НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ
ІГОРЯ СІКОРСЬКОГО ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Лабораторна робота №2

Виконали
команда “#000000 & #FFFFFF”
студенти КА-71
Зінченко Світлана
Бойко Павло

Прийняв
Андрій Лисенко

Київ-2020

Завдання

Найпершою і найголовнішою справою буде вибрати назву команди і записатись в табличку в цій же папці. У вас уже є OpenCV і навички роботи з зображеннями і не тільки, далі залишився мізер:

Кожен учасник або учасниця команди спершу обирає дескриптор (один із розглянутих у лекціях або ж знайдений окремо) та предмет на прикладі якого відбуватиметься дослідження. Враховуючи що вас багато, будь ласка обирайте унікальніші предмети за улюблену чашку/телефон/мишку. Маючи те і інше напоготові кожен учасник бригади має зняти не менше сотні фото предмета, варіюючи його розміщення та ракурс в кадрі, освітлення, наявність візуальних перешкод, зашакаленість зображення, фокусну віддаль та тремтіння рук. Сотня фото обраного предмету на однаковій сцені з однаковою якістю зйомки, але з різних ракурсів на жаль не підійде, постарайтесь наполегливо варіювати сцени і умови зйомки. До цих фото варто додати невелику підбірку зображень, що не містять предмет, або ж містять предмет візуально подібний до вашого, штук 20 повинно вистачити, якщо залишиться натхнення можна й більше. Після чого ми нарешті дійшли до цікавого, а саме до дослідження:

Вам потрібно згенерувати обраний дескриптор для обраного предмета, після чого з його допомогою розпізнати об’єкт на всій тестовій вибірці збираючи при цьому такі метрики: відносна кількість правильно суміщених ознак, похибка локалізації (відстань між реальним розміщенням предмета в кадрі та розпізнаним) та відносний час обробки фото в залежності від розміру зображення. Метрики мають зберегтись у файлику для подальших досліджень.

Наступним кроком ви обмінюєтесь об’єктом з колегою, і уже маючи готову збиралку метрик, обчислюєте їх для предмета вашого сусіда, таким чином у вас збирається 9 наборів даних, по три на дескриптор.

Самою ж ідеєю лаби є дослідити розбіжності у роботі ваших дескрипторів та виконати порівняльний аналіз їх поведінки, сформулювати висновки з викладками і прикладами так аби було зрозуміло вам та, сподіваюсь, усім вашим колегам. Таким чином кінцевим результатом буде від вас гуглдок з описом виняткових особливостей, сильних та слабких сторін дескриптора і обгрунтуванням чому вони поводяться саме так.

Бажано аби не було більше однієї групи з однаковою комбінацією дескрипторів, хіба що ви об’єднаєтесь і влаштуєте вашим двом дескрипторам страхітливо прискіпливе дослідження.

Сорци, дослідні вибірки фото, зібрані метрики залляти на Гітхаб та кинути посиланням мені на пошту, в сабж листа копіпастимо "[CVPR 2020]" і далі що хочете, хоч ента намалюйте. Оберіть у команді одну контактну персону з пошти якої прилетить готова лаба і відбуватиметься подальша комунікація.

Предмети, що розглядаються

Перший предмет це зошит, але враховуючи циклічність принту(пелюстки по всій квітці досить схожі), було цікаво наскільки корректно обрані алгоритми здатні впоратися з цією задачею.

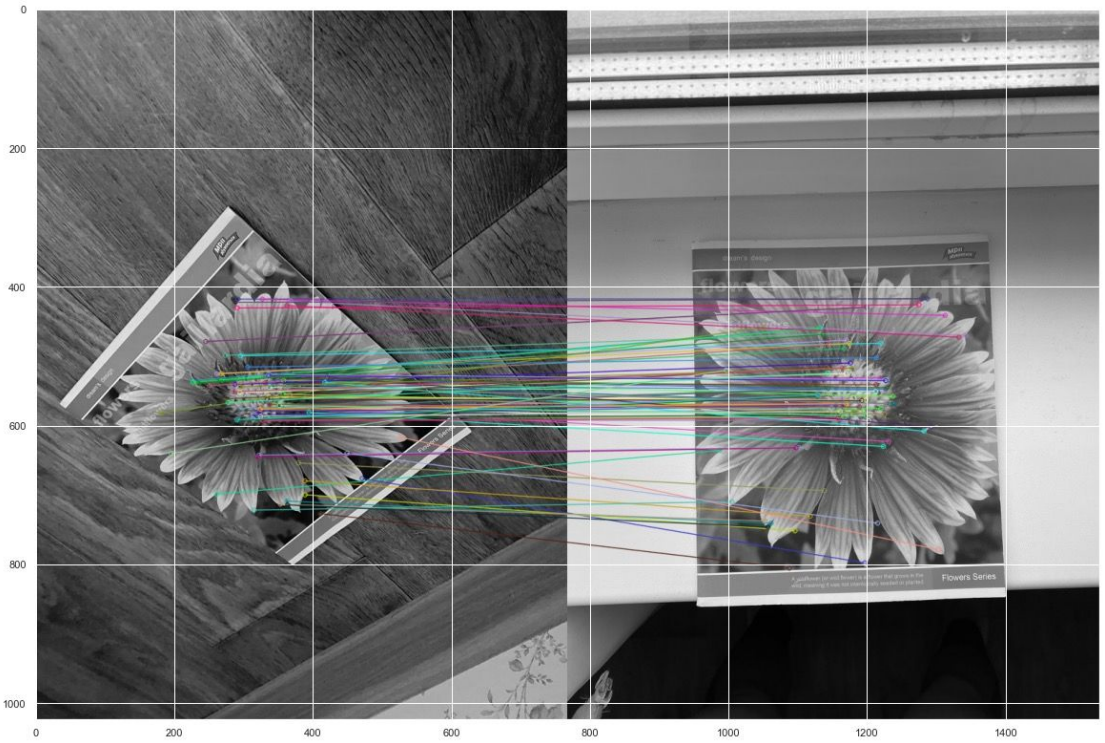
Другим предметом ми обрали автомобіль, суттєва різниця полягає у тому, що ми вже маємо справу з об’ємним предметом, від чого можуть виникати свої особливості у роботі алгоритмів.

Спочатку ми мали знайти особливі точки(key points) на зображеннях і згенерувати дескриптори для нашого train image і для всіх query image, за допомогою обраних алгоритмів. Надалі потрібно зметчити дескриптори на зображеннях, використовуючи FlannBasedMatcher і відфільтрувати правильні метчі для подальшого обчислення метрик і висновків про ефективність за допомогою Lowe's ratio test.

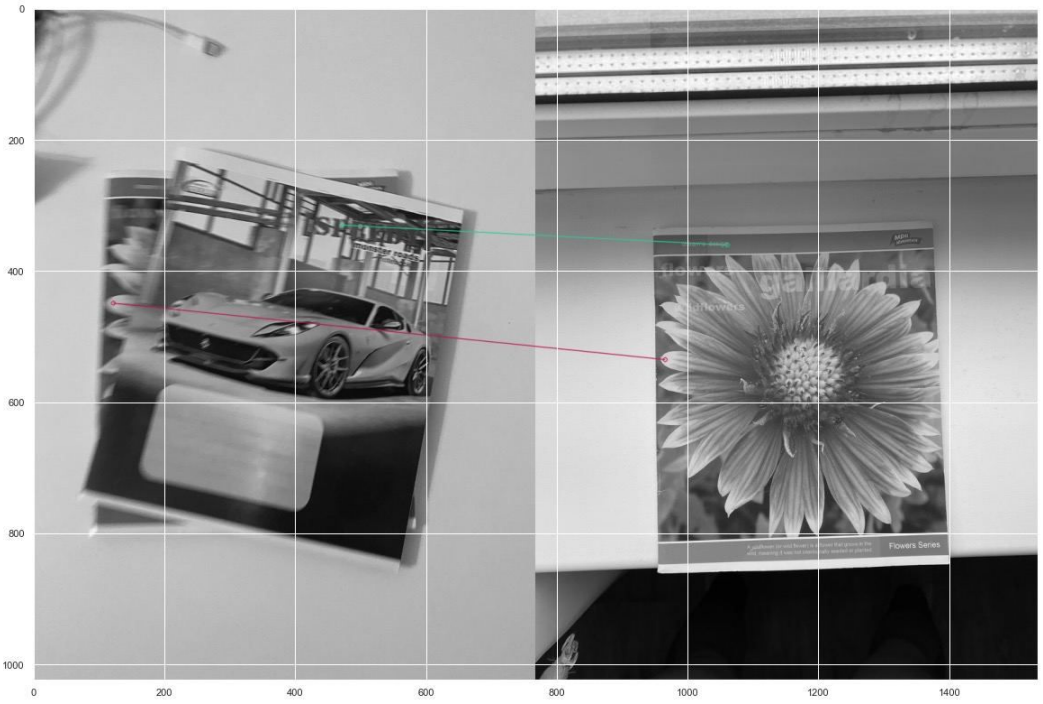
Приклади роботи алгоритмів:

