Assignment 3
Tudorache Vlad Adrian 442C

Assignment 3

Subsection 1:

For the first section my assigned set_x was set06.

```
import numpy as np
import cv2 as cv2
import matplotlib.pyplot as plt

# SUBSECTION 1 ------------------------------------------------------------
--------

img1 =
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_x\set06\shanghai-
02.png", 0)
img2 =
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_x\set06\shanghai-
03.png", 0)
img3 =
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_x\set06\shanghai-
04.png", 0)
img4 =
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_x\set06\shanghai-
05.png", 0)

orbObj = cv2.ORB_create()  # Create ORB Object
kp1, descriptors1 = orbObj.detectAndCompute(img1, None)  # Computing
descriptors and keypoints
kp2, descriptors2 = orbObj.detectAndCompute(img2, None)
kp3, descriptors3 = orbObj.detectAndCompute(img3, None)
kp4, descriptors4 = orbObj.detectAndCompute(img4, None)

im1 = cv2.drawKeypoints(img1, kp1, img1)
im2 = cv2.drawKeypoints(img2, kp2, img2)
im3 = cv2.drawKeypoints(img3, kp3, img3)
im4 = cv2.drawKeypoints(img4, kp4, img4)

plt.figure()
plt.imshow(im1);plt.title("shanghai-02 - keypoints")
plt.show()
plt.figure()
plt.imshow(im2);plt.title("shanghai-03 - keypoints")
plt.show()
plt.figure()
plt.imshow(im3);plt.title("shanghai-04 - keypoints")
plt.show()
plt.figure()
plt.imshow(im4);plt.title("shanghai-05 - keypoints")
plt.show()
```

Assignment 3
Tudorache Vlad Adrian 442C

```
matcher =
cv2.DescriptorMatcher_create(cv2.DESCRIPTOR_MATCHER_BRUTEFORCE_HAMMING)

matches43 = matcher.match(descriptors4, descriptors3, None)
matches43 = sorted(matches43, key=lambda x: x.distance, reverse=False)  #
Sorting Matches based on distance

GOOD_MATCH_PERCENT = 0.15
numGoodMatches43 = int(len(matches43) * GOOD_MATCH_PERCENT)  # Choosing only
the first 15%
matches43 = matches43[:numGoodMatches43]


imgMatches43 = cv2.drawMatches(img4, kp4, img3, kp3, matches43, None)

plt.figure(2)
plt.imshow(imgMatches43, cmap='gray')
plt.show()


points4 = np.zeros((len(matches43), 2), dtype=np.float32)
points3 = np.zeros((len(matches43), 2), dtype=np.float32)

for i, match in enumerate(matches43):
    points4[i, :] = kp4[match.queryIdx].pt
    points3[i, :] = kp3[match.trainIdx].pt


h1, mask1 = cv2.findHomography(points3, points4, cv2.RANSAC)


im4Height, im4Width = img4.shape
im3Height, im3Width = img3.shape

im3Aligned = cv2.warpPerspective(img3, h1, (im3Width+im4Width, im3Height))

plt.figure(3)
plt.imshow(im3Aligned, cmap='gray')
plt.show()

# Stitch Image 1 with aligned image 2
stitchedImage43 = np.copy(im3Aligned)
stitchedImage43[0:im4Height,0:im4Width] = img4

plt.figure(4)
plt.imshow(stitchedImage43, cmap='gray')
plt.show()

#43 si 2-----------------------------------------------------------------------

stitchedImage43 = stitchedImage43[:, :1010]
plt.figure(5)
plt.imshow(stitchedImage43, cmap='gray')
plt.show()
```

