МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту

Лабораторна робота №2

3 курсу

“Обробка зображень методами штучного інтелекту”

На тему:

«Суміщення зображень на основі використання дескрипторів»

Виконав:
ст. групи КН-408
Бурцьо Ольга
Викладач:
Пелешко Д. Д.

Львів – 2022

Варіант 5

Завдання: Вибрати з інтернету набори зображень з різною контрастністю і різним флуктуаціями освітленості. Для кожного зображення побудувати варіант спотвореного (видозміненого зображення). Для кожної отриманої пари побудувати дескриптор і проаналізувати можливість суміщення цих зображень і з визначення параметрів геметричних перетворень (кут повороту, зміщень в напрямку x і напрямку y).

AKAZE.

Для перевірки збігів необхідно написати власну функцію матчінгу, а результати її роботи перевірити засобами OpenCV. Якщо повної реалізації дескриптора не має в OpenCV, то такий необхідно створити власну функцію побудови цих дискрипторів. У цьому випадку матчінг можна здійснювати стандартними засобами (якщо це можливо).

Теоретичні відомості

Метод SIFT.

У 2004 році Д.Лоу, Університет Британської Колумбії, придумав алгоритм - Scale Invariant Feature Transform (SIFT), який видобуває ключові (особливі) точки і обчислює їх дескриптори.

Загалом алгоритм 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 побудови пірамід гаусіанів (Gaussian) і різниць гаусіанів (Difference of Gaussian, DoG).

2. Локалізація ключових точок (Keypoint Localization) - основним завданням етапу є подальше уточнення 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 етапу є побудова дескрипторів, які містять інформацію про окіл особливої точки для задачі подальшого порівння на збіг.

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

Алгоритм RANSAC - Random sample consensus

Для досягнення високої точності визначення збігів об'єктів на зображеннях зазвичай відфільтрувати дескриптори тільки за відстанню є недостатньо. Якщо об'єкт рухається на сцені або зображений з іншого ракурсу, то при застосуванні трансформації «накладення» n точок одного зображення на відповідні по найближчому сусіду n точок іншого, можна виявити особливості, що не відносяться до загального об'єкту і тим самим зменшити кількість хибно виявлених зв'язків.

Схема роботи алгоритму RANSAC полягає в циклічному повторенні пошуку матриці трансформації HН між чотирма особливими точками s_i, які випадково обираються i на одному зображенні, і відповідними їм точками на другому:

$$s_i \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim H \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix}$$

Найкращою матрицею трансформації вважається та, в якій досягнуто мінімум суми відхилень будь-яких спеціальних точок зображень при перетворенні НH, за задану кількість циклів (≤ 2000):

$$\sum_i \left[\left(x_i - \frac{h_{11}x'_i + h_{12}y'_i + h_{13}}{h_{31}x'_i + h_{32}y'_i + h_{33}}\right)^2 + \left(y_i - \frac{h_{21}x'_i + h_{22}y'_i + h_{23}}{h_{31}x'_i + h_{32}y'_i + h_{33}}\right)^2 \right]$$

У підсумкову множину srcPoints’ додаються тільки ті точки $srcPo$ srcPointsi$i$, відхилення яких становить менше заданого порогу:

$$|dstPointsi - H * srcPointsi| < reprojThreshold$$

де srcPoints$srcPoin$ - множина усіх особливих точок першого зображення, а dstPoints - множина відповідних їм особливих точок другого.

Основи Brute-Force Matcher

Brute-Force matcher (BF-matcher) реалізовує простий матчінг метод. Він приймає дескриптор однієї ознаки в першій множині і зіставляє за деякою метрикою з усіма ознаками в другій множині. Як результат повертається найближчий дескриптор (ознака).

Для використання BF-matcher спочатку використовуючи функцію cv.BFMatcher () необхідно створити об'єкт BFMatcher. Функція має два

необов'язкові параметри. Перший - normType. Він задає тип метрики для вимірювання відстані між дескрипторами. За замовчуванням використовується метрика L2 (cv.NORM_L2). Для SIFT та SURF і т.д. також рекомендується використовувати метрику L1 (cv.NORM_L1).

Для дескрипторів, заснованих на бінарних рядках, таких як ORB, BRIEF, BRISK і т.д., треба використовувати метрику Хемінга (cv.NORM_HAMMING). Якщо ORB використовує WTA_K == 3 або 4, то слід використовувати cv.NORM_HAMMING2.