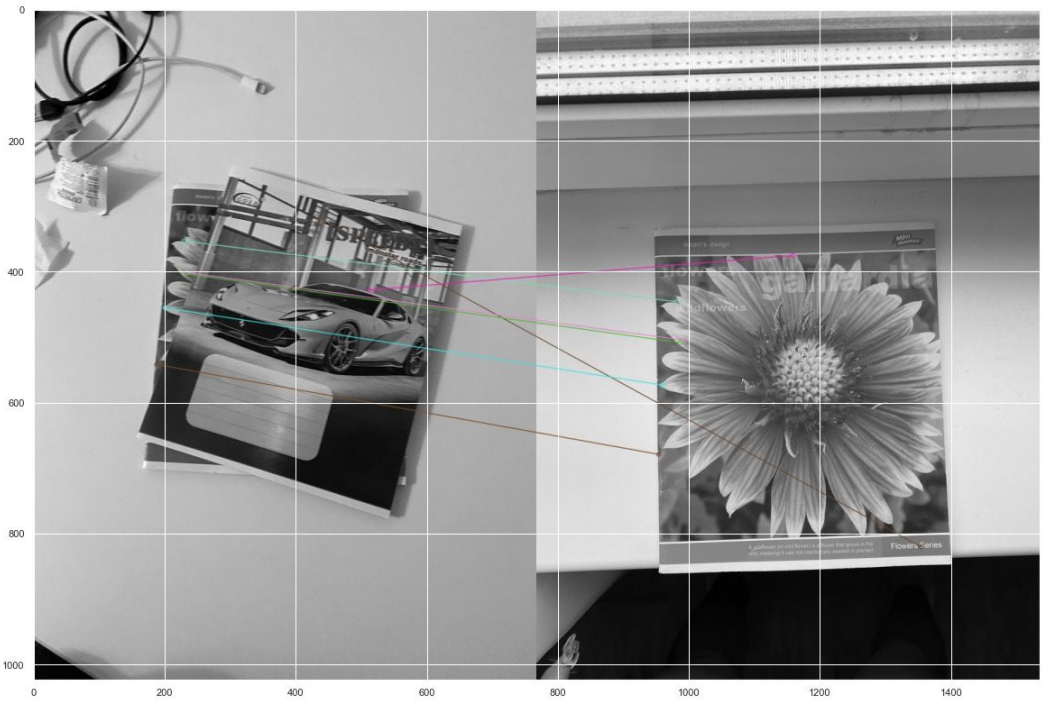
1. **BRISK**

- хороший приклад

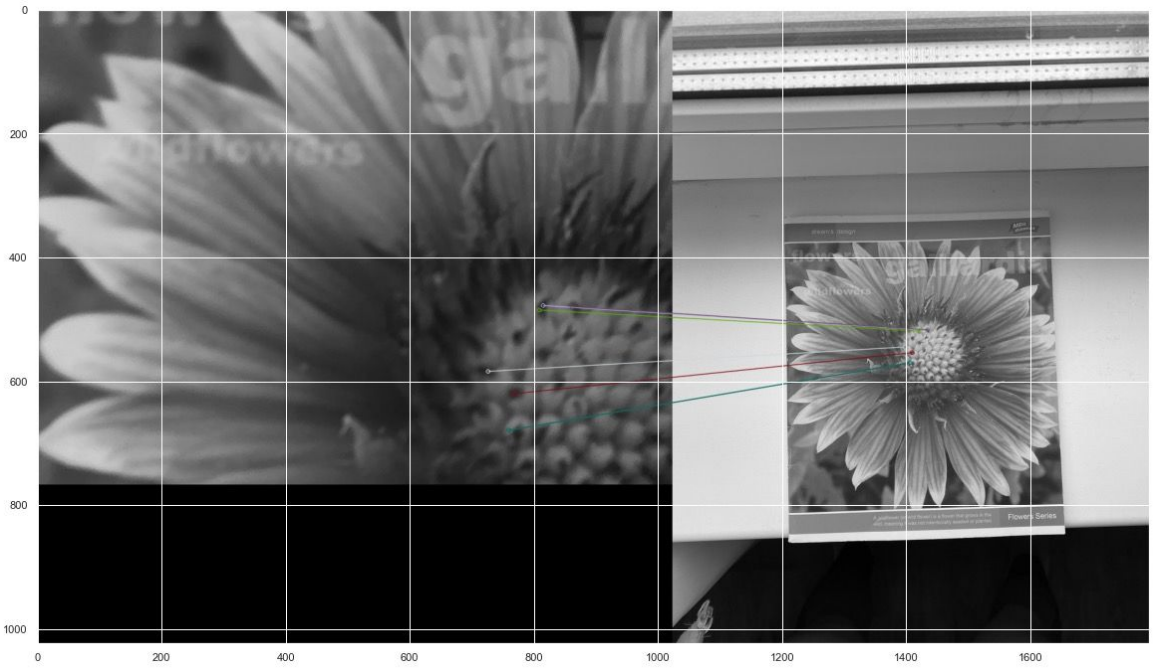


- без потрібного предмету

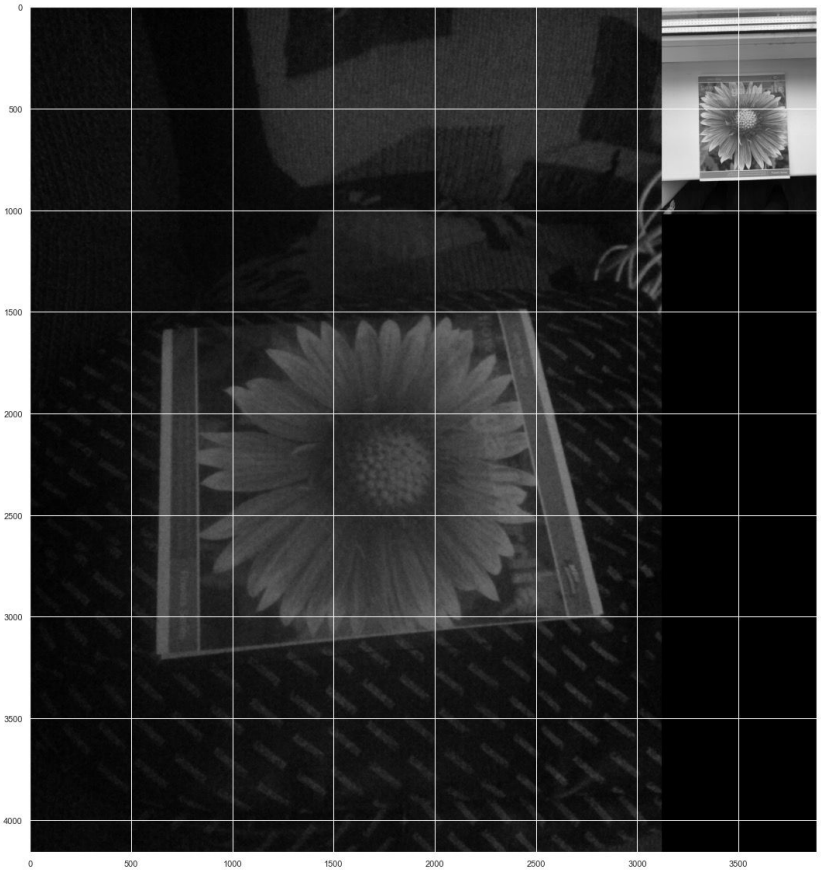


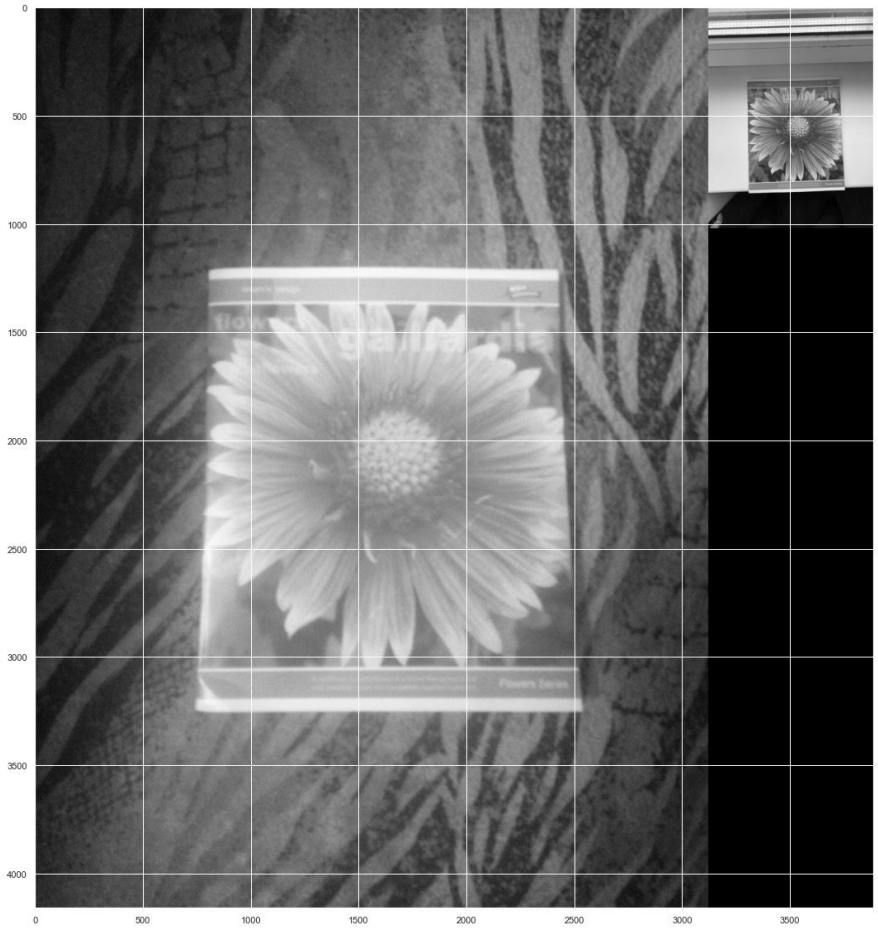


- кропнуте зображення

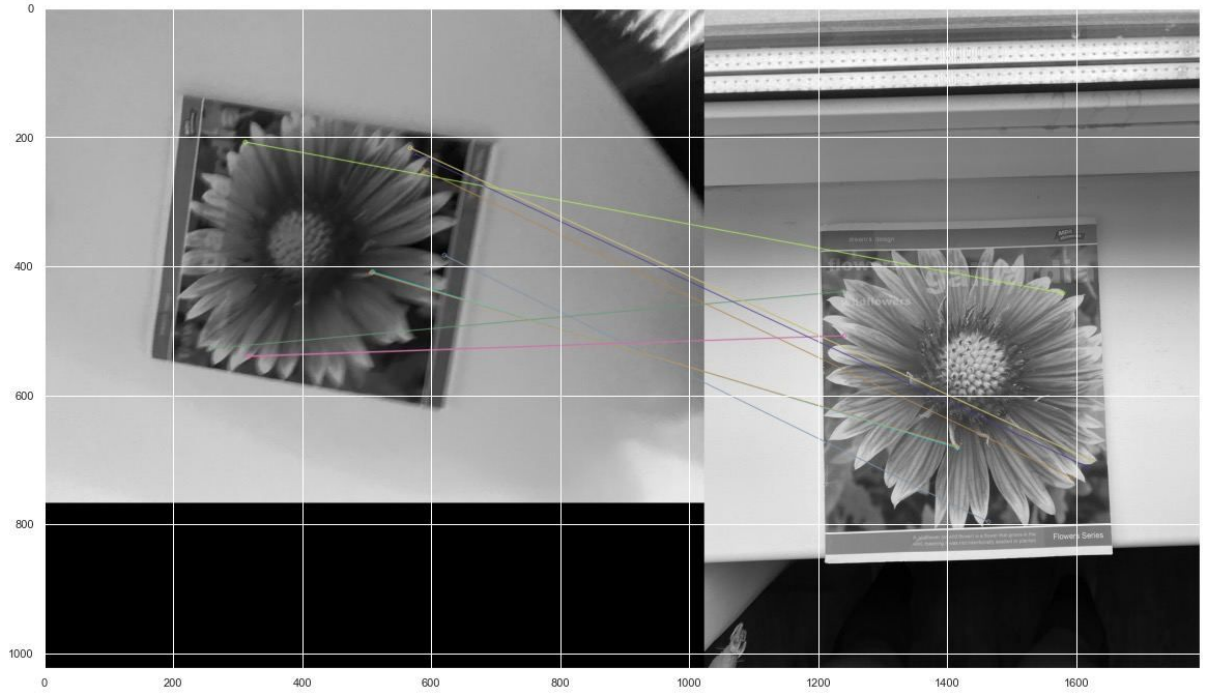


