National University
Of Computer and Emerging Sciences

Department of Computer Science

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

Computer Vision

Lab 4: Detection of Corner

Introduction

This laboratory exercise will focus on implementing Corner detection operators using OpenCV and NumPy. You will apply convolution operations to detect corners with Harris corner.

Objectives

- Understand and apply convolution Harris Corner Detector for image processing.

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

National University
Of Computer and Emerging Sciences

Lab Task 1 – Harris Corner Detector_____

Implement the Harris Corner Detector by completing the following steps:

(a) Compute the horizontal and vertical image derivatives (I_x, I_y) using Sobel or any suitable operator.

(b) Compute the product of the derivatives:

- $A= I_x^2$
- $B= I_y^2$
- $C= I_xI_y$

(c) Apply Gaussian smoothing (or any suitable blurring method) to each of the product images A, B, C to reduce noise.

(d) Compute the Harris corner response function:

- **Variant 1 (determinant–trace form, common in code)**
  $R=(AB-C^2) -k(A+B)^2$
  where K is a sensitivity constant (typically 0.04–0.06).
- **Variant 2 (eigenvalue form, common in theory):**
  $R=\lambda_1\lambda_2-k(\lambda_1+\lambda_2)^2$
  where λ_1, λ_2 are the eigenvalues of the second-moment matrix:

  $$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}$$

Note: Both formulas are mathematically equivalent because:

$\det(M) = AB - C^2 = \lambda_1\lambda_2$, $\quad\quad$ $\text{trace}(M)=A+B=\lambda_1+\lambda_2$

So, whether you compute R directly from A, B, C or from eigenvalues, the detected corners are the same.

(e) Apply thresholding on R to identify strong corner points and mark them on the original image.

National University
Of Computer and Emerging Sciences

Lab Task 2 –Shi Tomasi_____

Implement the **Shi–Tomasi corner detector** (also known as *Good Features to Track*) and compare the detected corners with those obtained using the Harris method. Explain why Shi–Tomasi often provides more stable corner points.

$$R=\min(\lambda_1, \lambda_2)$$