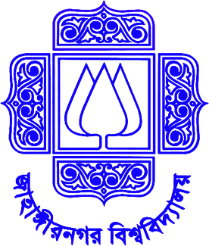
Institute of Information Technology

Jahangirnagar University

Savar, Dhaka-1342



Lab Manual

Course Code: ICT-4202

Course Title: Digital Image Processing Lab

**Lab No.: 2**

**Lab Title: INTRODUCTION TO IMAGE PROCESSING USING PYTHON**

Prepared by

**Mehrin Anannya**

Assistant Professor

Institute of Information Technology

Jahangirnagar University

**Lab Title: INTRODUCTION TO IMAGE PROCESSING USING PYTHON**

**OBJECTIVE**: To introduce students with Image processing basics and configure python platform in Anaconda.

**Lab Contents:**

* Computer Vision and Image Processing
* Download and install Anaconda
* Install OpenCV, Numpy and Matplotlib libraries in Anaconda
* Reading an image using opencv and matlpotlib
* Show the image in GUI window and plot it
* Show pixel information
* Gray scale image representation
* Writing an image in same and different directory

**Theory with Hands on Practice:**

Image processing and Computer Vision both are very exciting field of Computer Science.

Computer Vision:

In Computer Vision, computers or machines are made to gain high-level understanding from the input digital images or videos with the purpose of automating tasks that the human visual system can do. It uses many techniques and Image Processing is just one of them.

Image Processing:

Image Processing is the field of enhancing the images by tuning many parameter and features of the images. So, Image Processing is the subset of Computer Vision. Here, transformations are applied to an input image and the resultant output image is returned. Some of these transformations are- sharpening, smoothing, stretching etc.

Channels:

How many numbers are used to specify the color of each pixel is the number of channels each pixel has. In RGB as described above, an image has three numbers for each pixel that directly correspond to the three R, G and B elements in the computer display. Such RGB images have three channels.



Grayscale image:

Grayscale images are single-channeled images in which each pixel carries only information about the intensity of light. These images are exclusively made up of shades of gray.

Grayscale images should not be confused with black and white images (binary images) which contain only black and white pixels. In binary images, either a pixel is black or it is white. They have no colours in between. But Greyscale images have a wide range of shades of gray in their pixels.

RGB:

How many numbers we have per pixel is the number of channels that image has. A monochrome image that has one number per pixel has one channel. A more typical image that has three (R, G, B) numbers per pixel has three channels. Such images are called RGB images. Although images with three channels are most common because of the universal use of RGB formats in color

displays, there is no reason why an image could not contain more numbers per pixel for a greater number of channels than three.

For example, images used in sophisticated graphics editing use four channels: three channels for RGB plus an extra alpha or "a" channel. The alpha number says how transparent that particular pixel is supposed to be when it is combined in layers with other images. Such images are called RGBa images. Images coded with four, RGBa, channels can have regions of the image that are semi-transparent, so items in lower layers can be seen to a greater or lesser degree. Ordinary RGB images can have invisible pixels but this is a simple ON/OFF effect for each pixel. RGBa images can have a different percent transparency for each individual pixel in the image. This is called pixel transparency.

Anaconda:

Anaconda software helps you create an environment for many different versions of Python and package versions. Anaconda is also used to install, remove, and upgrade packages in your project environments. Furthermore, you may use Anaconda to deploy any required project with a few mouse clicks.

Download and install Anaconda from <https://www.anaconda.com/download>

OpenCV:

In this article, we’ll try to open an image by using OpenCV (Open Source Computer Vision) library. Following types of files are supported in OpenCV library:

Windows bitmaps – \*.bmp, \*.dib

JPEG files – \*.jpeg, \*.jpg

Portable Network Graphics – \*.png

WebP – \*.webp

Sun rasters – \*.sr, \*.ras

TIFF files – \*.tiff, \*.tif

Raster and Vector geospatial data supported by GDAL(which is a translator library for raster and geospatial data formats)

To use the OpenCV library in python, we need to install these libraries as a prerequisite:

Numpy Library : The computer processes images in the form of a matrix for which NumPy is used and OpenCV uses it in the background.

OpenCV python : OpenCV library previously it was cv but the updated version is cv2. It is used to manipulate images and videos.