- зашакалене зображення

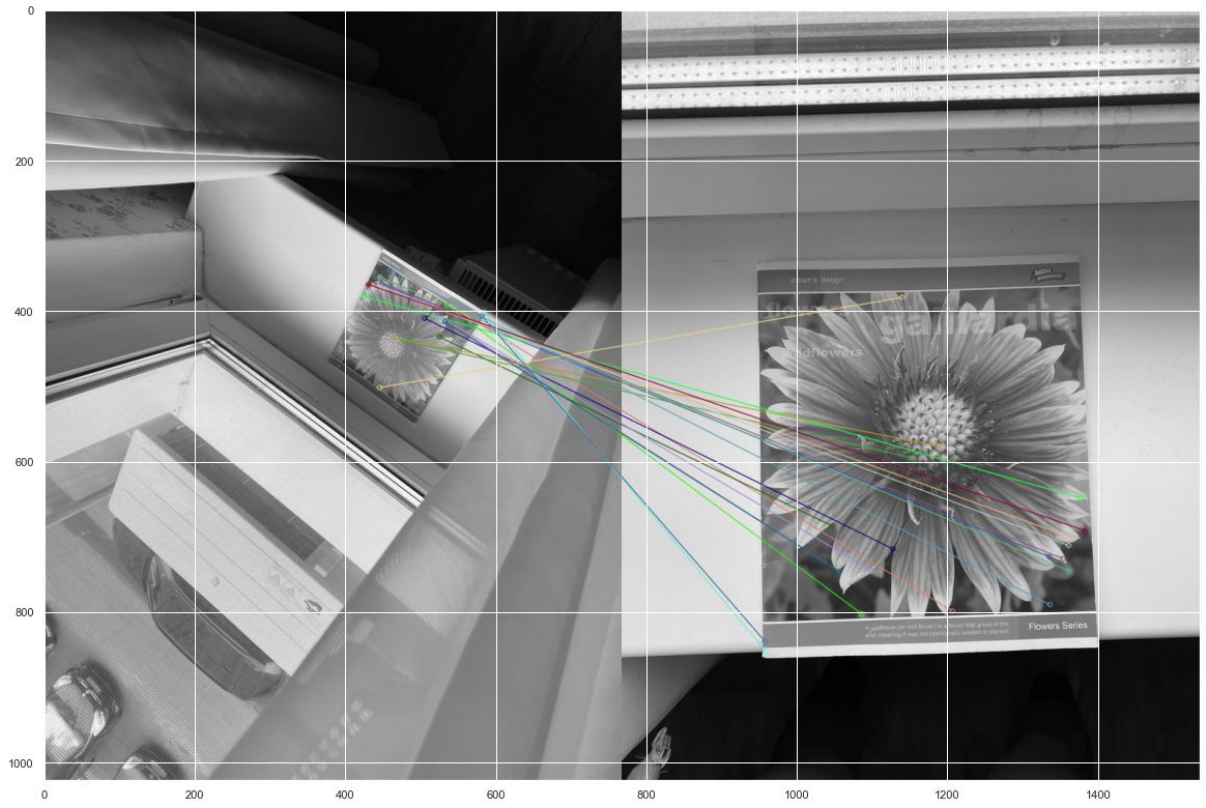




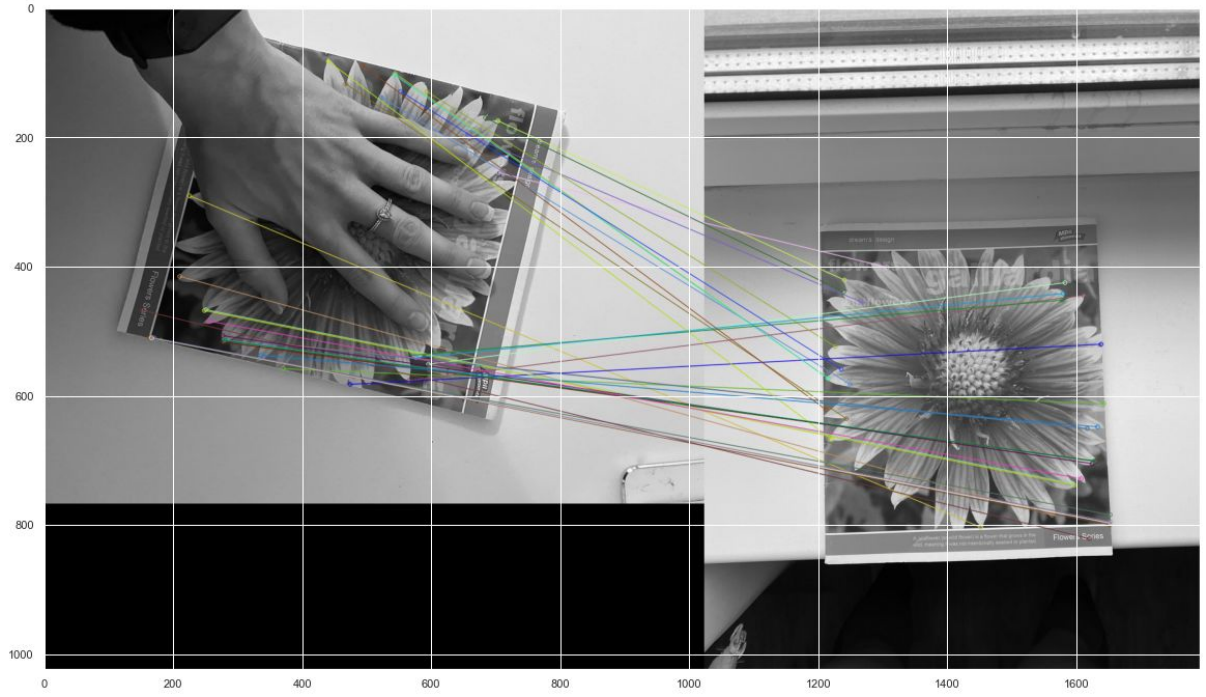
- поворот + тремтіння рук



- віддалено

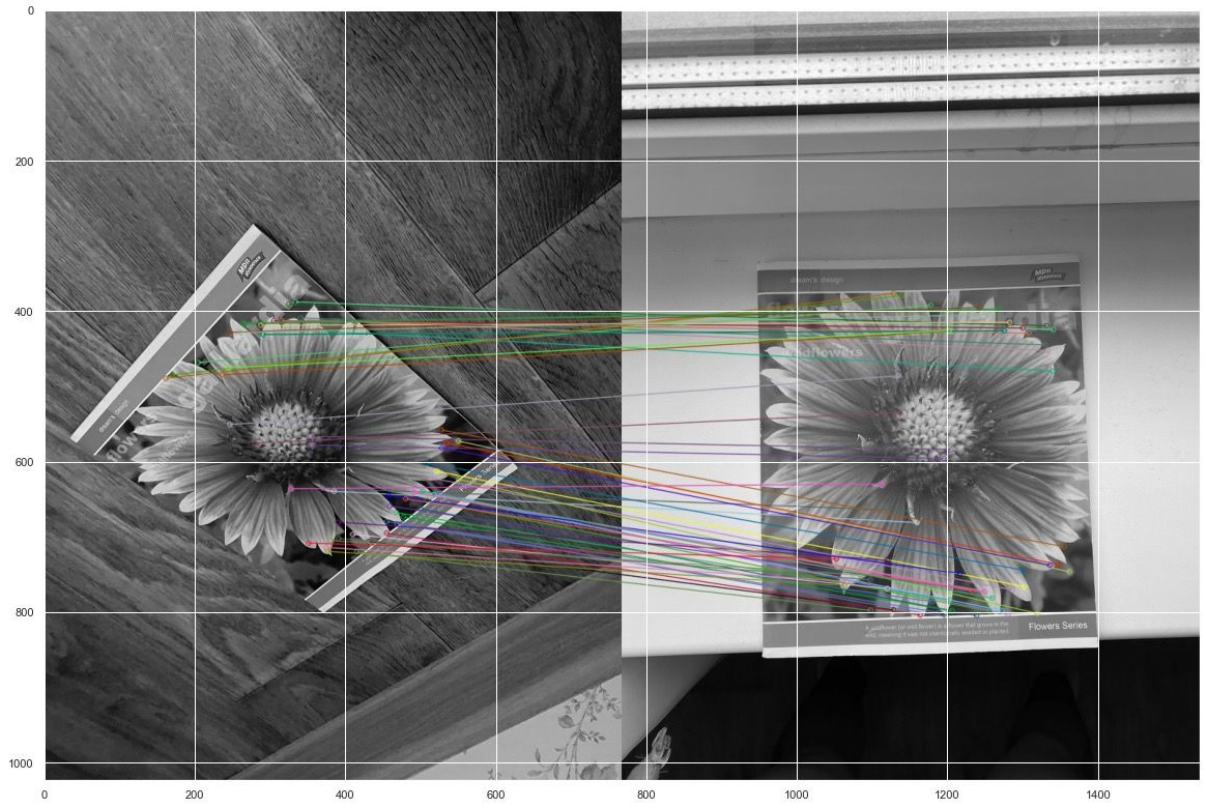


- при перекритті

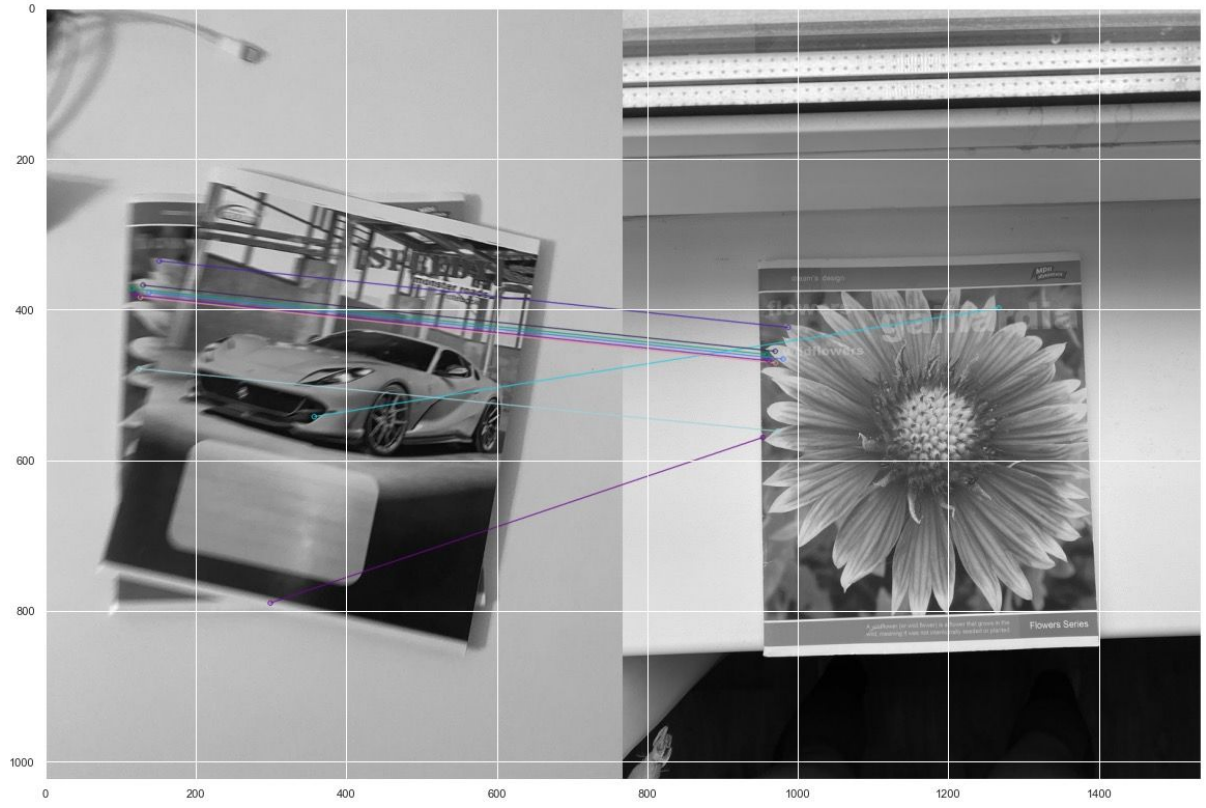


