = mmax[2] - mmin[2]

print("\n")
print("longitude ", longitude)
print("latitude  ", latitude)
print("altitude  ", altitude)

height = 170
width  = 350
qimg = np.zeros([height+1, width+1, 3], int)
position = [40,300]
y_end = position[1] + height
x_end = position[0] + width
total_pixel = 0
alti = 0
minx = longitude
miny = latitude
maxx = 0
maxy = 0
n = 0
for y in range(position[1], y_end):
    m = 0
    for x in range(position[0], x_end):
        q = img[y,x][0];
        if alti < q:
            alti = q

        if q >= 9:
            q = 1;
            if minx > x: minx = x
            if miny > y: miny = y
            if maxx < x: maxx = x
            if maxy < y: maxy = y
        else :
            q = 0;

        if q == 1:
            total_pixel = total_pixel + 1;

        qimg[n,m] = q * 255
        m = m+1;
    n = n+1;

alti = (alti / 255) * altitude
print("\n")
print("+---+------------------+")
print("|   |   min   | max    |")
print("+---+------------------+")
print("| x |  ", minx,"   |", maxx,"   |")
print("+---+------------------+")
print("| y |  ", miny,"  |", maxy,"   |")
print("+---+-------------------+")

length = ((maxx - minx) / qimg.shape[0]) * longitude;
width  = ((maxy - miny) / qimg.shape[1]) * latitude;
print("building altitude: ", alti)
print("building length  : ", length)
print("building width   : ", width)
print("building area    : ", length*width)
print("building pixels  : ", total_pixel)

cv2.imwrite('lidar-subimg-image.jpg', qimg)
```