تمرین سری ۳ - سوال ۲

محمدجواد شریعتی 96100414

```python
im1 = cv2.imread('../resources/01.JPG')
im2 = cv2.imread('../resources/02.JPG')

# RGB -> GRAY
im1_gray = cv2.cvtColor(im1, cv2.COLOR_BGR2GRAY)
im2_gray = cv2.cvtColor(im2, cv2.COLOR_BGR2GRAY)

# Create SIFT detector
sift = cv2.SIFT_create()

# Find interest points
interest_points_1 = sift.detect(im1_gray, None)
interest_points_2 = sift.detect(im2_gray, None)

# Compute Descriptors
_, descriptor1 = sift.compute(im1_gray, interest_points_1)
_, descriptor2 = sift.compute(im2_gray, interest_points_2)

# Use BFMatcher to match points between two images
bf_matcher = cv2.BFMatcher()
match_points = bf_matcher.knnMatch(descriptor1, descriptor2, k=2)

# Ratio between most similar and second most similar should less than 0.9
final_match_points = []
for m, n in match_points:
    if m.distance / n.distance < 0.8:
        final_match_points.append(m)

points1 = []
points2 = []
for match in final_match_points:
    points1.append(interest_points_1[match.queryIdx].pt)
    points2.append(interest_points_2[match.trainIdx].pt)
```

ابتدا با استفاده از SIFT نقاط interest تصویر را بدست می‌آورم و سپس descriptor آن‌ها را محاسبه می‌کنم و با استفاده از BFMatcher نقاط متناظر دو عکس را بدست می‌آورم (k=2 یعنی دوتا بهترین نقطه- برای انجام ratio test). سپس با کمک ratio test دقت نقاط بدست آمده را بهتر می‌کنم (با ضریب ۰.۸). این کد را از تمرین های قبلی خودم کپی کردم.

```python
F, mask = cv2.findFundamentalMat(points1, points2, cv2.FM_RANSAC)
print("Fundamental Matrix: ")
print(F, "\n")

# Separate inliers and outliers
inliers1 = points1[mask.ravel() == 1]
inliers2 = points2[mask.ravel() == 1]
outliers1 = points1[mask.ravel() == 0]
outliers2 = points1[mask.ravel() == 0]

# show inliers and outliers
im1_copy = copyof(im1)
im2_copy = copyof(im2)
draw_points(im1_copy, inliers1, color=GREEN)
draw_points(im1_copy, outliers1, color=RED)
draw_points(im2_copy, inliers2, color=GREEN)
draw_points(im2_copy, outliers2, color=RED)
cv2.imwrite("out/res05.jpg", np.concatenate((im1_copy, im2_copy), axis=1))
```

در مرحله بعدی با استفاده از تابع آماده findFundamentalMat ماتریس فاندامنتال را با روش RANSAC مطابق خواسته سوال بدست می‌آوریم و آن را نمایش می‌دهم:

```
Fundamental Matrix:
[[ 2.19955731e-09 -7.00058802e-08 -1.64946063e-04]
 [-6.15221831e-08  1.91816246e-08 -1.39798870e-03]
 [-1.71020525e-04  1.62924367e-03  1.00000000e+00]]
```

در مرحله بعد خواسته شده که نقاط inlier و outlier را نمایش دهم که این کار با mask که تابع findFundamentalMat به ما می‌دهد به راحتی انجام می‌شود. این mask در واقع می‌گوید هرکدام از نقاط ما (هرکدام از تناظرهای ما در واقع) براساس ماتریس فاندامنتال بدست آمده inlier هستند یا نه (در آن صدق می‌کنند یا نه)

در مرحله بعدی epipol ها را بدست می‌آورم. برای این‌کار مشابه آنچه در صورت سوال خواسته شده عمل می‌کنم. یعنی e را پیدا می‌کنم که Fe=0. برای پیدا کردن e هم SVD ماتریس F را محاسبه می‌کنم و ستون آخر V می‌شود e ما.

```python
# Find Epipols (Epipolars Points)
e1 = find_Epipol(F)
print("e1: ")
print(e1, "\n")
plt.imshow(cv2.cvtColor(im1, cv2.COLOR_BGR2RGB))
plt.scatter([e1[0]], [e1[1]])
plt.savefig("out/res06.jpg")
plt.clf()

e2 = find_Epipol(F.T)
print("e2: ")
print(e2, "\n")
plt.imshow(cv2.cvtColor(im2, cv2.COLOR_BGR2RGB))
plt.scatter([e2[0]], [e2[1]])
plt.savefig("out/res07.jpg")
```

```python
def find_Epipol(F):
    U, S, V_T = svd(F)
    e = V_T[-1].reshape(3, 1)
    return e / e[2]
```

```python
DESIRED_POINTS_COUNT = 10
points_count = len(inliers1)
category_count = points_count // DESIRED_POINTS_COUNT
for i in range(DESIRED_POINTS_COUNT):
    random_point_idx = random.randrange(i * category_count, (i + 1) * category_count)
    random_color = tuple(np.random.randint(0, 255, 3).tolist())

    point1 = np.array([
        [inliers1[random_point_idx][0]],
        [inliers1[random_point_idx][1]],
        [1]
    ])
    draw_points(im1, [inliers1[random_point_idx]], random_color)
    # corresponding line to point1 on im2
    l1 = F @ point1
    l1 /= np.sqrt(sum(l1[:2] ** 2))
    draw_epipolar_line(im2, l1, random_color)

    point2 = np.array([
        [inliers2[random_point_idx][0]],
        [inliers2[random_point_idx][1]],
        [1]
    ])
    draw_points(im2, [inliers2[random_point_idx]], random_color)
    # corresponding line to point2 on im1
    l2 = F.T @ point2
    l2 /= np.sqrt(sum(l2[:2] ** 2))
    draw_epipolar_line(im1, l2, random_color)

cv2.imwrite("out/res08.jpg", np.concatenate((im1, im2), axis=1))
```

در مرحله آخر هم برای کشیدن ۱۰ خط epiline ، به صورت رندوم و پخش این کار را می‌کنم. یعنی ابتدا بدست می‌آورم که کلا چند نقطه داریم، مثلا اگر ۵۰ تا داریم، از ۵ تای اول یکی را انتخاب می‌کنم و epiline متناظر آن را می‌کشم. سپس از ۵تای دوم یکی را رندوم انتخاب می‌کنم و epiline متناظر آن را می‌کشم و به همین ترتیب تا ۵تای آخر. خط متناظر یک نقطه هم از همان رابطه ای که خواندیم یعنی

$$l = F * x$$

که x نقطه ما و F ماتریس فاندامنتال است. سپس خط بدست آمده را نرمالایز می‌کنم و درنهایت آن‌را رسم می‌کنم. نقاط متناظر و خط های متناظر این نقاط همه با یک رنگ یکسان (و رندوم) کشیده می‌شوند.