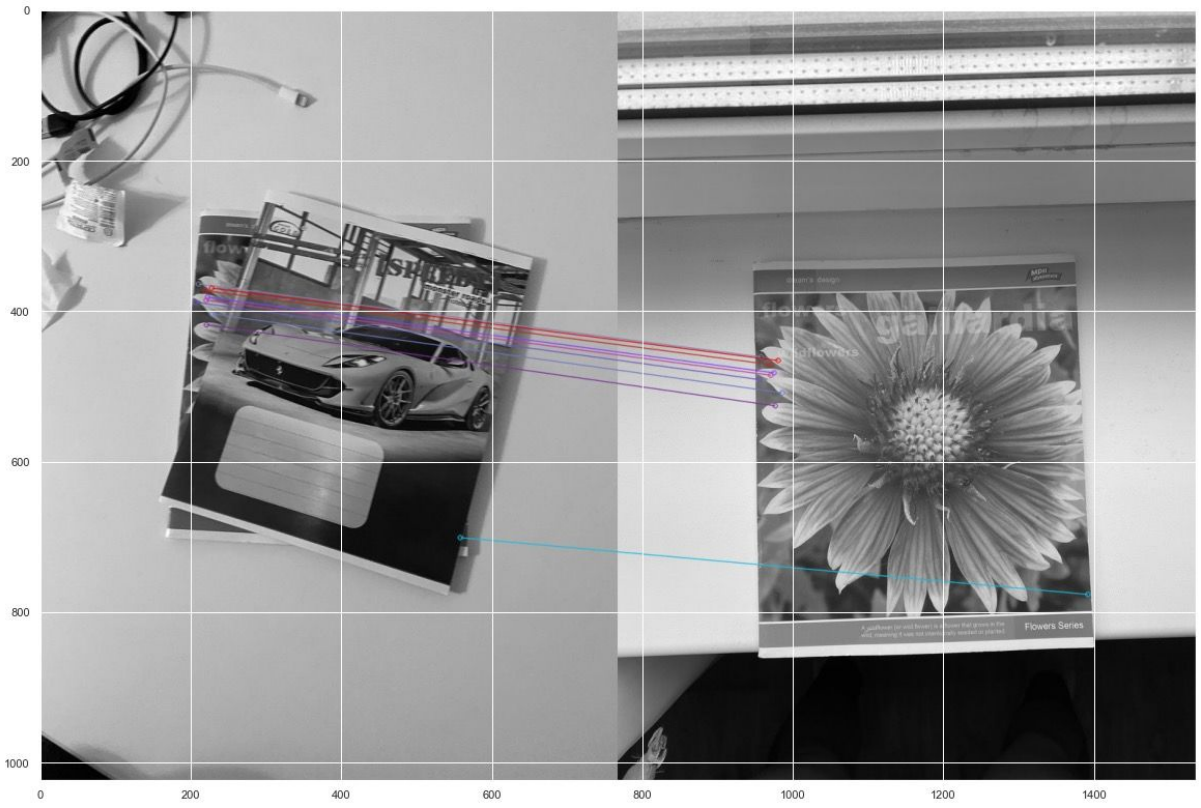
2. AKAZE

- хороший приклад

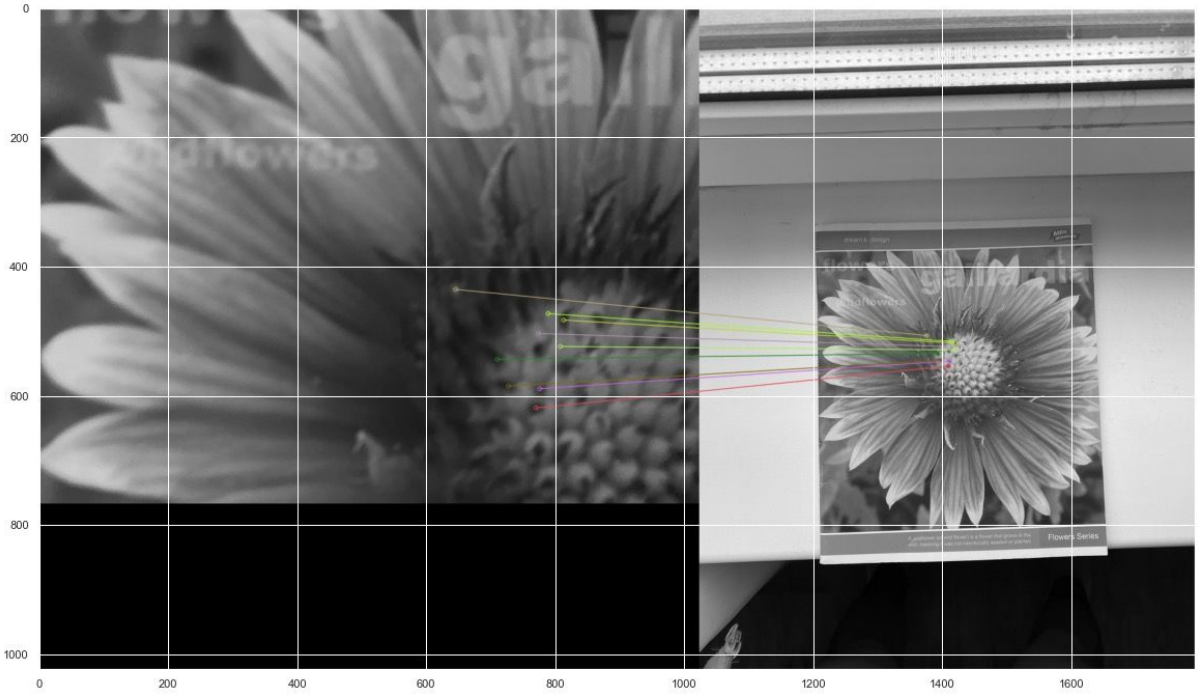


- без потрібного предмету

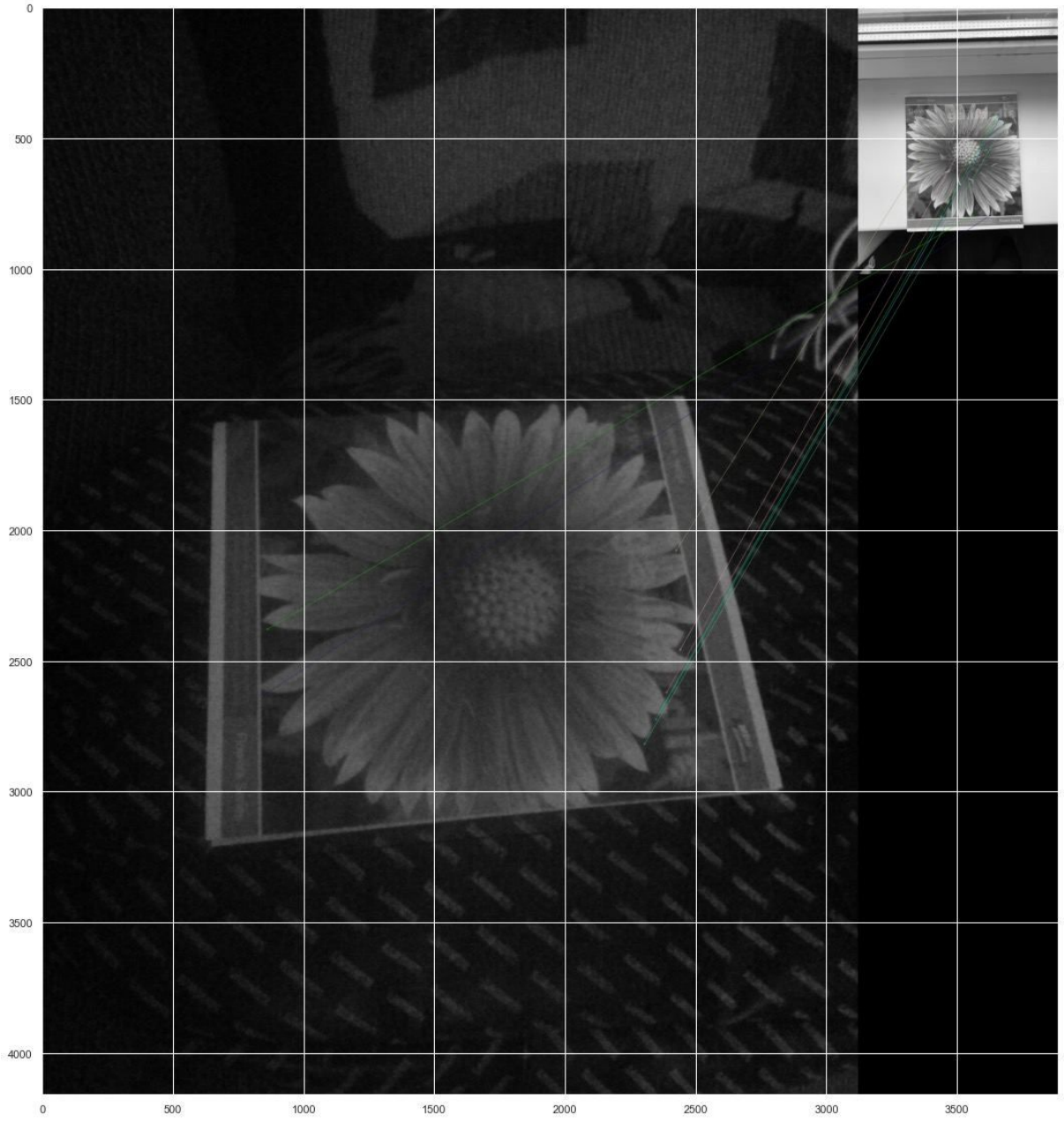




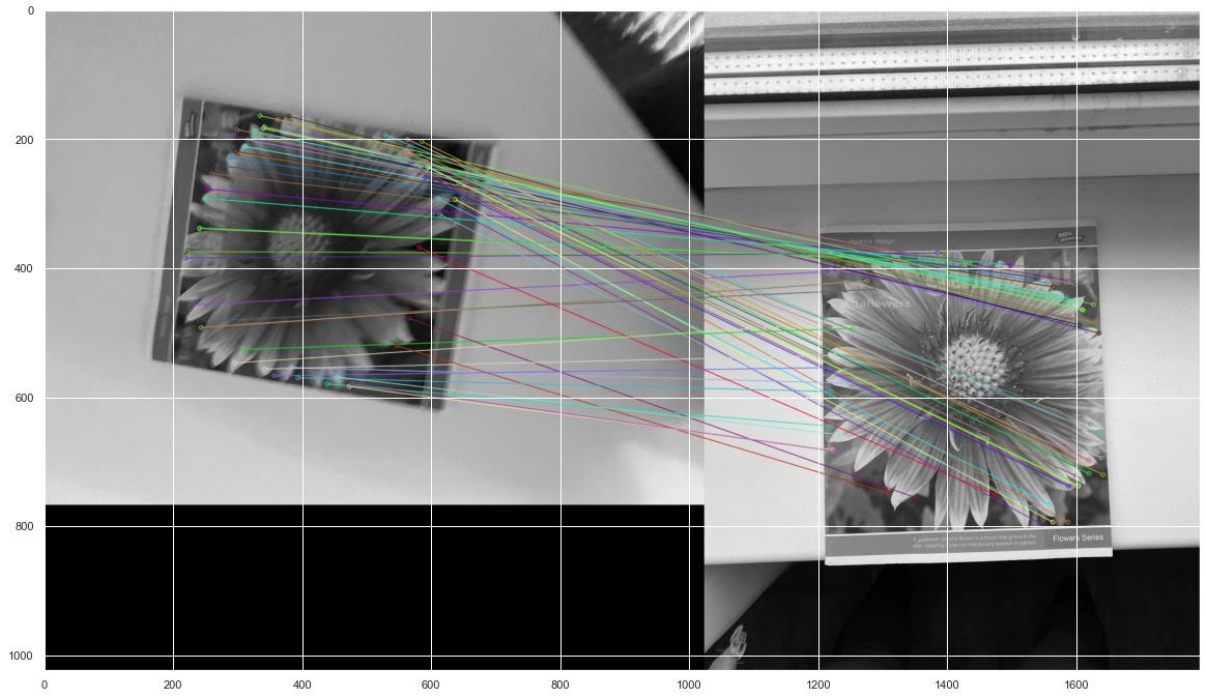
- кропнуте зображення



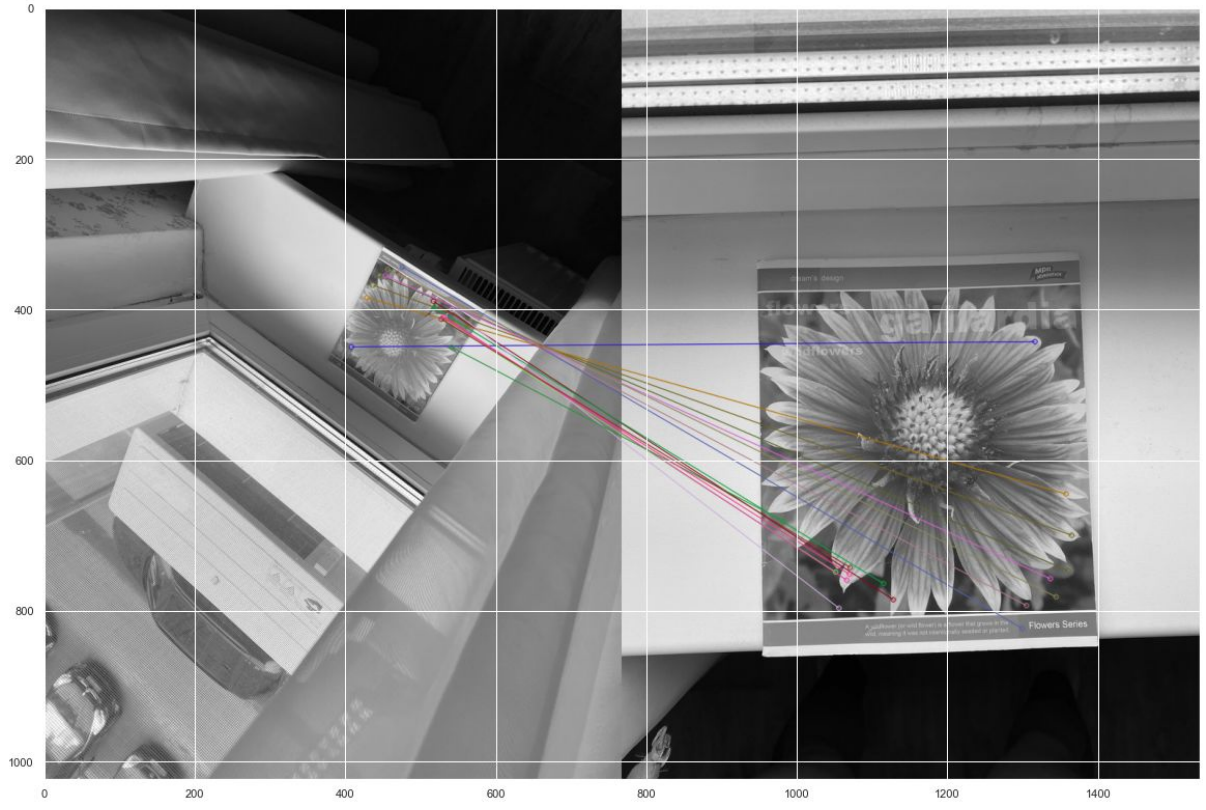
