National University
Of Computer and Emerging Sciences

Department of Computer Science

Faculty Member: <> **Date: <>**
Semester: <> **Group: <>**

Computer Vision

Lab 5: Spatial Filtering and Detection of Edges and Corners

Introduction

This laboratory exercise will focus on additional concepts pertaining to OpenCV. OpenCV is a popular and widely used library for image processing and computer vision applications. OpenCV contains a wide selection of functions for vision-based algorithms. These functions ranges from basic preprocessing such as blurring, edge-detection, thresholding etc to computer vision implementations such as image stitching, template matching and homogeneous transform etc.

Objectives

- Apply convolution operation for images
- Implement convolution of images with various filters
- Create and use the Gaussian filter
- Implement blurring/smoothing in images
- Detect edges from an image

- Detect corners from an image

Lab Conduct

- Respect faculty and peers through speech and actions
- The lab faculty will be available to assist the students. In case some aspect of the lab experiment is not understood, the students are advised to seek help from the faculty.
- In the tasks, there are commented lines such as #YOUR CODE STARTS HERE# where you have to provide the code. You must put the code between the #START and #END parts of these commented lines. Do NOT remove the commented lines.
- Use the tab key to provide the indentation in python.

Theory

OpenCV is a library that focuses on image processing and computer vision. An image is an array of colored square called pixels. Each pixel has a certain location in the array and color values in BGR format. By referring to the array indices, the individual pixels or a range of pixels can be accessed and modified. OpenCV provides many functions for centroid determination, color space changing, color range selection and perspective transformation.

A brief summary of the relevant keywords and functions in python is provided below. (For more details, check the slides for this lab)

print()	output text on console
input()	get input from user on console
range()	create a sequence of numbers
len()	gives the number of characters in a string
if	contains code that executes depending on a logical condition
else	connects with **if** and **elif**, executes when conditions are not met

elif	equivalent to **else if**
while	loops code as long as a condition is met
for	loops code through a sequence of items in an iterable object
break	exit loop immediately
continue	jump to the next iteration of the loop
def	used to define a function

In the lab, you will need to download at least 3 image files which you must use in the tasks. You will also need to convert those images into gray scale for the upcoming tasks.

Lab Task 1 – Convolution Operation _____

The convolution is a very important operation used in many areas. In image processing, a 2-D filter (kernel) can be convolved by placing the filter at the top-left of the image. The values of the filter that overlap with those of the image pixels are multiplied then added together to get an answer which is placed on an output array. Then, the filter is moved one step to the right and the process is repeated. Once the entire row is traversed, the window is moved one step down and starts again all the way from the left. This continues until the entire image is traversed.

In this task, you will write a function that takes as arguments: the input image and a 2-D filter. The function will return the output array of the convolution. Make use of NumPy arrays in your function. You will need to write the entire function from

scratch. Do NOT use any inbuilt convolution functions. Provide the code. In the next task, you will use this function.

TASK 1 CODE STARTS HERE
TASK 1 CODE ENDS HERE

Lab Task 2 – Averaging Filters _____

Create the following filters and convolve them (task 1) to get the output.

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

What do you observe when the filter size is increased? Apply different weighted filters on the same images and note down your observations. One such weighted filter is given below:

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Provide the code, all relevant screenshots and explanation of your observations below.

TASK 2 CODE STARTS HERE

TASK 2 CODE ENDS HERE

TASK 2 SCREENSHOT STARTS HERE

TASK 2 SCREENSHOT ENDS HERE

TASK 2 EXPLANATION STARTS HERE

TASK 2 EXPLANATION ENDS HERE

Lab Task 3 – Gaussian Filter _____

In this task, you will make a Gaussian filter from scratch. A 2-D Gaussian filter can be acquired using the given formula:

$$f[i,j] = \frac{1}{2\pi\sigma^2} e^{-\frac{(i^2+j^2)}{2\sigma^2}}$$

The σ is the standard deviation and is a parameter that changes the spread of the Gaussian filter. For this task, the size of the kernel along x and y can be roughly taken as d = 2πσ.

Using the above formula, create a function that implements the Gaussian filter. To display the Gaussian function, bring the values of the acquired Gaussian from 0 to 255 (this change is only to help visualize the Gaussian, do not use the 0-255 ranged Gaussian in the upcoming task). Also, ensure that the data type in the Gaussian is of type "uint8" other it may not be compatible with OpenCV functions. Use the imshow() function to display the Gaussian. For the submission, create 3 Gaussians of different σ values. Provide the code and all relevant screenshots.

TASK 3 CODE STARTS HERE

TASK 3 CODE ENDS HERE

TASK 3 SCREENSHOT STARTS HERE

TASK 3 SCREENSHOT ENDS HERE

Lab Task 4 – Gaussian Convolution _____

Using the functions you have written in tasks 1 and 3, convolve the images with a Gaussian filter (use a small σ value e.g. 1.5 if there are stack overflow errors). Provide the code and all screenshots showing the output.

TASK 4 CODE ENDS HERE

TASK 4 SCREENSHOT STARTS HERE

TASK 4 SCREENSHOT ENDS HERE

National University
Of Computer and Emerging Sciences

Lab Task 5 – Edge Map Filters _____

In this task, you will make a simple edge detector from scratch. Load the images and perform edge detection as follows:

- Convolve the image with a filter to get the horizontal edge matrix
- Convolve the image with another filter to get the vertical edge matrix
- Add the horizontal and vertical edge matrices to get the overall edge map

The following example shows the horizontal, vertical and combined edge maps respectively:

TASK 5 CODE STARTS HERE
TASK 5 CODE ENDS HERE

TASK 5 SCREENSHOT STARTS HERE

TASK 5 SCREENSHOT ENDS HERE

For the filters, you can use any type such as Sobel etc. Provide the code and all screenshots showing all 3 edge maps for each image.

Lab Task 6 – Gaussian Blur and Canny Edges _____

Load the house.jpg file for this task. Use the inbuilt Gaussian-blur function to blur at least one of the windows. Use 13x13 size kernel for the blur.

Additionally, you will need to apply Canny edge detection on the original house image. You must apply 2 of these edge detections; one on the original picture and second on a slightly blurred picture. Use same threshold values on both edge detections.

TASK 6 CODE STARTS HERE
TASK 6 CODE ENDS HERE

BLUR WINDOW SCREENSHOT STARTS HERE

BLUR WINDOW SCREENSHOT ENDS HERE

CANNY 1 SCREENSHOT STARTS HERE

CANNY 1 SCREENSHOT ENDS HERE

CANNY 2 SCREENSHOT STARTS HERE

CANNY 2 SCREENSHOT ENDS HERE

Lab Task 7 – Corner Detection _____

Load the images and use the inbuilt Harris corner detector function to highlight corner points in your images. Vary the parameters of the function (blockSize, kSize, k) and observe the changes. Provide the code and all relevant screenshots.

TASK 7 CODE STARTS HERE

TASK 7 SCREENSHOT ENDS HERE