**ЛАБОРАТОРНА РОБОТА № 8**

**ДОСЛІДЖЕННЯ МЕТОДІВ КОМП’ЮТЕРНОГО ЗОРУ**

***Мета:*** *використовуючи спеціалізовані бібліотеки та мову програмування Python навчитися обробляти зображення за допомогою бібліотеки OpenCV.*

**Хід роботи:**

**Завдання 1.** **Завантаження зображень та відео в OpenCV.**

Лістинг коду файлу LR\_8\_task\_1.py:

import cv2  
  
frameWidth = 640  
frameHeight = 480  
cap = cv2.VideoCapture(0)  
cap.set(3, frameWidth)  
cap.set(4, frameHeight)  
cap.set(10, 150)  
while True:  
 success, img = cap.read()  
 cv2.imshow("Result", img)  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
  
  
  
  
img = cv2.imread("kravchenko.jpg")  
cv2.imshow("kravchenko",img)  
cv2.waitKey(0)

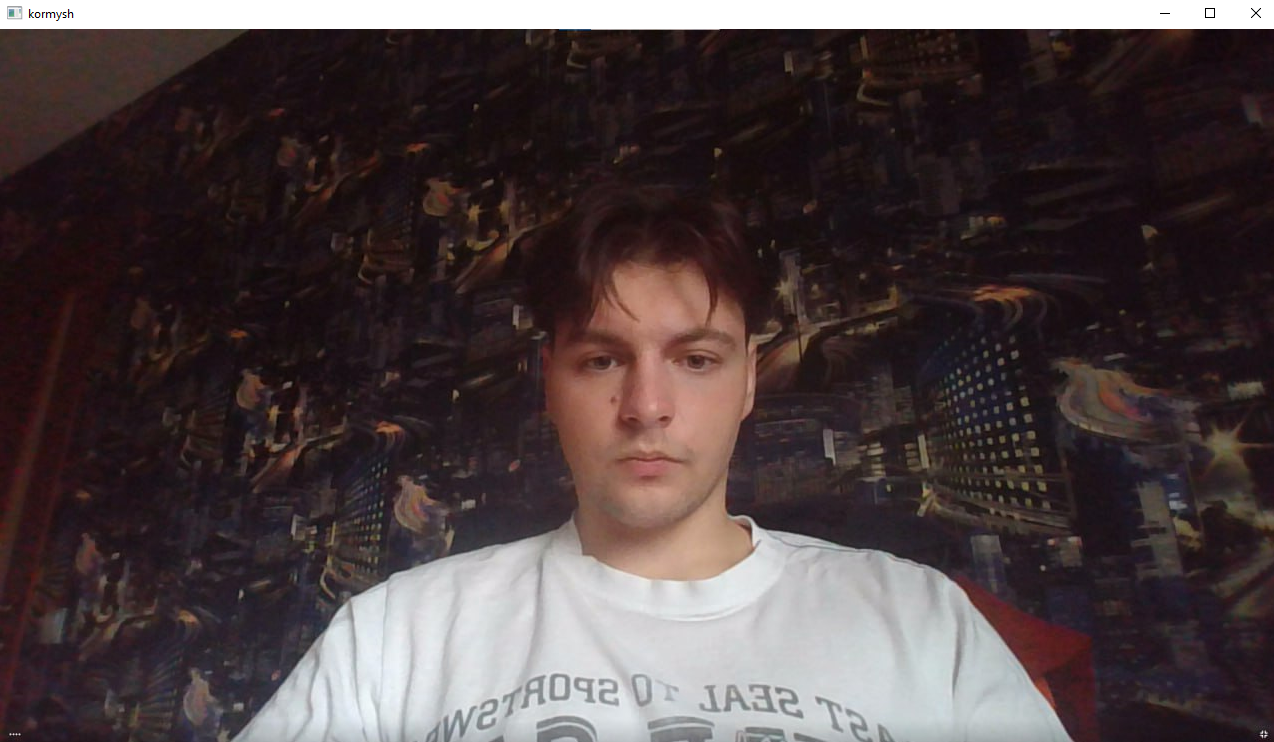


Рис.1 – Успішно завантажене фото.

