Cs512- Computer Vision

Kinjal Kachi

Hawk id: A20449343

Abstract

This report gives a brief description of the programming part of Assignment 2. The Assignment had questions to do basic computer vision operations on the image like to convert the colored image to grey scale, find gradients in x and y direction, plot gradient vectors and so on. Along with using the Python and OpenCV libraries the Assignment 2 expected us to implement user defined functions.

Problem Statement:

Learn about the Computer Vision operations such as smoothening, image down-sampling, rotation, finding gradients and gradient vectors, see the effects of normalization.

Problem Solution:

Used the python based OpenCV library for Computer Vision,
NumPy, spicy, matplotlib to perform the above-mentioned operations.

Implementation details:

Capture, display and reload, save Image:

Capture Image from command line or capture image from desktop camera. Use cv2 library for the same.

Convert Image to Grayscale using pre-defined library:

Use function cv2.cvtColor(image, cv2.COLOR_BGR2GRAY) to do this.

Convert Image to Grayscale using user defined logic:

Use np.average() function to average using weights=[0.114, 0.587, 0.299] to get the grayscale image. The motivation for using these weights comes from the fact the image is broken down using the following formula:

$$I' = 0.229 \times I_r + 0.587 \times I_g + 0.114 \times I_b$$

Cycle the Colors Blue, Red, Green image in a continuous loop:

Extract blue components in the image using: image[:, :, 1:]
Extract green components in the image using: image[:, :, (0,2)]
Extract red components in the image using: image[:,:,:2]

Smoothen the image using pre-defined library:

Convert the image to grayscale image
Set up a kernel to smoothen the image using: np.ones((n,n),np.float32)/(n*n)

A Kernel tells you how to change the value of any given pixel by combining it with different amounts of the neighboring pixels.
Use function cv2.filter2D() and the defined kernel to smoothen the grayscale image.

Smoothen the image using user defined library:

Convert the image to greyscale image.
Use Gaussian filter to smoothen the image. Gaussian filter. A Gaussian filter is a linear filter. It's usually used to blur the image or to reduce noise.

Down sample the image with smoothing:

Define a kernel and use cv2.filter function to smoothen the image.
Downsample using cv2.resize() function.

Down sample the image without smoothing:

Simply downsample using cv2.resize() function.

Perform convolution with X and Y derivative of an image:

Convert the image to grayscale.
Use Sobel filter. Set the parameter (1,0) to take x derivative and (0,1) to take y derivative.
Normalize the image using cv2.normalize(). Normalization is used to bring the image into a range of intensity values that are physically less stressful to our (visual) sense.

Show the magnitude of the gradient normalized to the range [0, 255].

Compute of the gradients of the image using Sobel filters. Find the magnitude using cv2. cartToPolar() function.

Rotate the image to any angle between 0 to 360 degrees:

Get the dimensions of the image.i.e. number of rows and columns.
Use the functions: cv2.getRotationMatrix2D() and cv2.warpAffine() to rotate the image to the desired angle.

Convert the image to grayscale and plot the gradient vector of the image every N pixel and let the plotted gradient vector have a length of K.

Referred to the code on the link below:

https://stackoverflow.com/questions/32907838/plotting-a-gradient-vector-field-in-opencv

The other approaches were to combine cv2 library and quiver to solve this problem, but I didn't get desired results.

Results and Discussions:

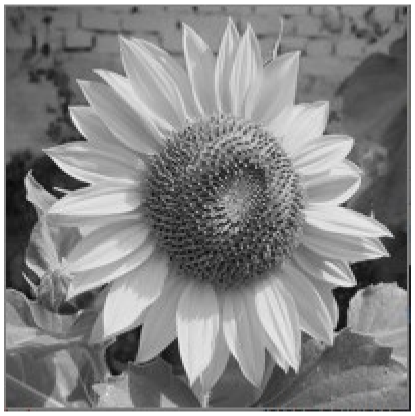
Original Image:

Dimensions of the original image are: 200X200 pixels

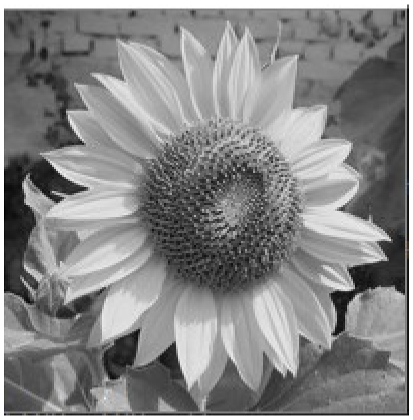
Picture Captured from the desktop camera when any image is not passed explicitly while running the program on command prompt.

Dimensions: 480X640 pixels

Option g: Grayscale Image with python library implementation:



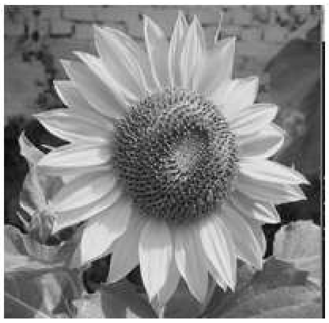
Option G: Grayscale Image using User defined function:



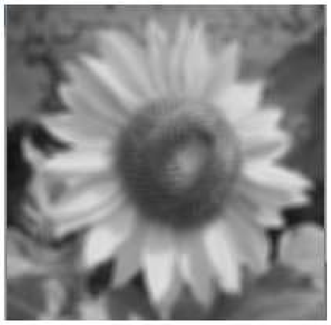
Option s: Gray scaled and smoothened image using the openCV function:

