**Практична робота № 8**

**Варіант 13**

**Дослідження методів комп’ютерного зору**

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

**Хід роботи:**

**Завдання 8.1:** Завантаження зображень та відео в OpenCV

Лістинг:

import cv2  
  
def load\_image():  
 img = cv2.imread('data/makovska.png')  
 cv2.imshow('Output', img)  
 cv2.waitKey(0)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 load\_image()



Рис.8.1. Результат виконання

**Завдання 8.2:** Дослідження перетворень зображення

Лістинг:

import cv2  
import numpy as np  
  
img = cv2.imread('data/makovska.png')  
kernel = np.ones((5, 5), np.uint8)  
  
imgGray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  
imgBlur = cv2.GaussianBlur(imgGray, (7, 7), 0)  
imgCanny = cv2.Canny(img, 150, 200)  
imgDilation = cv2.dilate(imgCanny, kernel, iterations=1)  
imgEroded = cv2.erode(imgDilation, kernel, iterations=1)  
  
cv2.imshow('Gray Image', imgGray)  
cv2.imshow('Blur Image', imgBlur)  
cv2.imshow('Canny Image', imgCanny)  
cv2.imshow('Dilation Image', imgDilation)  
cv2.imshow('Eroded Image', imgEroded)  
cv2.waitKey(0)

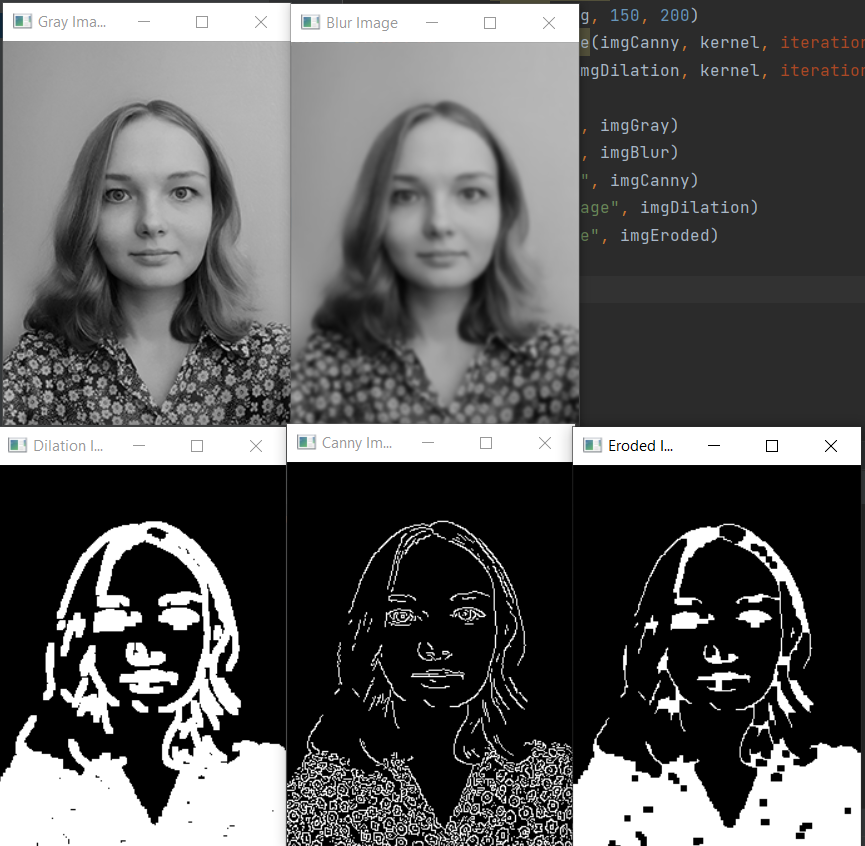


Рис.8.2. Результат виконання

**Завдання 8.3:** Вирізання частини зображення

Лістинг:

import cv2  
  
img = cv2.imread('data/makovska.png')  
print(img.shape)  
imgResize = cv2.resize(img, (1000, 500))  
print(imgResize.shape)  
imgCropped = img[75:400, 30:350]  
cv2.imshow('Image', img)  
  
cv2.imshow('Image Cropped', imgCropped)  
cv2.waitKey(0)