Другий параметр функції створення матчера є булівська змінна crossCheck, яка за замовчуванням рівна False. Якщо встановити її в True, то Matcher повертає тільки ті збіги, коли обидві ознаки в обох множинах збігаються одна з одною.

Після створення матчера його двома важливими методами є BFMatcher.match () і BFMatcher.knnMatch (). Перший повертає кращий збіг. Другий метод - повертає k кращих збігів, де k задається параметром.

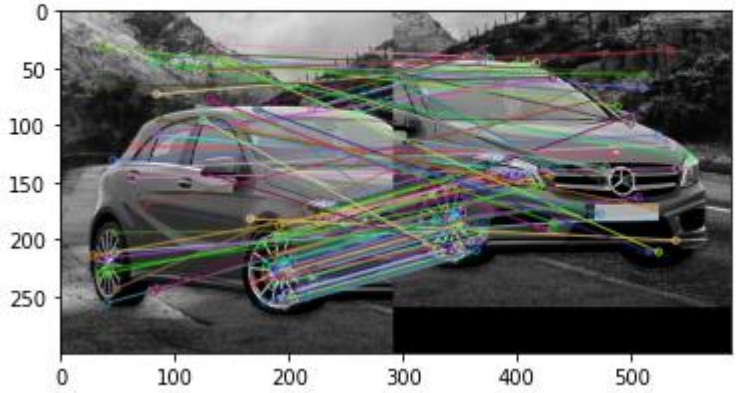
Як і ми використовували cv.drawKeypoints () для відтворення ключових точок, cv.drawMatches () допомагає нам малювати відповідності. Вона складає два зображення по горизонталі і малює лінії від першого зображення до другого, показуючи найкращі збіги. Також існує cv.drawMatchesKnn, який відображає всі k кращих збігів. Якщо k = 2, то він відобразить дві лінії збігів для кожної ключової точки. Тому ми повинні передати маску, якщо ми хочемо вибірково намалювати її.

Результати роботи програми:

- Вбудована функція

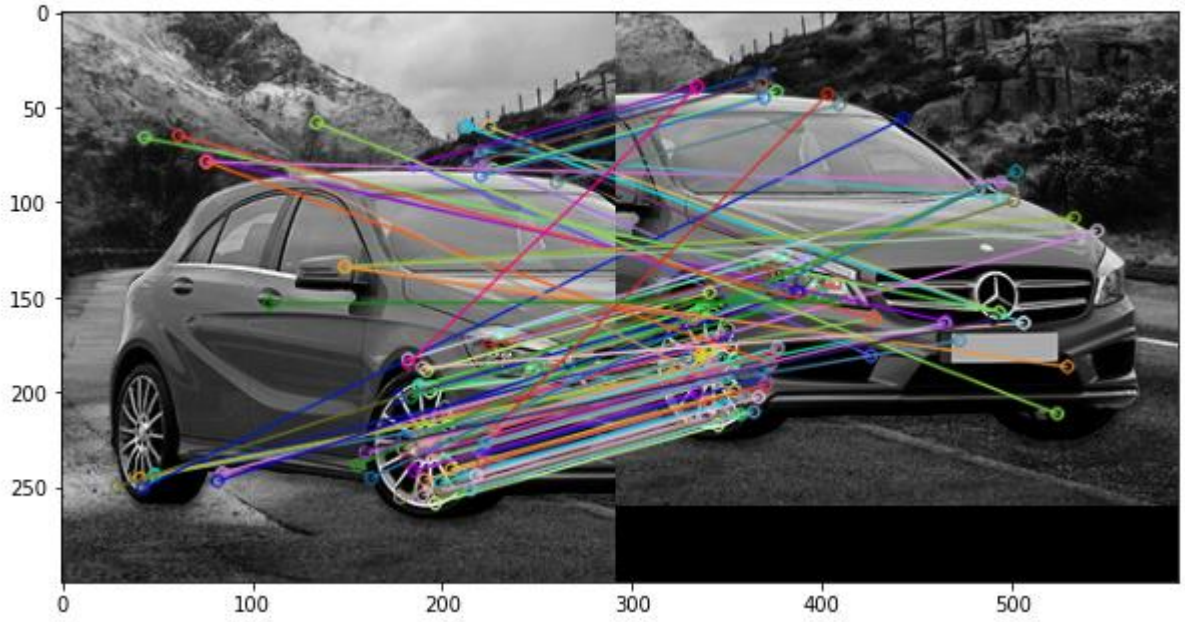
```
display_output(gray1,kps1,gray2,kps2,descs1,descs2)
```

```
Total Number of Features matches found are 114
```



