Домашна задача 5

Препознавање на слики од постери со SIFT

Во директориумот Database се дадени слики од 22 постери. Сликите hw7_poster_1.jpg, hw7_poster_2.jpg, и hw7_poster_3.jpg се прашални слики.

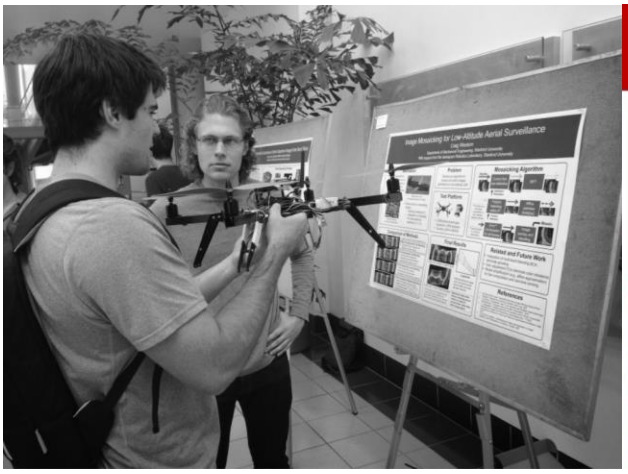
За секоја прашална слика искористете го следниот алгоритам за да ја одредите најсличната слика (сликата која го содржи соодветниот постер) од сликите зададени во директориумот Database:

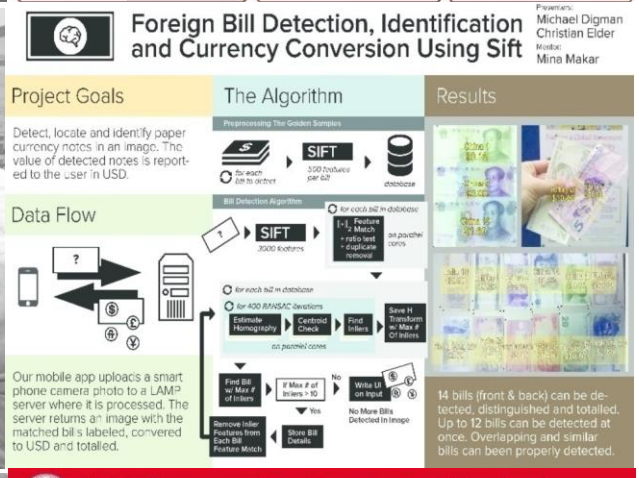
- Генерирајте SIFT дескриптори за прашалната слика
- Со користење на пребарување со најблиски соседи споредете ги генерираните SIFT дескриптори на прашалната слика со SIFT дескрипторите на сликите од директориумот Database
- За локалните дескриптори кои се совпаднале пронајдете inliers со користење на RANSAC и хомографија
- Прикажете ја сликата со најголем број на inliers која всушност е најслична на прашалната слика

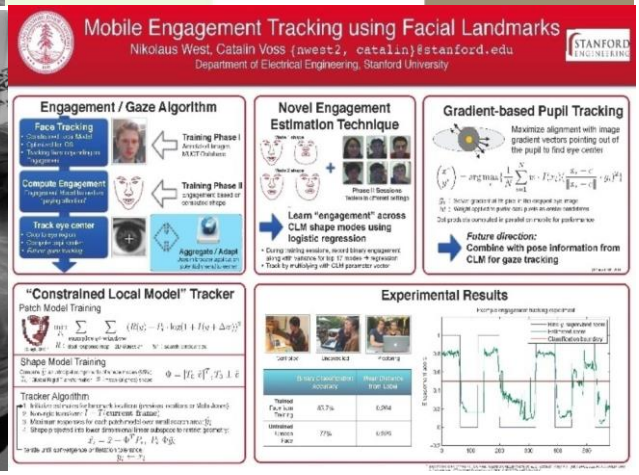
Решение:

```
import cv2
import numpy as np
import os

def find_best_match(query_descriptors, database_descriptors, query_keypoints,
database_keypoints):
    matcher = cv2.BFMatcher()
    matches = matcher.knnMatch(query_descriptors, database_descriptors, k=2)

    good_matches = []
    for m, n in matches:
        if m.distance < 0.75 * n.distance:
            good_matches.append(m)

    if len(good_matches) > 10:
        src_pts = np.float32([query_keypoints[m.queryIdx].pt for m in
good_matches]).reshape(-1, 1, 2)
        dst_pts = np.float32([database_keypoints[m.trainIdx].pt for m in
good_matches]).reshape(-1, 1, 2)

        M, mask = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, 5.0)
        if M is not None:
            matches_mask = mask.ravel().tolist()
            inliers = sum(matches_mask)
            return inliers

    return 0


directory = os.getcwd()

query_images = ["hw7_poster_1.jpg", "hw7_poster_2.jpg", "hw7_poster_3.jpg"]


database_images = sorted([file for file in os.listdir("Database") if
file.lower().endswith(('.png', '.jpg', '.jpeg'))])


for database_image_path in database_images:
    image = cv2.imread(os.path.join(directory, "Database",
database_image_path))
    image = cv2.resize(image, (800, 600))
    cv2.imwrite(os.path.join(directory, "Database", database_image_path),
image)

for query_image_path in query_images:
    image = cv2.imread(os.path.join(directory, query_image_path))
    image = cv2.resize(image, (800, 600))
    cv2.imwrite(os.path.join(directory, query_image_path), image)

sift = cv2.SIFT_create()
```

```
for query_image_path in query_images:

    query_image = cv2.imread(os.path.join(directory, query_image_path),
cv2.IMREAD_GRAYSCALE)


    query_keypoints, query_descriptors = sift.detectAndCompute(query_image,
None)

    best_match_image = None
    best_match_inliers = 0

    for database_image_path in database_images:

        database_image = cv2.imread(os.path.join(directory, "Database",
database_image_path), cv2.IMREAD_GRAYSCALE)


        database_keypoints, database_descriptors =
sift.detectAndCompute(database_image, None)


        inliers = find_best_match(query_descriptors, database_descriptors,
query_keypoints, database_keypoints)


        if inliers > best_match_inliers:
            best_match_inliers = inliers
            best_match_image = database_image_path


    best_match = cv2.imread(os.path.join(directory, "Database",
best_match_image))
    cv2.imshow("Query Image", query_image)
    cv2.imshow("Best Match", best_match)
    cv2.waitKey(0)

cv2.destroyAllWindows()
```







Изработил:

Бојан Ристов (211151)