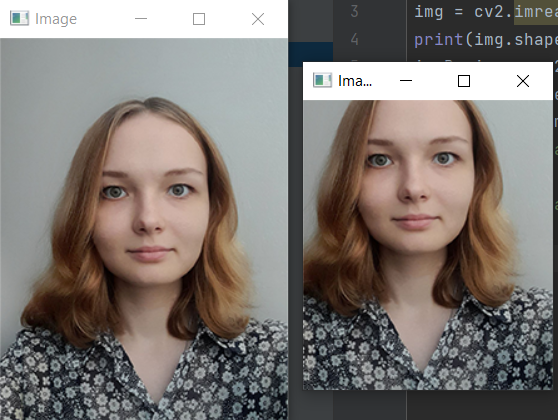


Рис.8.3. Результат виконання

**Завдання 8.4:** Розпізнавання обличчя на зображенні

Лістинг:

import cv2  
  
faceCascade = cv2.CascadeClassifier('data/haarcascade\_frontalface\_default.xml')  
img = cv2.imread('data/makovska.png')  
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)

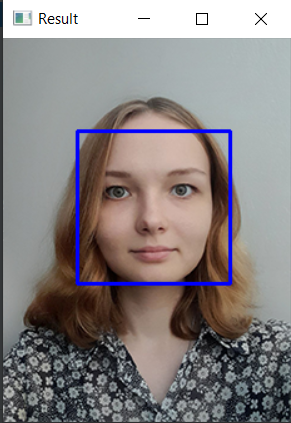
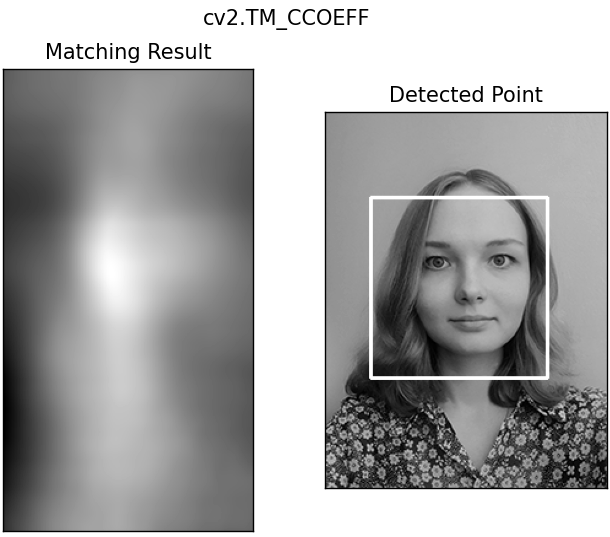
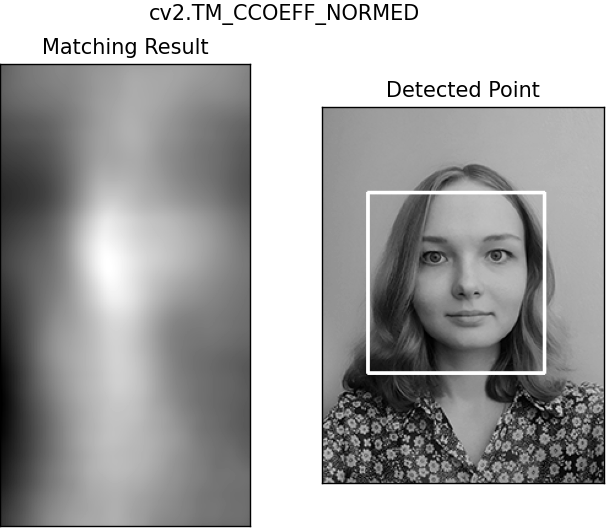