**Завдання 2. Дослідження перетворень зображення.**

Лістинг коду файлу LR\_8\_task\_2.py:

import cv2  
import numpy as np  
  
img = cv2.imread("kravchenko.jpg")  
kernel = np.ones((5,5),np.uint8)  
*# is used to convert an image from one color space to another*imgGray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)  
*# Blurred image*imgBlur = cv2.GaussianBlur(imgGray,(7,7),0)  
*# used to detect the edges in an image.*imgCanny = cv2.Canny(img,150,200)  
*# used to apply the dilation operation on the given image with the specified kernel*imgDialation = cv2.dilate(imgCanny,kernel,iterations=1)  
*# method is used to perform erosion on the image.*imgEroded = cv2.erode(imgDialation,kernel,iterations=1)  
cv2.imshow("Gray Image",imgGray)  
cv2.imshow("Blur Image",imgBlur)  
cv2.imshow("Canny Image",imgCanny)  
cv2.imshow("Dialation Image",imgDialation)  
cv2.imshow("Eroded Image",imgEroded)  
cv2.waitKey(0)

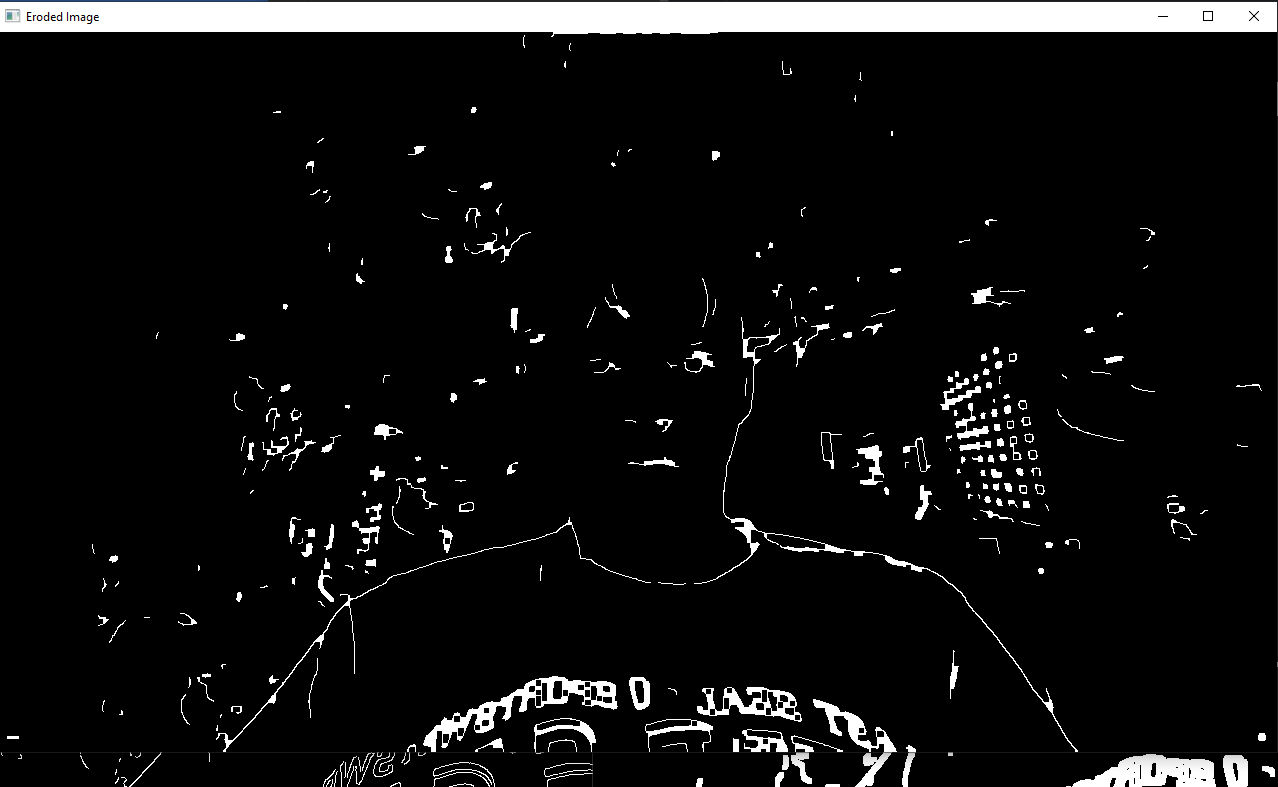


Рис.2 – Результутат виконання програми.



Рис.3 – Результутат виконання програми.

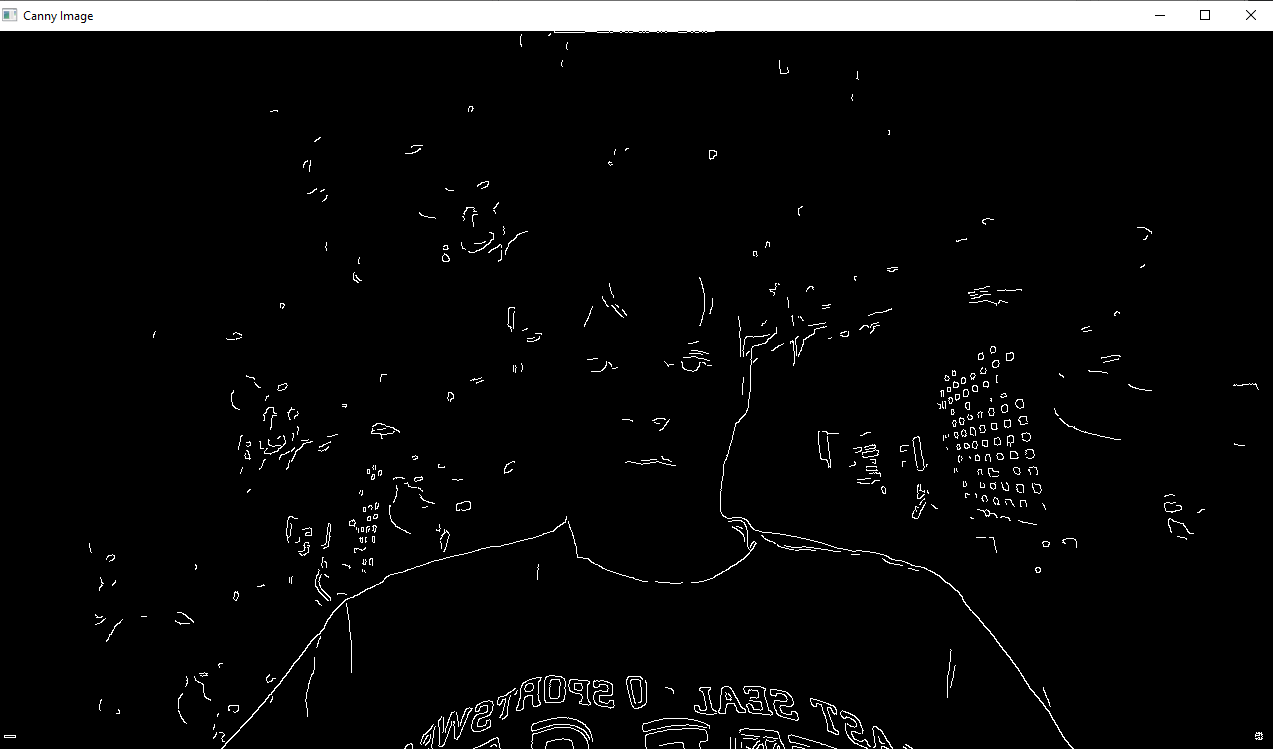


Рис.4 – Результутат виконання програми.



Рис.5 – Результутат виконання програми.

