CV Assignment 1 Report

Name: Soha Abd El Azim Abd El Azim
ID: 28-12512
Tut: 11

Q1:-

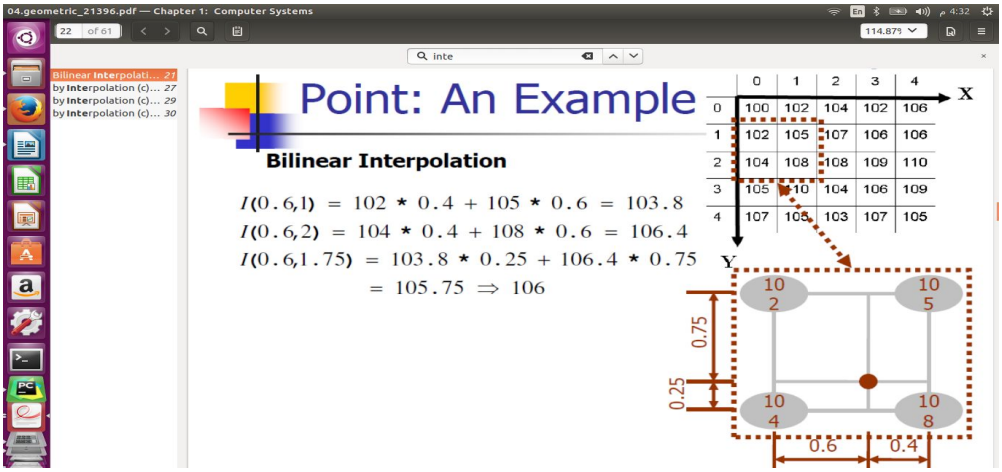
I loaded the image as binary scales image and looped over it pixel by pixel comparing its intensity . If the intensity > 127 => set it in the destination to 255 (white) otherwise I set it to zero (black) . At the end I display the result.

Q2:-

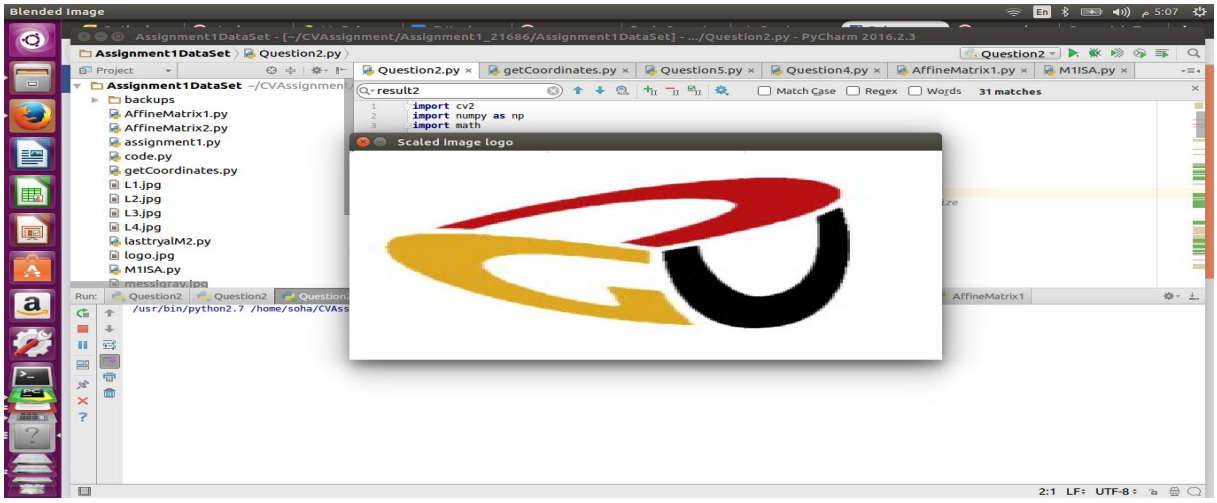
1-I load the two images: logo and L1
2-Scale image logo to be exact size like L1
-get the scaling ration in X dimension by dividing L1 rows/logo rows and same thing for the ratio in the y dimension
-create our matrix [[Sx,0],[0,Sy]]
-Get the inverse of the matrix
-create our result image with the same size as L1
-loop from the destination to the source (original logo image) and make bilinear interpolation to get accurate intensity.
Bilinear interpolation works as follows

a) Multiple the matrix we created with the coordinate of each point in the destination (resulted image)
b) We get the coordinate in the source image (logo) check if the coordinates we have are both integers then we will take the intensity of the pixel located at the resulted coordinates directly
c) If the coordinates contains fractions we make several checks to know which coordinate containing the fraction for example if i have point like 1.4 and 1.5 I have to make Interpolation 3 times . Just like we did in the lecture. I made a floor to get the exact pixel my point is located let's call them x,y. Then I did calculation between points (x,y) and (x,y-1) and interpolation between points (x-1,y) and (x-1,y-1) then I got an Interpolation between the results calculated from those 4 points.
   Note that:- I took care that the axis does not follow the convention we have in our course As in OpenCV the horizontal line represents the y axis and the vertical line represents the x axis.
d) Just a tryal to make the code efficient I checked if only one of the axis is fractional to make only one interpolation instead of 3 each time.