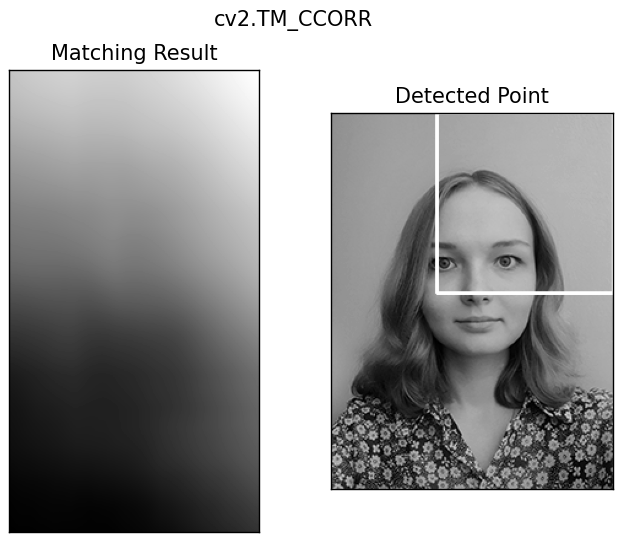
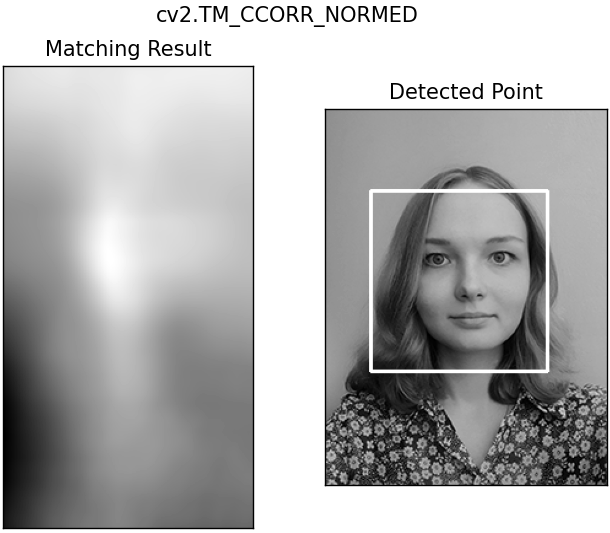
Рис.8.4. Результат виконання

**Завдання 8.5:** Розпізнавання об’єктів на зображенні за допомогою методів зіставлення шаблонів

Лістинг:

img = cv.imread('data/makovska.png', 0)  
img2 = img.copy()  
template = cv.imread('data/makovska\_face.png', 0)  
w, h = template.shape[::-1]  
  
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)  
  
 res = cv.matchTemplate(img, template, method)  
 min\_val, max\_val, min\_loc, max\_loc = cv.minMaxLoc(res)  
  
 if method in [cv.TM\_SQDIFF, cv.TM\_SQDIFF\_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()

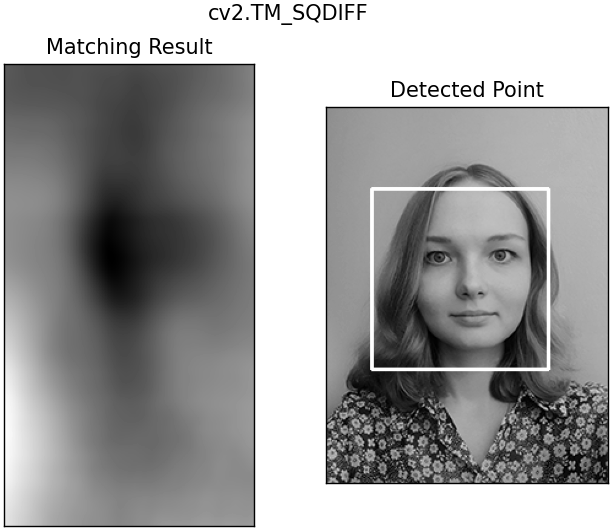
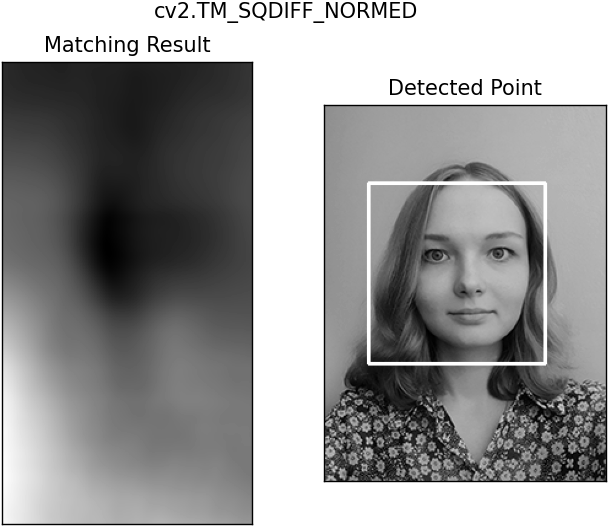
 