****

Рис.6 – Результутат виконання програми.

Метод cvtColor - використовується для перетворення зображення з одного колірного простору на інший.

Метод dilate() приймає два вхідні параметри, один із яких є нашим вхідним зображенням; другий називається структуруючим елементом чи ядром, яке визначає характер операції. Розширення зображення Збільшує об'єкт.

Метод erode - метод використовується для виконання розмивання зображення.

Метод GaussianBlur - розмите зображення.

Метод Canny - використовується для визначення країв зображення.

**Завдання 3. Вирізання частини зображення.**

Лістинг коду файлу LR\_8\_task\_3.py:

import cv2  
  
img = cv2.imread("kravchenko.jpg")  
print(img.shape)  
  
imgResize = cv2.resize(img, (1000, 500))  
print(imgResize.shape)  
  
imgCropped = img[46:119, 352:495]  
  
cv2.imshow("Image", img)  
cv2.imshow("Image Resize",imgResize)  
cv2.imshow("Image Cropped",imgCropped)  
cv2.waitKey(0)

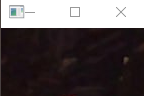


Рис.7 – Результутат виконання програми.

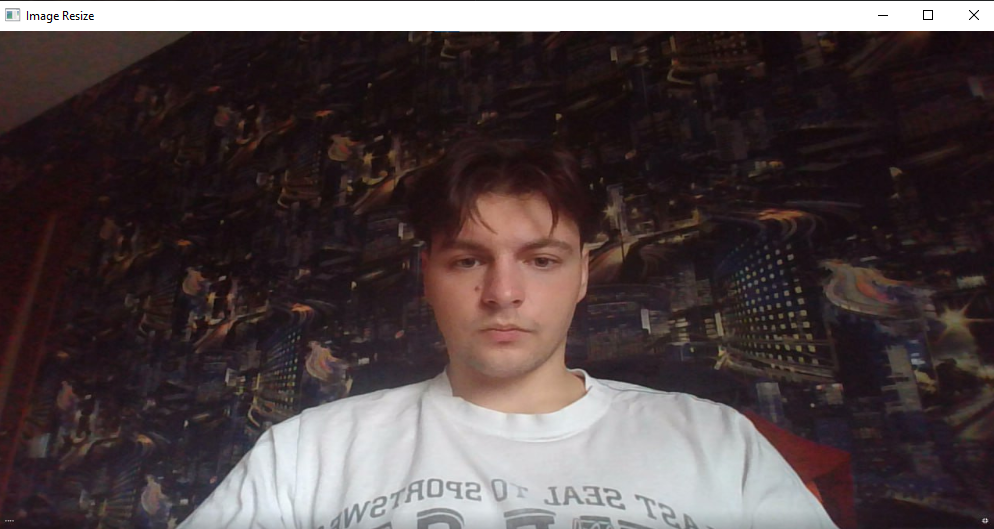


Рис.8 – Результутат виконання програми.

