МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

КАФЕДРА СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ

ЗВІТ

про виконання лабораторної роботи №2

з дисципліни: «Обробка зображень методами штучного інтелекту»

на тему: «Суміщення зображень на основі використання дескрипторів»

Виконав:

студент групи КН-410

Шиманський П.С.

Прийняв:

Пелешко Д.Д.

Львів – 2021

Мета: навчитись вирішувати задачу суміщення зображень засобом видобування особливих точок і використання їх в процедурах.

Теоретичні відомості

Загалом алгоритмі SIFT складається з п’яти основних етапів:

1. Виявлення масштабно-просторових екстремумів (Scale-space Extrema Detection) - основним завданням етапу є виділення локальних екстремальних точок засобом побудова пірамід гаусіанів (Gaussian) і різниць гаусіанів (Difference of Gaussian, DoG).

2. Локалізація ключових точок (Keypoint Localization) - основним завданням етапу є подальше учточнення локальних екстремумів з метою фільтрації їх набору - тобто видалення з подальшого аналізу точок, які є краєвими, або мають низьку контрастність.

3. Визначення орієнтації (Orientation Assignment) - для досягнення інваріантності повороту растра на цьому етапі кожній ключовій точці присвоюється орієнтація.

4. Дескриптор ключових точок (Keypoint Descriptor) - завданням етпау є побудова дескрипторів, які містяь інформація про окіл особливої точки для задачі подальшого порівння на збіг.

5. Зіставлення по ключових точках (Keypoint Matching) - пошук збігів для вирішення завдання суміщення зображень.

Код програми

```
import numpy as np

import cv2

img1 = cv2.imread('a11.jpg')
```

```
img2 = cv2.imread('a22.jpg')


img1 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
img2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)


sift = cv2.SIFT_create()


keypoints_1, descriptors_1 = sift.detectAndCompute(img1,None)
keypoints_2, descriptors_2 = sift.detectAndCompute(img2,None)
def dist(x,y):
    n=len(x)
    assert len(x) == len(y)
    return sum([(x[i]-y[i])**2 for i in range(n)])


matches = []


for i, k1 in enumerate(descriptors_1):
    for j, k2 in enumerate(descriptors_2):
        matches.append(cv2.DMatch(_distance=dist(k1, k2), _imgIdx=0, _queryIdx=i, _trainIdx=j))
bf = cv2.BFMatcher(cv2.NORM_L1, crossCheck=True)


matches = bf.match(descriptors_1,descriptors_2)
matches = sorted(matches, key = lambda x:x.distance)


matched_img = cv2.drawMatches(img1, keypoints_1, img2, keypoints_2, matches[:50], img2, flags=2)
cv2.imshow('image', matched_img)


cv2.imwrite("matched_images.jpg", matched_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```