- Власна функція

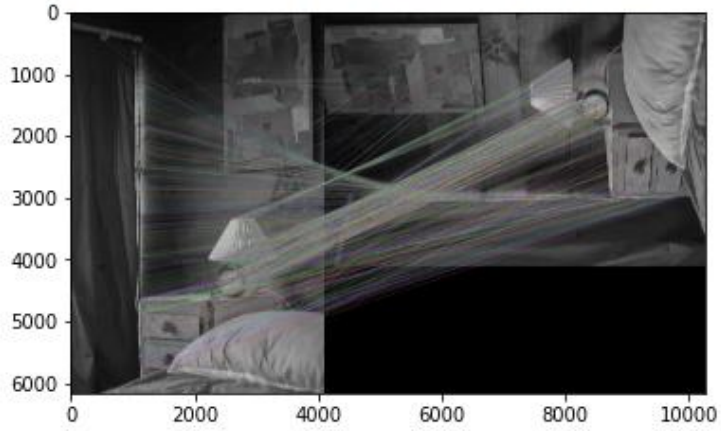
```
matches_to_draw = custom_match(kps1, descs1, kps2, descs2,gray1,gray2)
draw_custom_match(kps1, kps2, matches_to_draw, gray1, gray2)
```



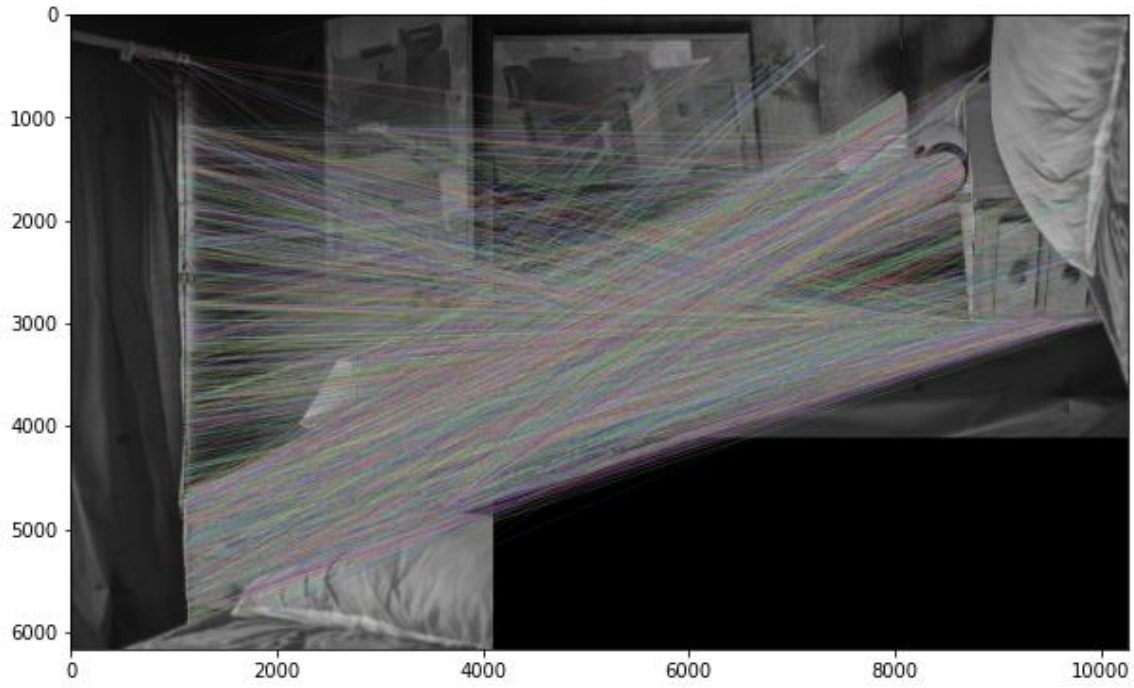
2 приклад (фото дуже деталізоване):

```
display_output(gray3,kps3,gray4,kps4,descs3,descs4)
```

```
Total Number of Features matches found are 1850
```




```
matches_to_draw = custom_match(kps3, descs3, kps4, descs4, gray3, gray4)
draw_custom_match(kps3, kps4, matches_to_draw, gray3, gray4)
```



Висновки: в результаті роботи над цією лабораторною роботою я написала власний матчер, який працює майже так само як і брут форс вбудований. Також на прикладі 2 перевірила що дескриптор AKAZE інваріантний до поворотів зображень.