Рис.8.5. Результат виконання

**Завдання 8.6:** Сегментація зображення алгоритмом водорозподілу

Лістинг:

import numpy as np  
import cv2  
  
img = cv2.imread('data/coins\_2.jpg')  
cv2.imshow('Original', 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('Threshold', thresh)  
cv2.waitKey(0)  
  
kernel = np.ones((3, 3), np.uint8)  
opening = cv2.morphologyEx(thresh, cv2.MORPH\_OPEN, kernel, iterations=2)  
  
sure\_bg = cv2.dilate(opening, kernel, iterations=3)  
dist\_transform = cv2.distanceTransform(opening, cv2.DIST\_L2, 5)  
ret, sure\_fg = cv2.threshold(dist\_transform, 0.7 \* dist\_transform.max(), 255, 0)  
  
sure\_fg = np.uint8(sure\_fg)  
unknown = cv2.subtract(sure\_bg, sure\_fg)  
cv2.imshow('Coins', opening)  
cv2.waitKey(0)  
  
ret, markers = cv2.connectedComponents(sure\_fg)  
markers = markers + 1  
markers[unknown == 255] = 0  
  
markers = cv2.watershed(img, markers)  
img[markers == -1] = [255, 0, 0]  
  
cv2.imshow('Markers', img)  
cv2.waitKey(0)



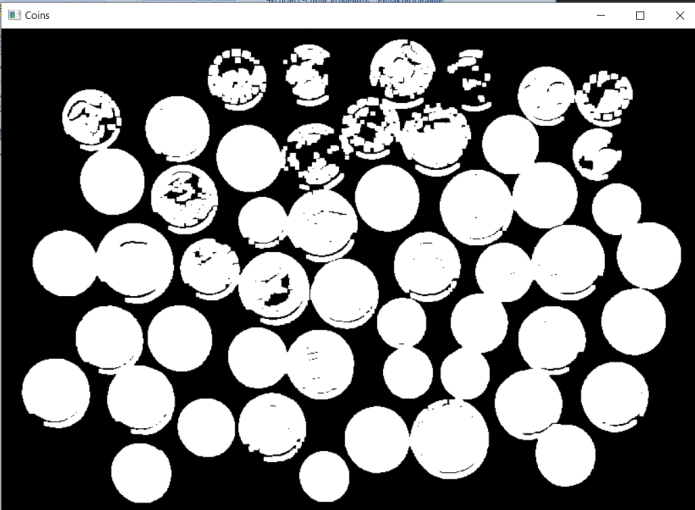
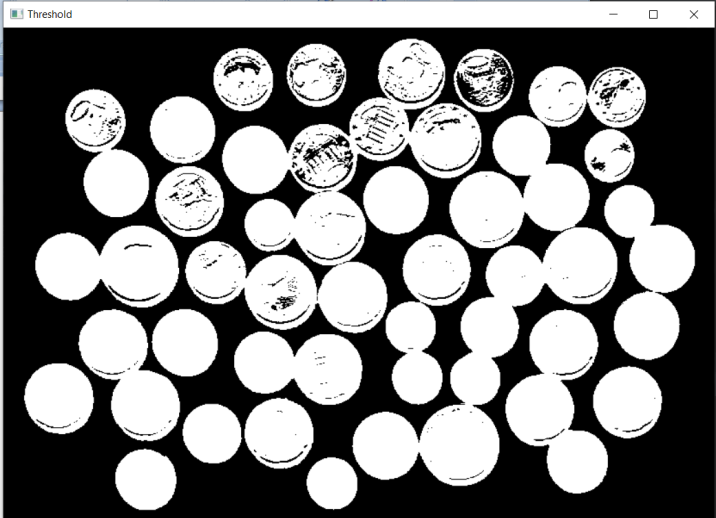