Smoothening slider scale: 0-15

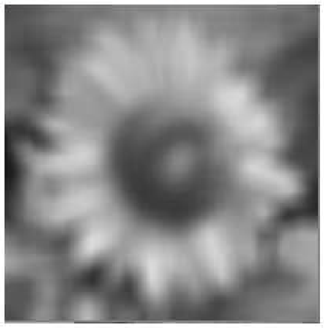
Scale level:0



Scale level:6



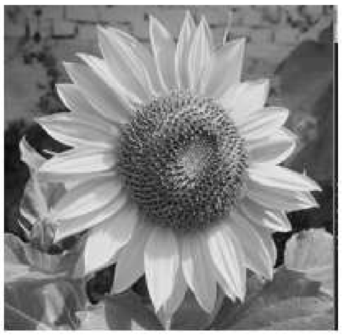
Scale level:15



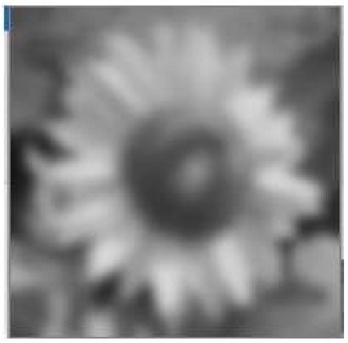
Option s: Gray scaled and smoothened image using the user defined function:

Smoothening slider scale: 0-10

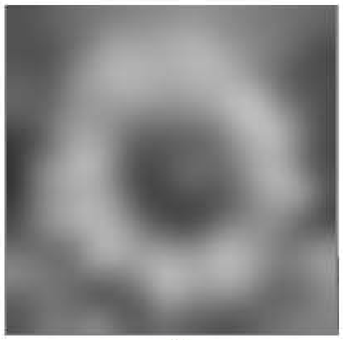
Scale level:0



Scale level: 5



Scale level:10

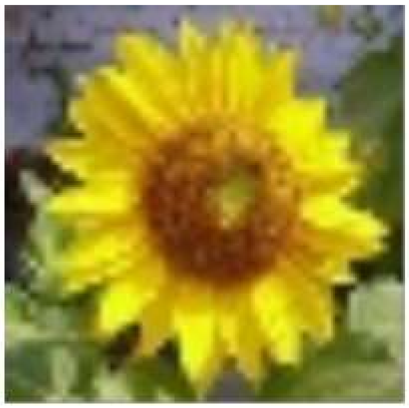


Option d and D: Down sampled Image

Dimensions: 50X50 pixels

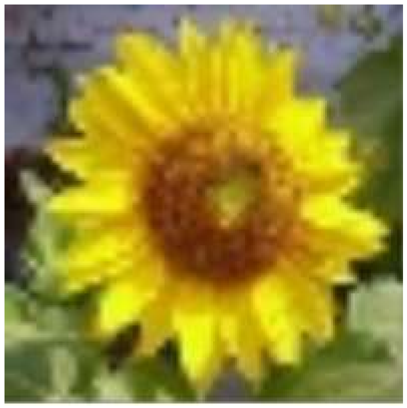
Without smoothening.

The enlarged image appears to be pixelated but not much smooth.



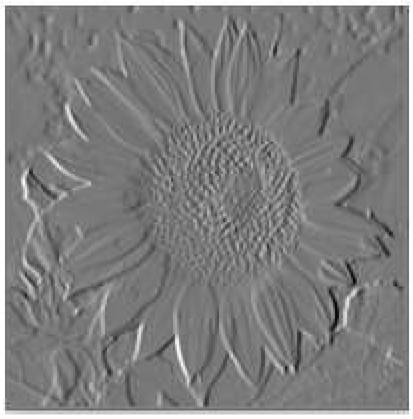
With smoothening.

When enlarged the image appears to be more blur.

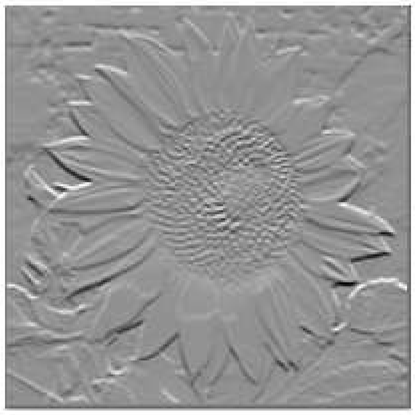


Option x and y:

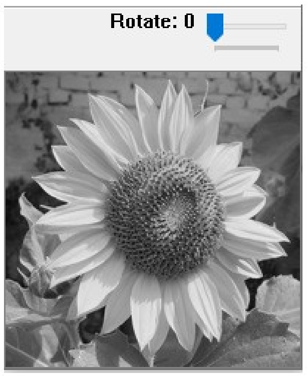
X Gradient

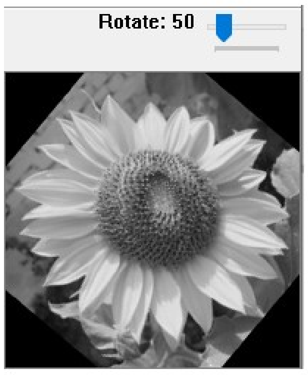


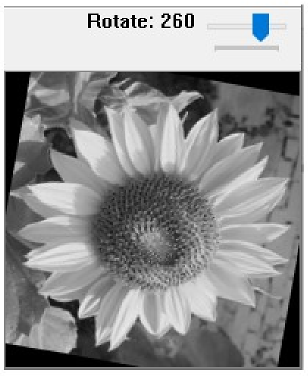
Y Gradient



Option r: Rotation in desired degree:







Option p:

Gradient 30