Matplotlib: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

Differences between OpenCV and Matplotlib:

OpenCv uses BGR color format and matplotlib uses RGB color format.

**To install these libraries, we need to run these pip commands in cmd:**

pip install opencv-python

pip install numpy

pip install matplotlib

The steps to read and display an image in OpenCV are:

1. Read an image using imread() function.

2. Create a GUI window and display image using imshow() function.

3. Use function waitkey(0) to hold the image window on the screen by the specified number of seconds, 0 means till the user closes it, it will hold GUI window on the screen.

4. Delete image window from the memory after displaying using destroyAllWindows() function.

Let’s start reading an image. using cv2.

To read the images cv2.imread() method is used. This method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.

Syntax: cv2.imread(path, flag)

Parameters:

path: A string representing the path of the image to be read.

flag: It specifies the way in which image should be read. It’s default value is cv2.IMREAD\_COLOR

[If set, always convert image to the 3 channel BGR color image.]

Return Value: This method returns an image that is loaded from the specified file.

Note:

The image should be in the working directory or a full path of image should be given.

By default, OpenCV stores colored images in BGR(Blue Green and Red) format.

All three types of flags are described below:

* cv2.IMREAD\_COLOR: It specifies to load a color image. Any transparency of image will be neglected. It is the default flag. Alternatively, we can pass integer value 1 for this flag.
* cv2.IMREAD\_GRAYSCALE: It specifies to load an image in grayscale mode. Alternatively, we can pass integer value 0 for this flag.
* cv2.IMREAD\_UNCHANGED: It specifies to load an image as such including alpha channel. Alternatively, we can pass integer value -1 for this flag.

Other flags are mentioned in the given link: [Other Flag Links](https://docs.opencv.org/4.x/d8/d6a/group__imgcodecs__flags.html#gga61d9b0126a3e57d9277ac48327799c80af660544735200cbe942eea09232eb822)

Below codes are implementations to read images and display images on the screen using OpenCV and matplotlib libraries functions.

**Python code to read image with OpenCV:**

**import** cv2

# To read image from disk, we use

# cv2.imread function, in below method,

img **=** cv2.imread("Flower.jpeg", cv2.IMREAD\_COLOR)

# Creating GUI window to display an image on screen

# first Parameter is windows title (should be in string format)

# Second Parameter is image array

# It doesn’t return anything.

cv2.imshow("image", img)

# To hold the window on screen, we use cv2.waitKey method

# Once it detected the close input, it will release the control

# To the next line

# First Parameter is for holding screen for specified milliseconds

# It should be positive integer. If 0 pass an parameter, then it will

# hold the screen until user close it.

cv2.waitKey(0)

# It is for removing/deleting created GUI window from screen

# and memory

cv2.destroyAllWindows()

**Python code to read image with Matplotlib:**

from matplotlib import pyplot as plt

import cv2

im = cv2.imread("Flower.jpeg")

#to convert bgr2rgb color format

color = cv2.cvtColor(im, cv2.COLOR\_BGR2RGB)

plt.imshow(color)

plt.title('Image')

plt.show()

#For showing it in window, however having some runtime error

#cv2.imshow("Flower\_Image", im)

**Pixel and Channel information:**

# shows width, height and channels information

img.shape

**Grayscale Image:**

**import** cv2

# path

# R Means ‘Raw String’

An ‘r’ before a string tells the Python interpreter to treat backslashes as a literal (raw) character. Normally, Python uses backslashes as escape characters. Prefacing the string definition with ‘r’ is a useful way to define a string where you need the backslash to be an actual backslash and not part of an escape code that means something else in the string.

path **=** r'Flower.jpeg'

# Using cv2.imread() method

# Using 0 to read image in grayscale mode

img **=** cv2.imread(path, cv2.IMREAD\_GRAYSCALE)

# Displaying the image

cv2.imshow('image', img)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Saving image in the current directory:**

cv2.imwrite() method is used to save an image to any storage device. This will save the image according to the specified format in current working directory.

Syntax: cv2.imwrite(filename, image)

Parameters:

filename: A string representing the file name. The filename must include image format like .jpg, .png, etc.

image: It is the image that is to be saved.

Return Value: It returns true if image is saved successfully.

**Tasks:**

**Task-1:** Save our university logo from website and read it in gray scale and as colour image. Show it in GUI. Then save it in different directory