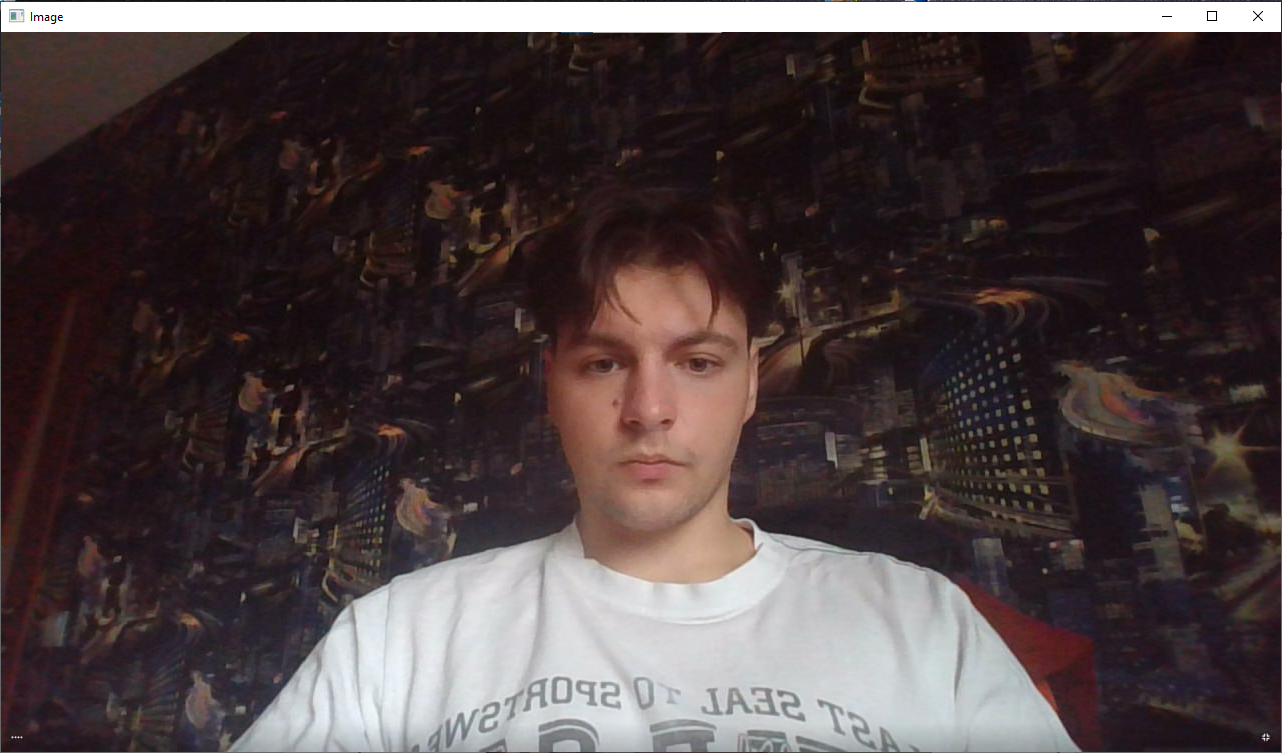


Рис.9 – Результутат виконання програми.

**Завдання 4.** **Розпізнавання обличчя на зображенні.**

Лістинг коду файлу LR\_8\_task\_4.py:

import cv2  
  
faceCascade = cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")  
img = cv2.imread('kravchenko.jpg')  
imgGray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  
  
faces = faceCascade.detectMultiScale(imgGray, 1.1, 4)  
for (x, y, w, h) in faces: cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)  
cv2.imshow("Result", img)  
cv2.waitKey(0)



Рис.10 – Результутат виконання програми.

**Завдання 5. Розпізнавання об’єктів на зображенні за допомогою методів зіставлення шаблонів (Template Matching).**

Лістинг коду файлу LR\_8\_task\_5.py:

import cv2 as cv  
import numpy as np  
from matplotlib import pyplot as plt  
  
img = cv.imread('kravchenko.JPG', 0)  
img2 = img.copy()  
template = cv.imread('kravchenko\_face.png', 0)  
  
w, h = template.shape[::-1]  
*# All the 6 methods for comparison in a list*methods = ['cv.TM\_CCOEFF', 'cv.TM\_CCOEFF\_NORMED', 'cv.TM\_CCORR',  
 'cv.TM\_CCORR\_NORMED', 'cv.TM\_SQDIFF', 'cv.TM\_SQDIFF\_NORMED']  
for meth in methods:  
 img = img2.copy()  
 method = eval(meth)  
 *# Apply template Matching* res = cv.matchTemplate(img, template, method)  
 min\_val, max\_val, min\_loc, max\_loc = cv.minMaxLoc(res)  
  
 if method in [cv.TM\_CCOEFF, cv.TM\_CCOEFF\_NORMED]:  
 top\_left = min\_loc  
 else:  
 top\_left = max\_loc  
 bottom\_right = (top\_left[0] + w, top\_left[1] + h)  
 cv.rectangle(img, top\_left, bottom\_right, 255, 2)  
  
 plt.subplot(121), plt.imshow(res, cmap='gray')  
 plt.title('Matching Result'), plt.xticks([]), plt.yticks([])  
 plt.subplot(122), plt.imshow(img, cmap='gray')  
 plt.title('Detected Point'), plt.xticks([]), plt.yticks([])  
 plt.suptitle(meth)  
  
 plt.show()

