**Digital Image Processing**

**Title: Template Matching**

**Objectives:** The purpose of today’s lab is to introduce you to the process of template matching and edge detection. You will find objects in an image using Template Matching.

**Tools Used:** Python 3.10

**Procedure:** Open IDLE and perform the following tasks

**Task**

* + Write a template matching algorithm (function) in python.
  + It should be generic. This means any image and/or template can be given to it.
  + Find a crowd picture on the internet (should not be same as other class fellows). Make your template from it in photoshop or paint. Apply the template matching function on it. Draw a box around the window with closest 3 matches.
  + A sample program with output is shown below.

A screenshot of a video game

Description automatically generated with medium confidence Map

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

**Code:**

import cv2

import numpy as np

path = r'C:\Users\Naeem\Desktop\Jahanzeb\DIP\DIP Lab\Project 2\og.jpg'

img = cv2.imread(path, cv2.IMREAD\_COLOR)

img = cv2.resize(img, (0, 0), None, .50, .50)

path2 = r'C:\Users\Naeem\Desktop\Jahanzeb\DIP\DIP Lab\Project 2\mask.jpg'

temp = cv2.imread(path2, cv2.IMREAD\_COLOR)

temp = cv2.resize(temp, (0, 0), None, .50, .50)

gray = cv2.cvtColor(img, cv2.COLOR\_RGB2GRAY)

template = cv2.cvtColor(temp, cv2.COLOR\_RGB2GRAY)

height, width = template.shape

match = cv2.matchTemplate(gray, template, cv2.TM\_CCOEFF\_NORMED)

threshold = 0.39

loc = np.where(match >= threshold)

label = 1

for pt in zip(\*loc[::-1]):

cv2.rectangle(img, pt, (pt[0] + width, pt[1] + height), (0, 0, 255), 1)

cv2.putText(img, str(label), (pt[0], pt[1] - 5), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9, (0, 0, 255), 1)

if label < 3:

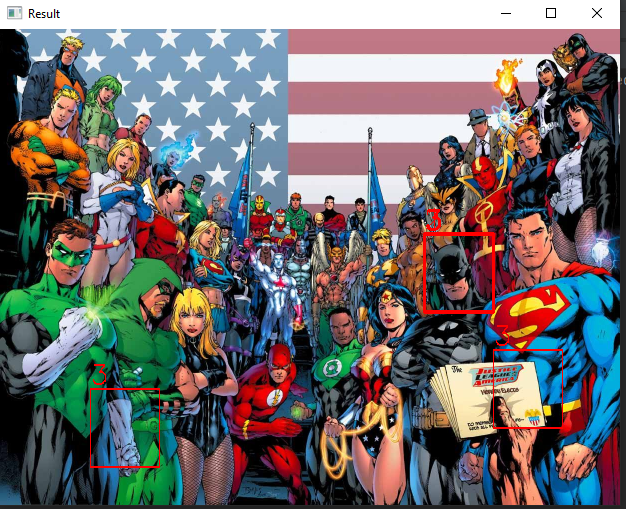
label = label + 1

cv2.imshow('Result', img)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Screenshot:**



****

Figure 1: Original Photo

****

Figure 2: Mask