e)



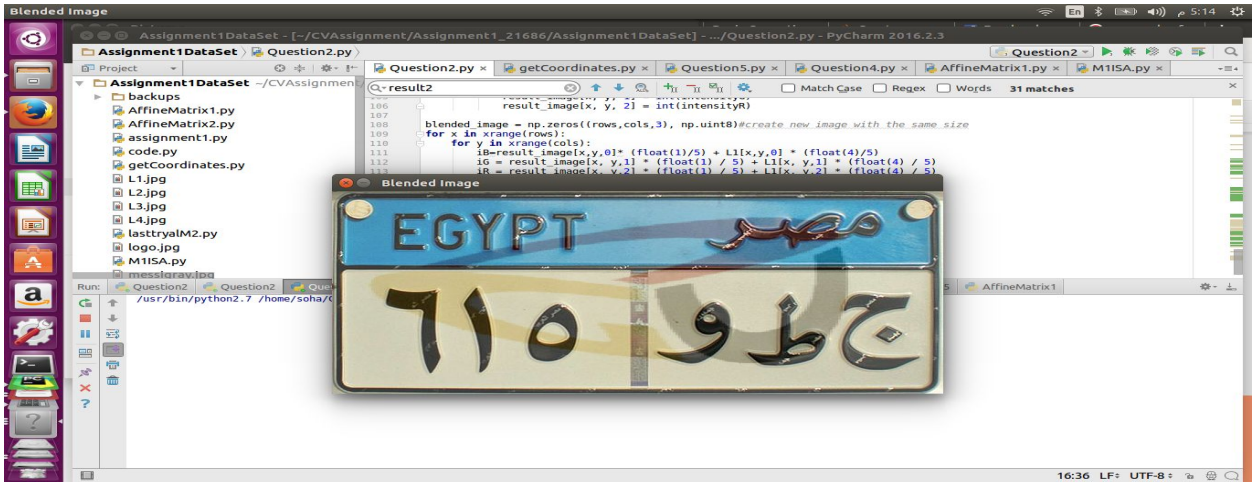
The resulted Logo image after Scaling



3-Blend L1 and L2

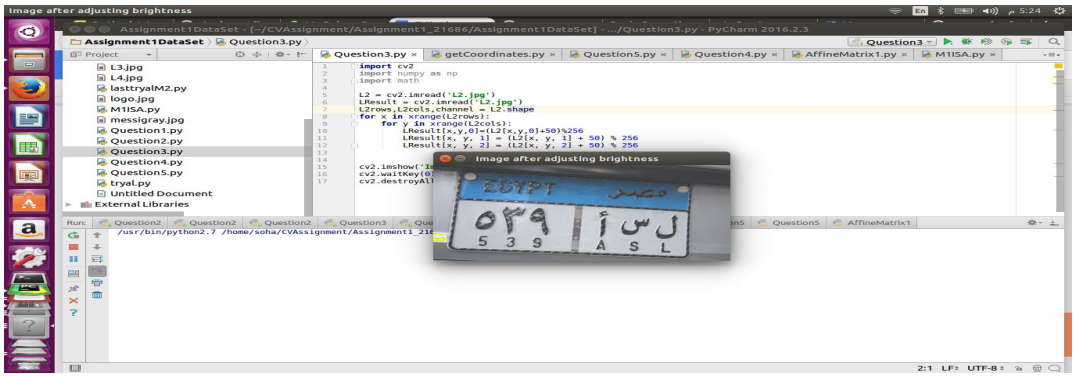
I looped over the L1 image and for each pixel I used the following equation

Resulted Image Intensity Component= L1(intensity component) *(4/5) + Scaled Logo(intensity component) *(1/5)



Q3:-

In order to adjust the brightness I looped over the image and increased the intensity of each of the RGB component with 50 and I used wrapping technique if the resulted value exceeded 255 (using the % operator)



Q4:-

1-Load Image L3

2-Created First Affine transformation

a)Get the inverse of the matrix containing 3 points forming a triangle in the original image [[x1,y1,1],[x2,y2,1],[x3,y3,1]] =[[2, 5, 1], [694, 900, 1], [467, 68, 1]]

b) multiple the resulted matrix we just formed with the a matrix containing all xs components of the 3 destination points [[x1dash],[x2dash],[x3dash]] to get first row in our affine matrix

c) repeat the previous states but with the y components to get the 2nd row of our affine matrix . and repeat another time but multiply with vector [[1],[1],[1]] to get the 3rd row

d)concatenate those 3 rows to form our affine matrix
e)Get the inverse of our affine as we will loop from the destination to the source
Note:- our destination points are (0,0) , (690,1000) , (690,0)

3-Repeat the previous steps to get the 2nd Affine matrix but with different source and destination points.
Source points: ([[10, 5, 1], [121, 970, 1], [694, 900, 1]])
Destination points:(0,0) , (0,1000) , (690,1000)

4-loops from destination to source and as we have two affine matrix we needed to figure out the equation of the diagonal (when will we switch from using Affine 1 to Affine 2)
-Using small equation (Cross multiplication) I figured out that the maximum y we can have will be equal to (500 * x) / 345 where X is the raw number ,500 is number of cols/2 and 345= raw/2
Note that (rows/2,cols/2) is the center of the triangle diagonal.