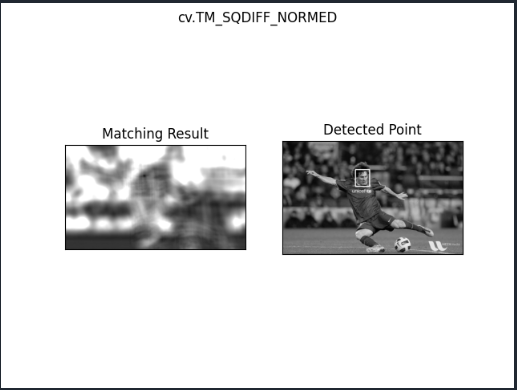


Рис.11 – Результутат виконання програми.

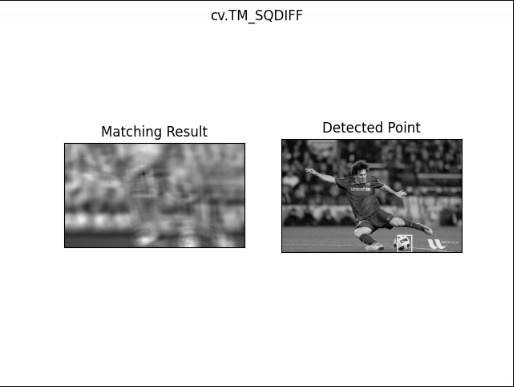


Рис.12 – Результутат виконання програми.

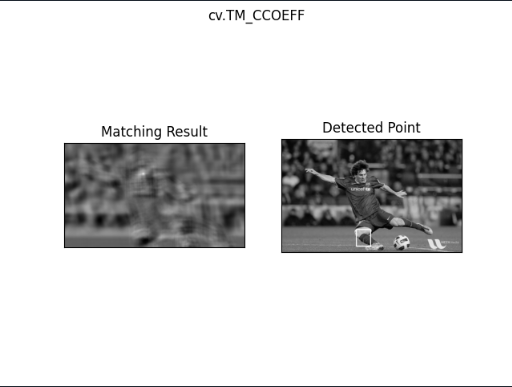


Рис.13 – Результутат виконання програми.

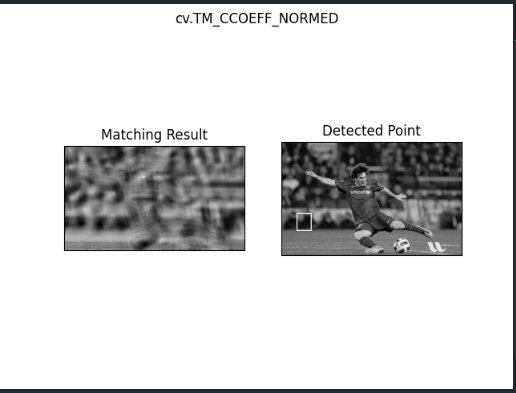


Рис.14 – Результутат виконання програми.

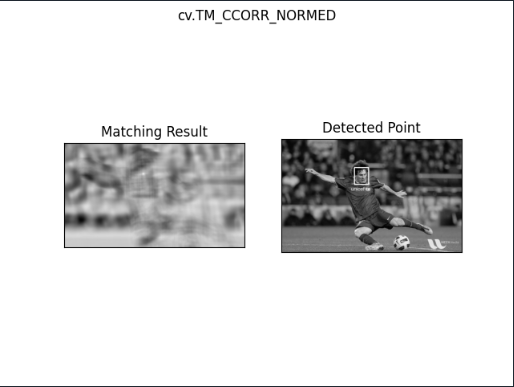


Рис.15 – Результутат виконання програми.