Рис.8.6. Результат виконання

**Завдання 8.7:** Дослідження перетворень зображення

Лістинг:

import cv2  
import numpy as np  
from scipy import ndimage as ndi  
from skimage.feature import peak\_local\_max  
from skimage.segmentation import watershed  
import matplotlib.pyplot as plt  
  
img = cv2.imread('coins\_2.jpg')  
filtered = cv2.pyrMeanShiftFiltering(img, 20, 40)  
  
gray = cv2.cvtColor(filtered, cv2.COLOR\_BGR2GRAY)  
\_, thresh\_img = cv2.threshold(gray, 0, 255, cv2.THRESH\_BINARY\_INV | cv2.THRESH\_OTSU)  
  
contours, \_ = cv2.findContours(thresh\_img, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
contours\_filtered = []  
for con in contours:  
 area = cv2.contourArea(con)  
 if area < 1000:  
 contours\_filtered.append(con)  
  
cv2.drawContours(thresh\_img, contours\_filtered, -1, 255, -1)  
dist = ndi.distance\_transform\_edt(thresh\_img)  
dist\_copy = dist.copy()  
  
local\_max = peak\_local\_max(dist, indices=False, min\_distance=20, labels=thresh\_img)  
markers = ndi.label(local\_max, structure=np.ones((3, 3)))[0]  
  
watershed\_img = watershed(-dist, markers, mask=thresh\_img)  
titles = ['Original image', 'Binary Image', 'Watershed']  
images = [img, thresh\_img, watershed\_img]  
  
fig = plt.gcf()  
fig.set\_size\_inches(8, 6)  
  
for i, img in enumerate(images):  
 plt.subplot(3, 1, i + 1)  
 plt.imshow(img, 'jet' if i == 2 else 'gray')  
 plt.title(titles[i])  
 plt.xticks([]), plt.yticks([])  
  
plt.show()

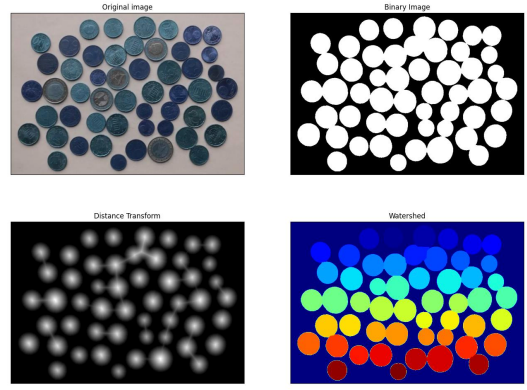


Рис.8.7. Результат виконання

**<https://github.com/avrorilka/AI_Python>**

**Висновки:** протягом виконання завдань лабораторної роботи було отримано навички з обробки зображення використовуючи бібліотеку OpenCV та мову програмування Python

Під час цих лабораторних завдань ми навчилися використовувати бібліотеки OpenCV для обробки зображень. Ми використали методи cvtColor, GaussianBlur, Canny, dilate та erode для перетворення зображення з одного колірного простору в інший, зменшення шуму та деталізації, виявлення країв і збільшення або зменшення розміру об’єкта на зображенні. Результатом застосування цих методів є зображення в новому колірному просторі зі зниженим шумом і деталізацією, з виділеними краями, а також зі збільшеним або зменшеним розміром об’єкта.

Метод cvtColor використовується для перетворення зображення з одного колірного простору в інший. Його можна використовувати, наприклад, для перетворення зображення з RGB на градації сірого. Результатом застосування цього методу є зображення в новому колірному просторі.

Метод GaussianBlur використовується для зменшення шуму та деталізації зображення. Він застосовує до зображення фільтр Гауса, який розмиває зображення та зменшує кількість шумів і деталей. Результатом застосування цього методу є зображення зі зниженим шумом і деталізацією.