Q5:-

Used OpenCv method to apply homography transformation using 4 points in the source (were determined using mouse click method ([53.0,53.0],[188.0,20.0],[186.0,147.0],[56.0,185.0]) and 4 points of the destination (coordinates of the resulted image)

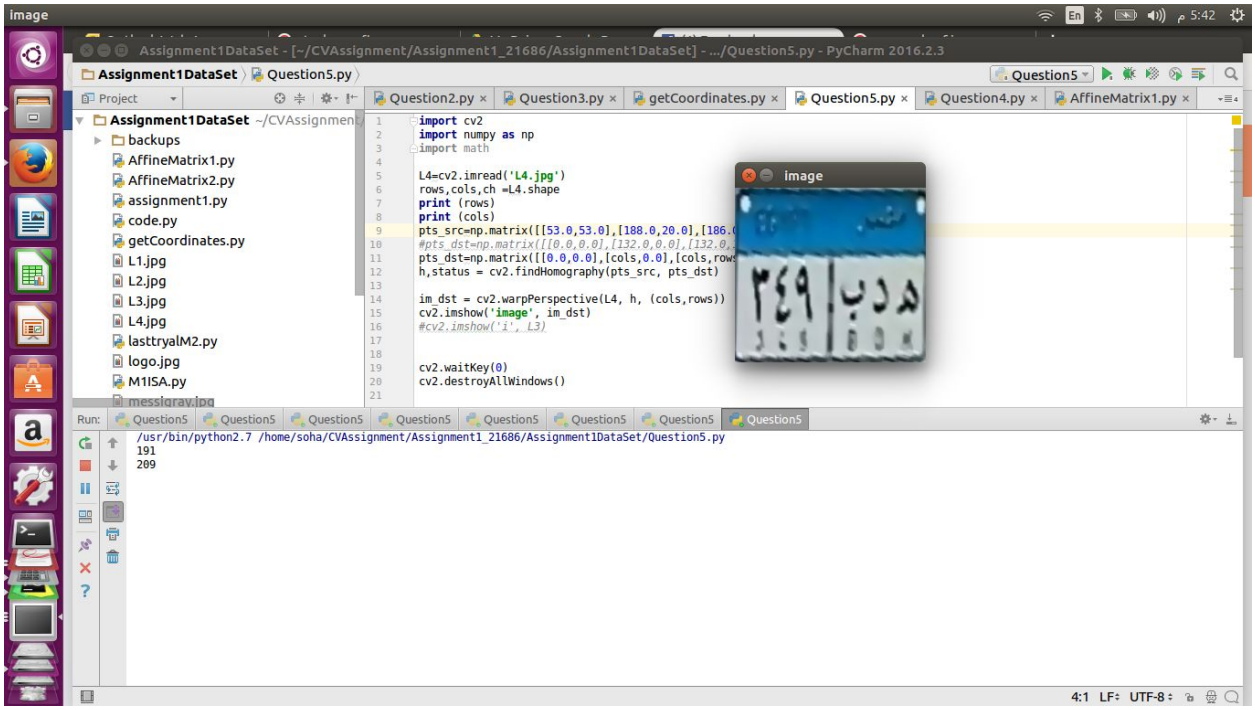


image
Assignment1DataSet - [~/CVAssignment/Assignment1_21686/Assignment1DataSet] - .../Question5.py - PyCharm 2016.2.3
Assignment1DataSet
Question5.py
Question5
Project
Question2.py
Question3.py
getCoordinates.py
Question5.py
Question4.py
AffineMatrix1.py
Assignment1DataSet ~/CVAssignment
backups
AffineMatrix1.py
AffineMatrix2.py
assignment1.py
code.py
getCoordinates.py
L1.jpg
L2.jpg
L3.jpg
L4.jpg
lasttryalM2.py
logo.jpg
M1ISA.py
messigray.jpg
import cv2
import numpy as np
import math
L4=cv2.imread('L4.jpg')
rows,cols,ch =L4.shape
print (rows)
print (cols)
pts_src=np.matrix([[53.0,53.0],[188.0,20.0],[186.
#pts_dst=np.matrix([[0.0,0.0],[132.0,0.0],[132.0,
pts_dst=np.matrix([[0.0,0.0],[cols,0.0],[cols,rows
h,status = cv2.findHomography(pts_src, pts_dst)
im_dst = cv2.warpPerspective(L4, h, (cols,rows))
cv2.imshow('image', im_dst)
#cv2.imshow('i', L3)
cv2.waitKey(0)
cv2.destroyAllWindows()
image
Run:
Question5
/usr/bin/python2.7 /home/soha/CVAssignment/Assignment1_21686/Assignment1DataSet/Question5.py
191
209
4:1
LF
UTF-8