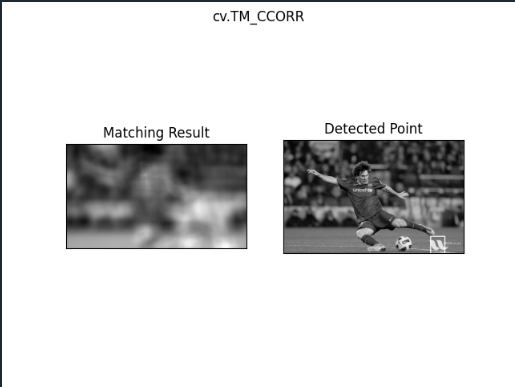


Рис.16 – Результутат виконання програми.

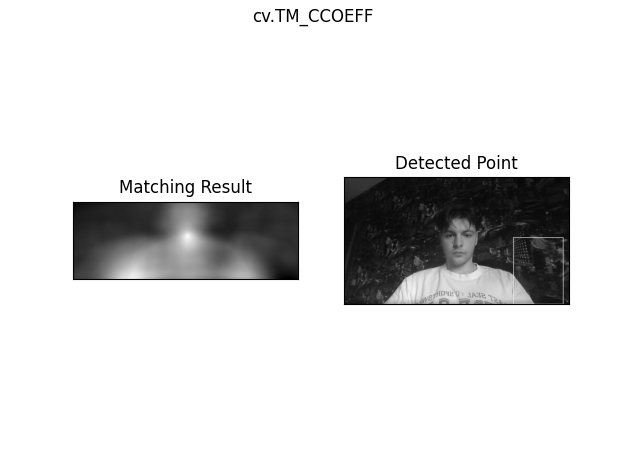


Рис.17 – Результутат виконання програми.

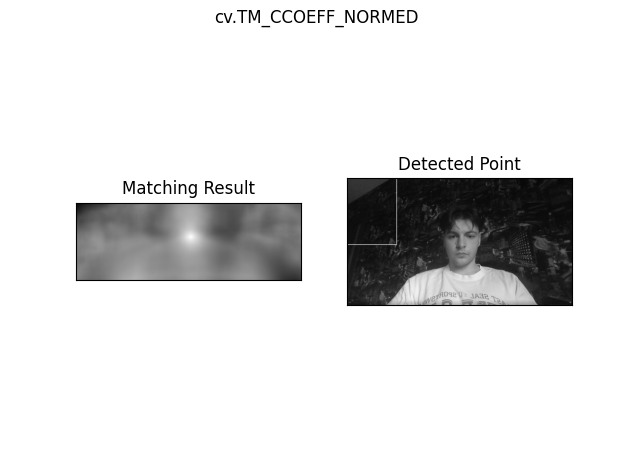


Рис.18 – Результутат виконання програми.



Рис.19 – Результутат виконання програми.

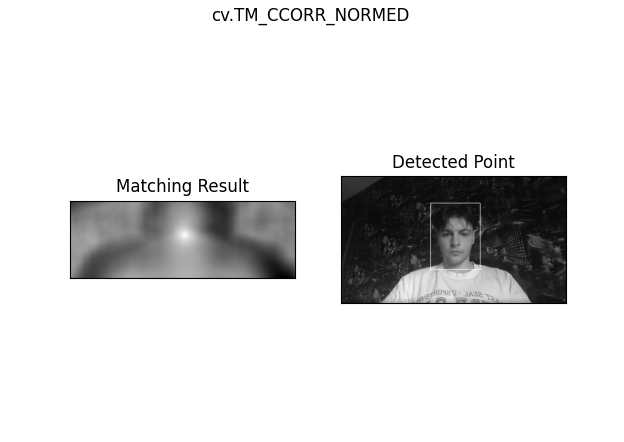


Рис.20 – Результутат виконання програми.