- зашакалене зображення


0
500
1000
1500
2000
2500
3000
3500
4000
0
500
1000
1500
2000
2500
3000
3500

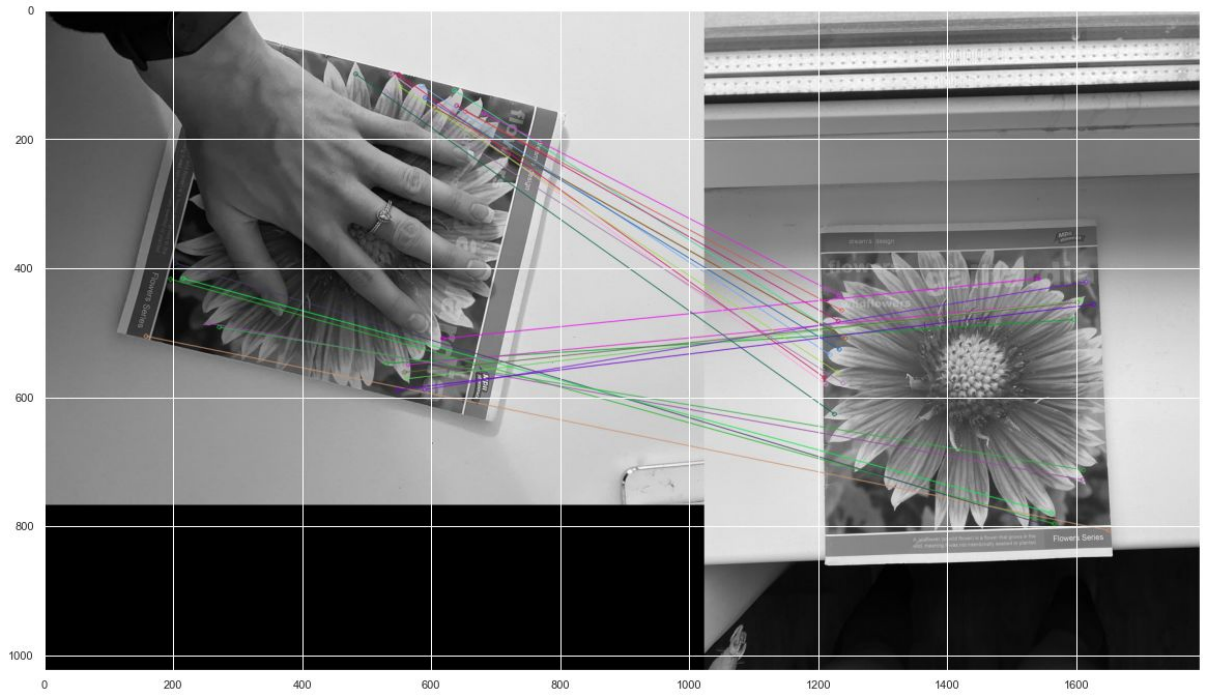
- поворот + тремтіння рук



- віддалено



- при перекритті



Приклади з другим предметом можете побачити у jupyter notebook.