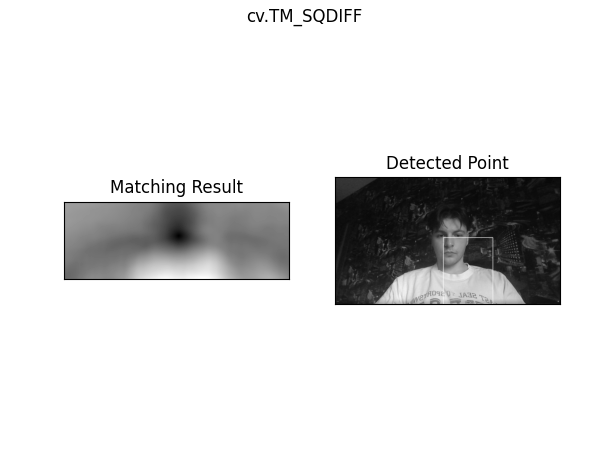


Рис.21 – Результутат виконання програми.



Рис.22 – Результутат виконання програми.



Рис.23 – Результутат виконання програми.



Рис.24 – Результутат виконання програми.

**Завдання 6. Сегментація зображення алгоритмом водорозподілу.**

Лістинг коду файлу LR\_8\_task\_6.py:

import numpy as np  
import cv2  
from matplotlib import pyplot as plt  
  
img = cv2.imread('coins.jpg')  
cv2.imshow("coins", img)  
cv2.waitKey(0)  
  
gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  
ret, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH\_BINARY\_INV + cv2.THRESH\_OTSU)  
cv2.imshow("coins bin ", thresh)  
cv2.waitKey(0)  
  
*# видалення шуму*kernel = np.ones((3, 3), np.uint8)  
opening = cv2.morphologyEx(thresh, cv2.MORPH\_OPEN, kernel, iterations=2)  
*# певна фонова область*sure\_bg = cv2.dilate(opening, kernel, iterations=3) *# Пошук впевненої області переднього плану*dist\_transform = cv2.distanceTransform(opening, cv2.DIST\_L2, 5)  
ret, sure\_fg = cv2.threshold(dist\_transform, 0.7 \* dist\_transform.max(), 255, 0)  
*# Пошук невідомого регіону*sure\_fg = np.uint8(sure\_fg)  
unknown = cv2.subtract(sure\_bg, sure\_fg)  
cv2.imshow("coins ", opening)  
cv2.waitKey(0)  
  
*# Маркування міток*ret, markers = cv2.connectedComponents(sure\_fg)  
*# Додайте один до всіх міток, щоб впевнений фон був не 0, а 1*markers = markers + 1  
*# Тепер позначте область невідомого нулем*markers[unknown == 255] = 0  
  
markers = cv2.watershed(img, markers)  
img[markers == -1] = [255, 0, 0]  
  
cv2.imshow("coins\_markers", img)  
cv2.waitKey(0)



Рис.25 – Результутат виконання програми.

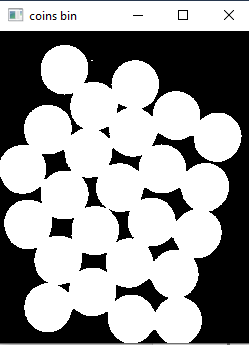


Рис.26 – Результутат виконання програми.

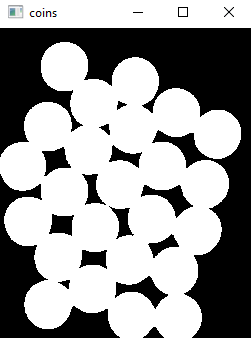


Рис.27 – Результутат виконання програми.



Рис.28 – Результутат виконання програми

Посилання на GitHub: <https://github.com/Raimhal1/AI>

**Висновки:** під час виконання завдання лабораторної роботи використовуючи спеціалізовані бібліотеки та мову програмування Python ми навчилися обробляти зображення за допомогою бібліотеки OpenCV.