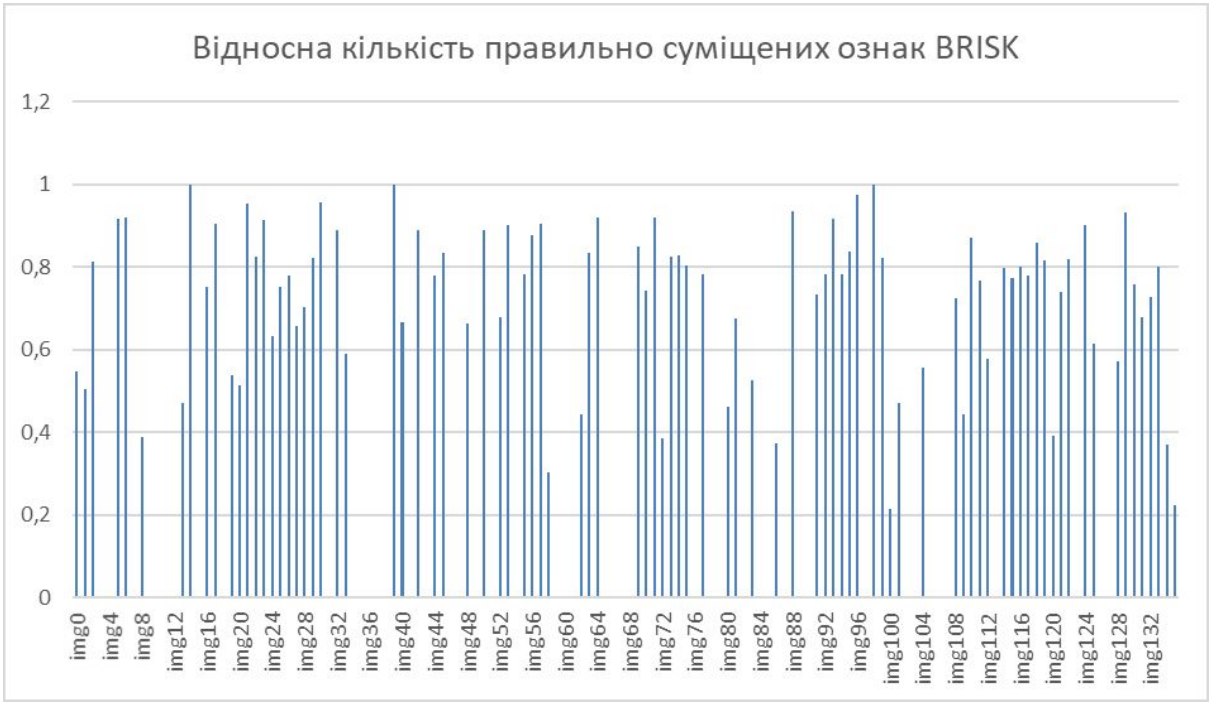
Оцінка метрик

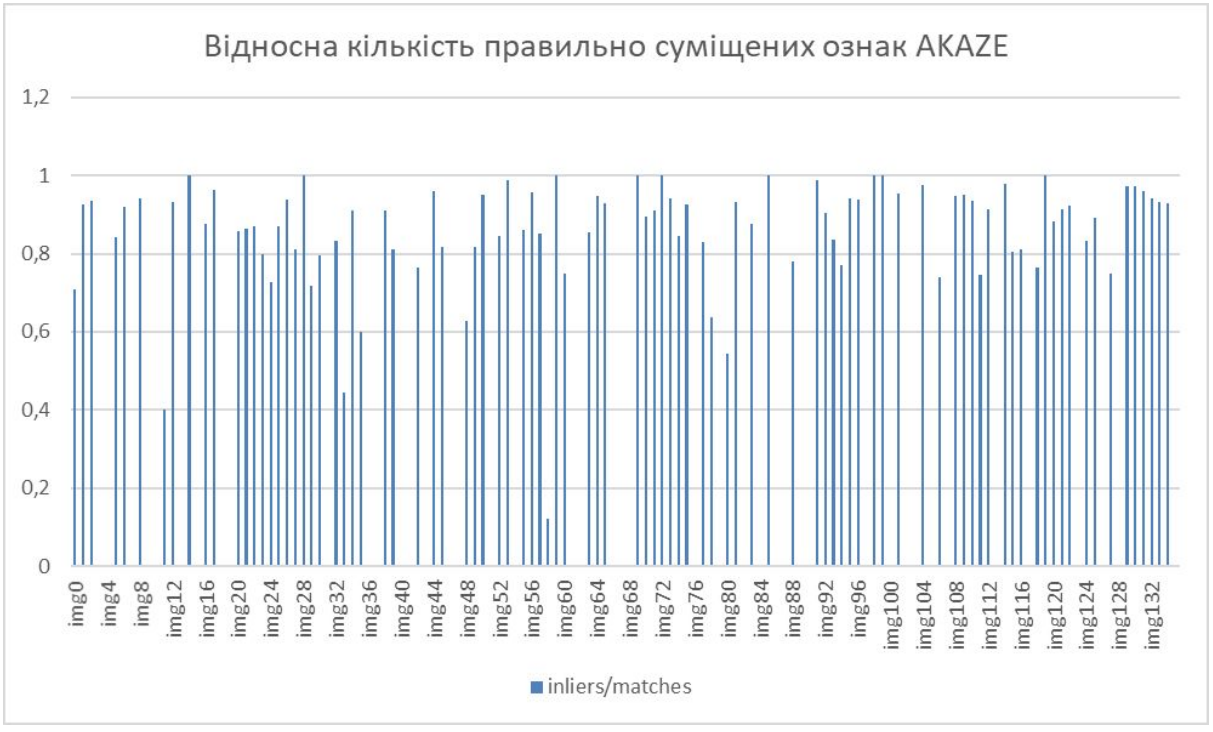
Метрика 1. Відносна кількість правильно суміщених ознак. Маємо ключові точки на першому зображенні та на другому. Об’єктом FlannBasedMatcher отримуємо масив співпадінь matches(для кожної точки еталону шукаємо два найближчих сусіди із тестового зображення), далі підраховуємо кількість співпадінь що пройшли Lowe’s ratio test(m.distance < ratio_thresh*n.distance)(m.distance - перший сусід за методом kNN, n.distance - другий сусід за методом knn).

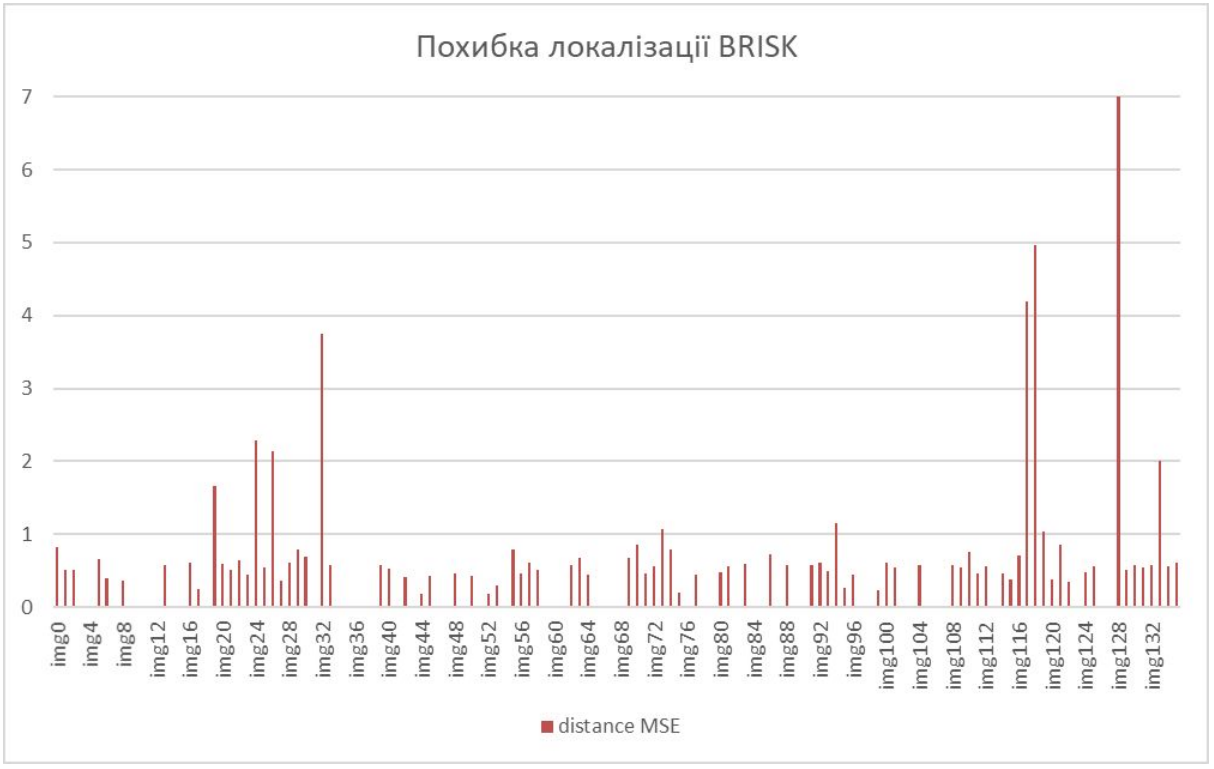
Метрика 2. Похибка локалізації (відстань між реальним розміщенням предмета в кадрі та розпізнаним). Середня відстань між точками “гарних співпадінь” - похибка локалізації.

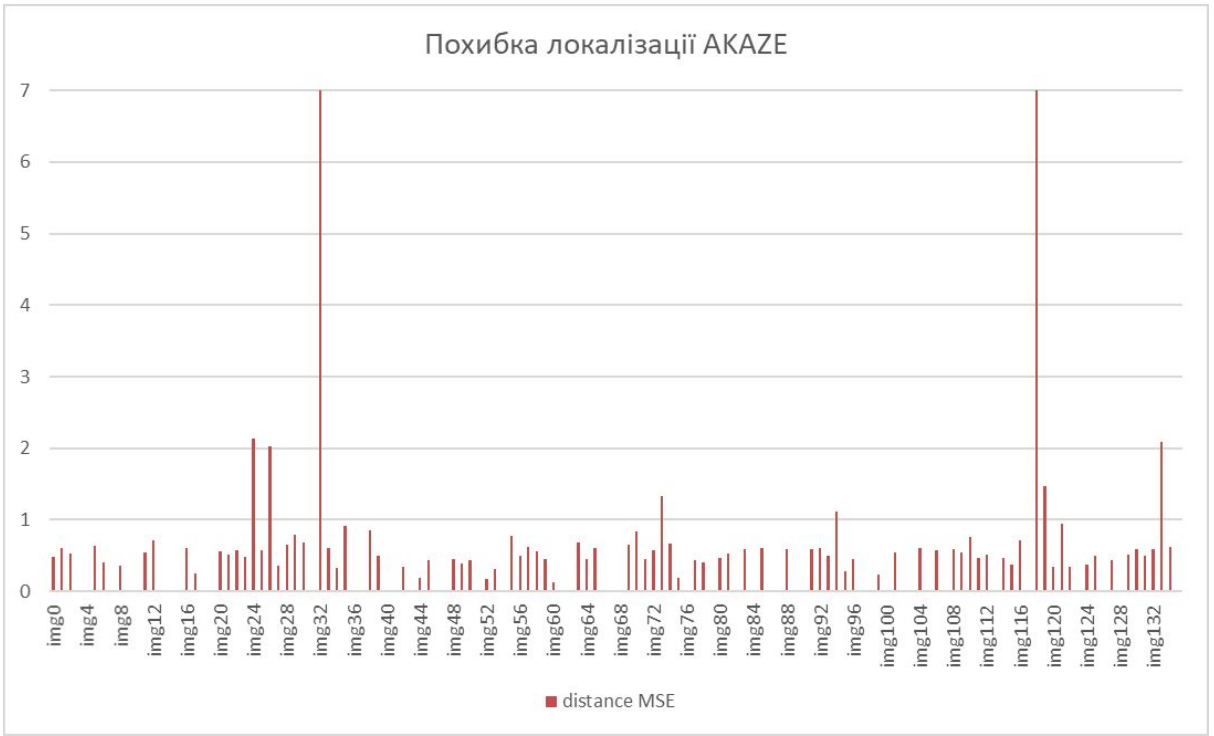
Метрика 3. Час обробки зображення(роботи методу detect_match).

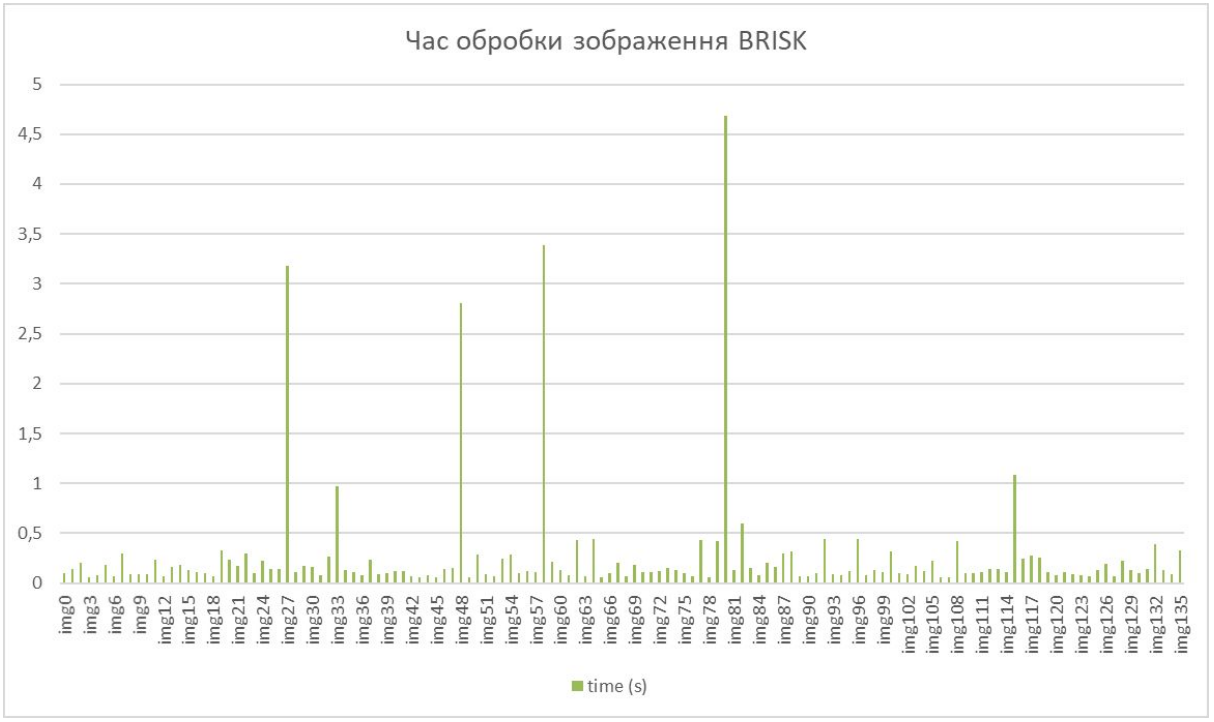
Для зошита маємо:

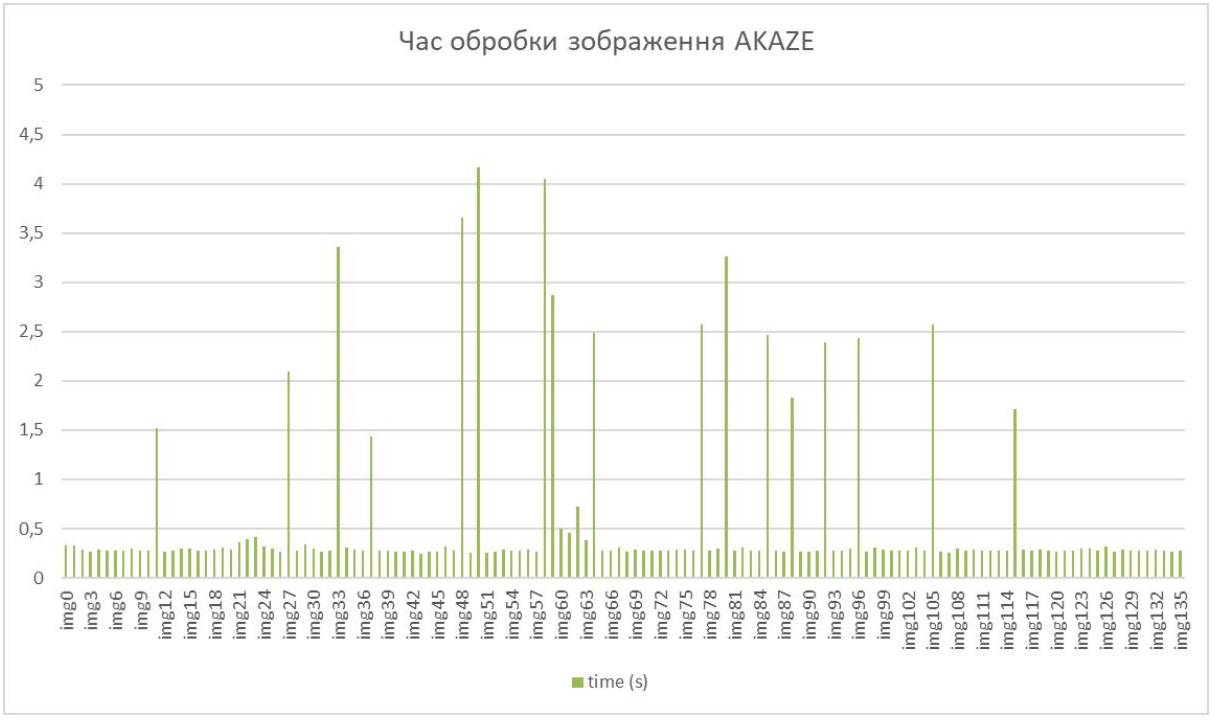

Відносна кількість правильно суміщених ознак BRISK
1,2
1
0,8
0,6
0,4
0,2
0
img0 img4 img8 img12 img16 img20 img24 img28 img32 img36 img40 img44 img48 img52 img56 img60 img64 img68 img72 img76 img80 img84 img88 img92 img96 img100 img104 img108 img112 img116 img120 img124 img128 img132


Відносна кількість правильно суміщених ознак AKAZE
1,2
1
0,8
0,6
0,4
0,2
0
img0 img4 img8 img12 img16 img20 img24 img28 img32 img36 img40 img44 img48 img52 img56 img60 img64 img68 img72 img76 img80 img84 img88 img92 img96 img100 img104 img108 img112 img116 img120 img124 img128 img132
inliers/matches


Похибка локалізації BRISK
7
6
5
4
3
2
1
0
img0
img4
img8
img12
img16
img20
img24
img28
img32
img36
img40
img44
img48
img52
img56
img60
img64
img68
img72
img76
img80
img84
img88
img92
img96
img100
img104
img108
img112
img116
img120
img124
img128
img132
distance MSE

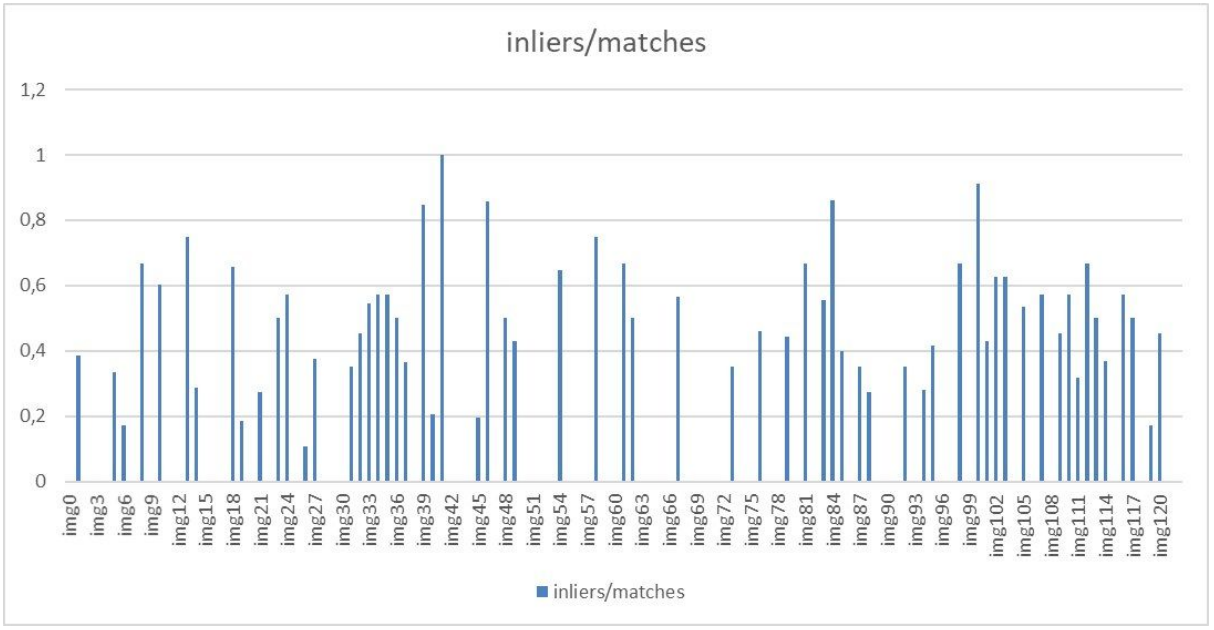

Похибка локалізації AKAZE
7
6
5
4
3
2
1
0
img0
img4
img8
img12
img16
img20
img24
img28
img32
img36
img40
img44
img48
img52
img56
img60
img64
img68
img72
img76
img80
img84
img88
img92
img96
img100
img104
img108
img112
img116
img120
img124
img128
img132
distance MSE



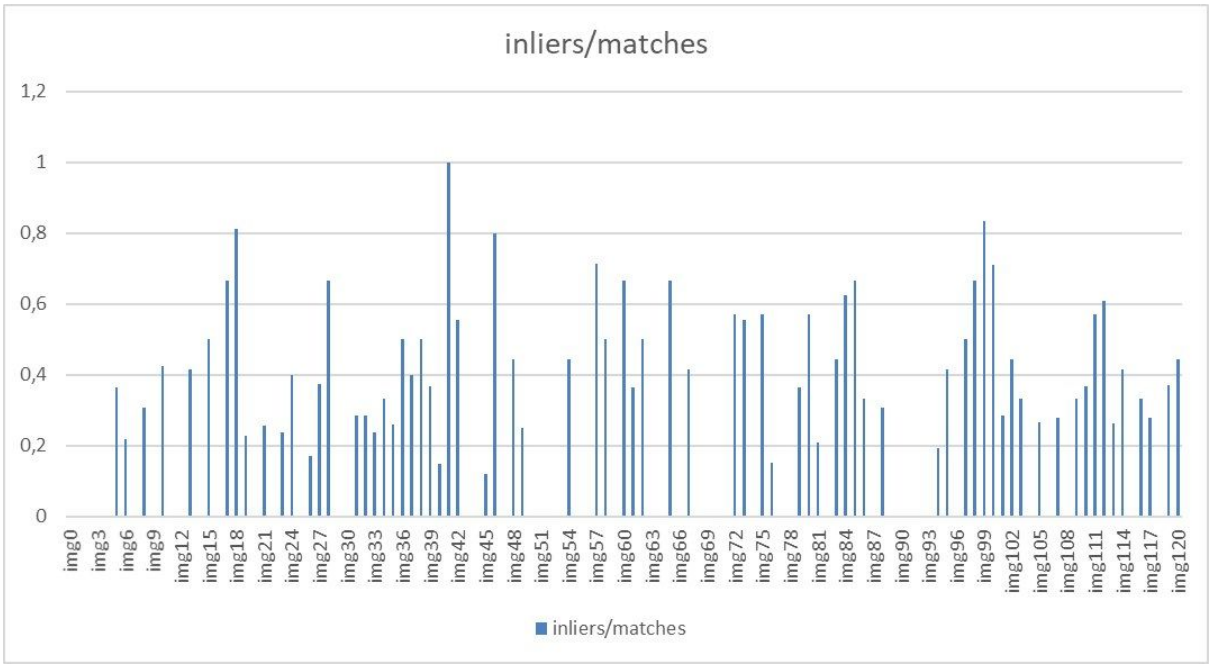


Для машинки:

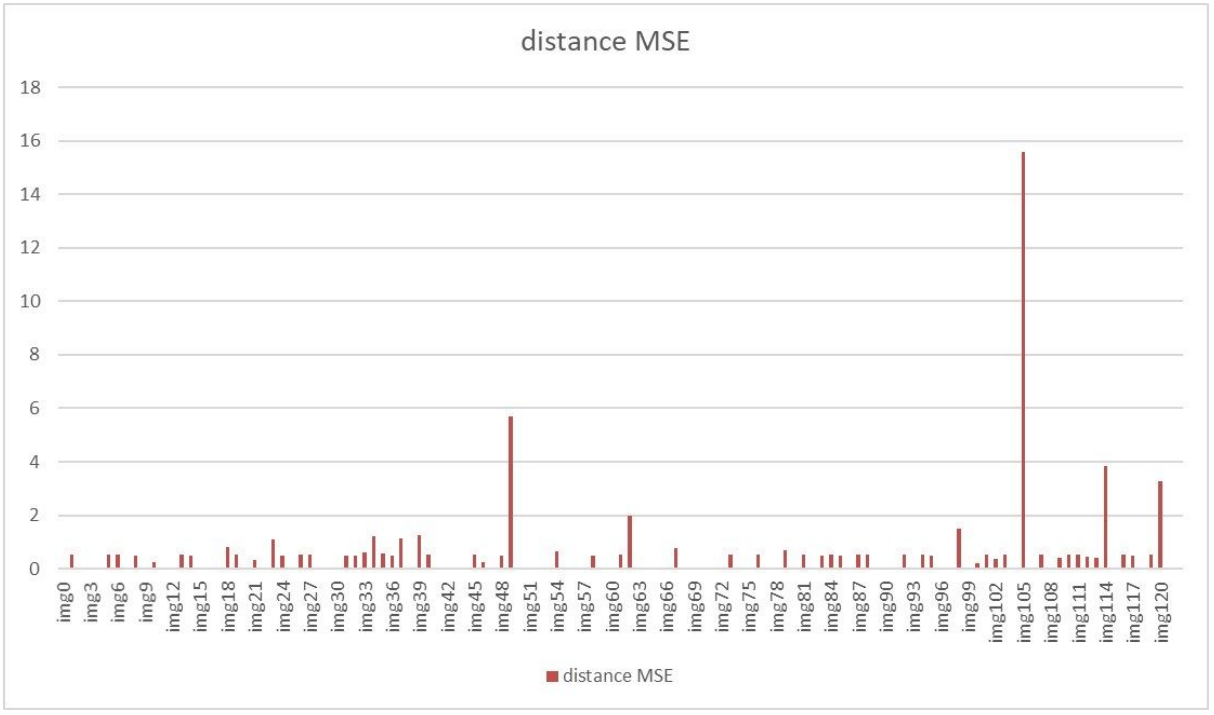
Відносна кількість правильно суміщених ознак BRISK:



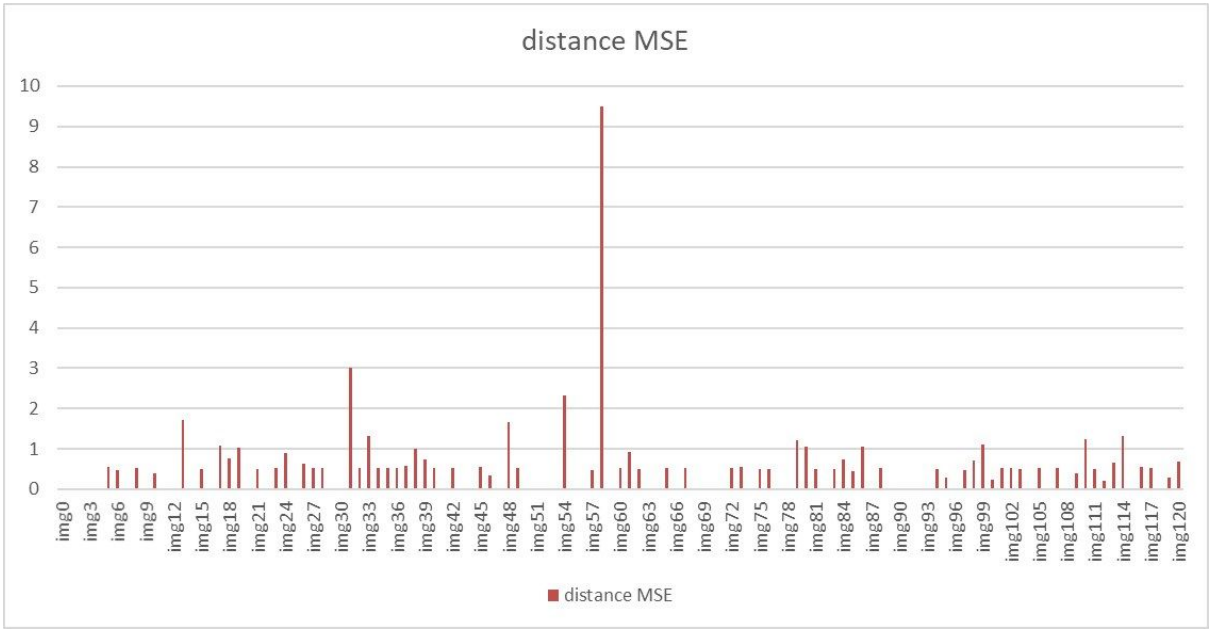
Відносна кількість правильно суміщених ознак AKAZE:



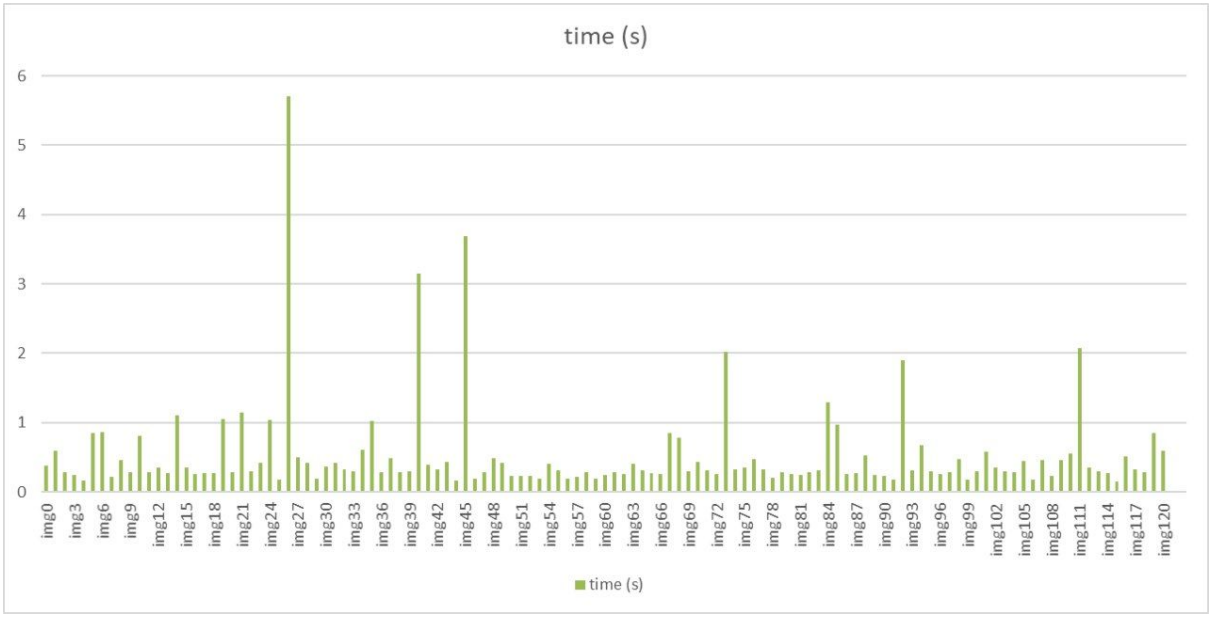
Похибка локалізації BRISK:



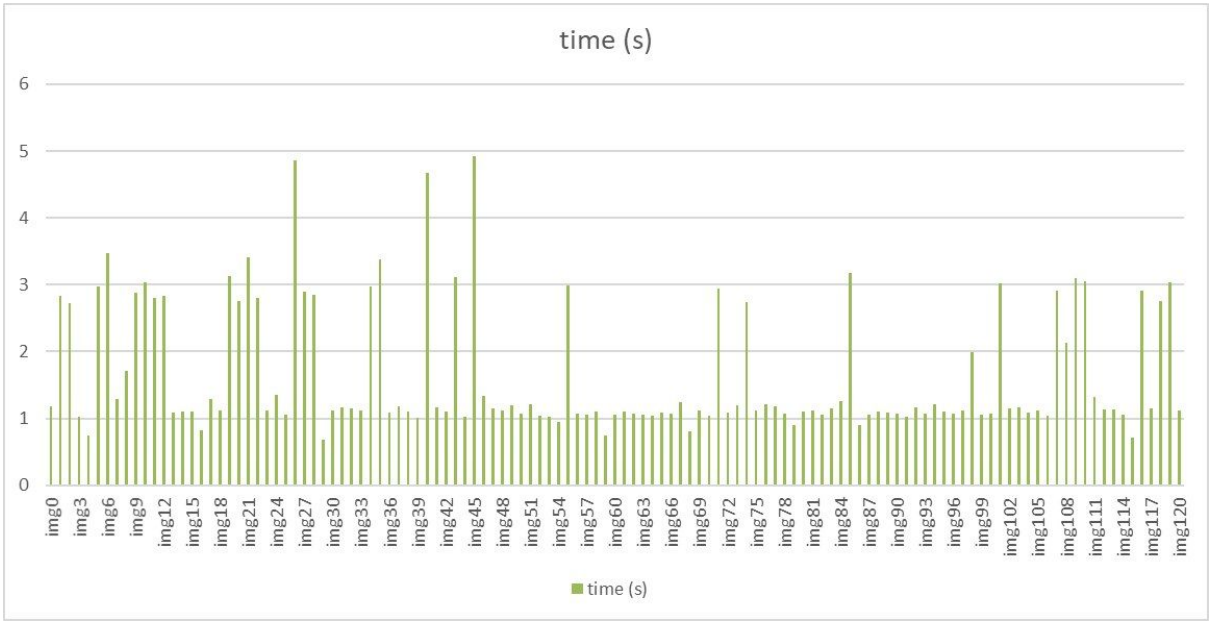
Похибка локалізації AKAZE:



Час обробки зображення BRISK:



Час обробки зображення AKAZE:



Також для зручності приведемо табличку із середніми значеннями метрик по кожному предмету і дескриптору:

		inliers/matches AVG	distance MSE AVG	time (s) AVG
Зошит	BRISK	0,507951558	1,000432475	0,270259126
	AKAZE	0,638386577	0,798801165	0,586326086
Машинка	BRISK	0,30435813	1,010400394	0,531109803
	AKAZE	0,287940816	0,822433962	1,63974085

Висновки

1. AKAZE краще за BRISK здатен впоратися з фото поганої якості або тими, що містять шуми, розмитими фото та в цілому точність виявлення є вищою. Через особливості його структури, таких як пошук особливих точок на нелінійній багатомасштабній піраміді і опис дескрипторів по трьом параметрам, замість одного, як у BRISK, отримуємо високу точність при зіставленні зображень. Він є кращим для зображень зошита.
2. BRISK працює набагато швидше, тому за гарних умов фото його неодмінно можна вважати ефективнішим. BRISK відрізняється від інших методів тим, що він визначає найбільшу кількість особливих точок, але, на жаль, в них потрапляє і цифровий шум.
3. BRISK гарно спрацьовує з текстурами, об'ємними зображеннями, він показав себе краще за AKAZE на прикладі машинки.
4. При поворотах чи перекритті зображення обидва дескриптори показують себе добре.
5. Обидва дескриптори краще впоралися з зображеннями плоского предмета.

Загалом ми очікували гірших результатів ніж отримали під час виконання роботи.