Assignment 3
Tudorache Vlad Adrian 442C

```
kp43, descriptors43 = orbObj.detectAndCompute(stitchedImage43, None)

matches432 = matcher.match(descriptors2, descriptors43, None)
matches432 = sorted(matches432, key=lambda x: x.distance, reverse=False)  #
Sorting Matches based on distance

GOOD_MATCH_PERCENT = 0.15
numGoodMatches432 = int(len(matches432) * GOOD_MATCH_PERCENT)  # Choosing
only the first 15%
matches432 = matches432[:numGoodMatches432]


imgMatches432 = cv2.drawMatches(img2, kp2, stitchedImage43, kp43, matches432,
None)

plt.figure(6)
plt.imshow(imgMatches432, cmap='gray')
plt.show()

points432 = np.zeros((len(matches432), 2), dtype=np.float32)
points4322 = np.zeros((len(matches432), 2), dtype=np.float32)

for j, match in enumerate(matches432):
    points432[j, :] = kp2[match.queryIdx].pt
    points4322[j, :] = kp43[match.trainIdx].pt

h2, mask2 = cv2.findHomography(points4322, points432, cv2.RANSAC)


im43Height, im43Width = stitchedImage43.shape
im2Height, im2Width = img2.shape

im43Aligned = cv2.warpPerspective(stitchedImage43, h2, (im2Width + im43Width,
im2Height))

plt.figure(7)
plt.imshow(im43Aligned, cmap='gray')
plt.show()

# Stitch Image 43 with aligned image 2
stitchedImage432 = np.copy(im43Aligned)
stitchedImage432[0:im2Height, 0:im2Width] = img2

plt.figure(8)
plt.imshow(stitchedImage432, cmap='gray')
plt.show()

#432 si 1-----------------------------------------------------------------

stitchedImage432 = stitchedImage432[:, :1650]
plt.figure(9)
plt.imshow(stitchedImage432, cmap='gray')
plt.show()

kp432, descriptors432 = orbObj.detectAndCompute(stitchedImage432, None)
```

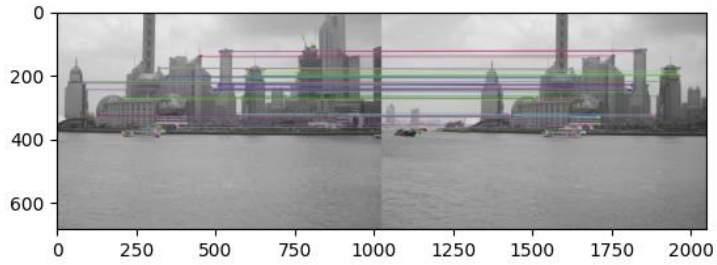
Assignment 3
Tudorache Vlad Adrian 442C

```
matches4321 = matcher.match(descriptors1, descriptors432, None)
matches4321 = sorted(matches4321, key=lambda x: x.distance, reverse=False)  #
Sorting Matches based on distance

GOOD_MATCH_PERCENT = 0.15
numGoodMatches4321 = int(len(matches4321) * GOOD_MATCH_PERCENT)  # Choosing
only the first 15%
matches4321 = matches4321[:numGoodMatches4321]


imgMatches4321 = cv2.drawMatches(img1, kp1, stitchedImage432, kp432,
matches4321, None)

plt.figure(10)
plt.imshow(imgMatches4321, cmap='gray')
plt.show()

points4321 = np.zeros((len(matches4321), 2), dtype=np.float32)
points43212 = np.zeros((len(matches4321), 2), dtype=np.float32)

for j, match in enumerate(matches4321):
    points4321[j, :] = kp1[match.queryIdx].pt
    points43212[j, :] = kp432[match.trainIdx].pt

h3, mask3 = cv2.findHomography(points43212, points4321, cv2.RANSAC)


im432Height, im432Width = stitchedImage432.shape
im1Height, im1Width = img1.shape

im432Aligned = cv2.warpPerspective(stitchedImage432, h3, (im1Width +
im432Width, im1Height))

plt.figure(11)
plt.imshow(im432Aligned, cmap='gray')
plt.show()

# Stitch Image 432 with aligned image 1
stitchedImage4321 = np.copy(im432Aligned)
stitchedImage4321[0:im1Height, 0:im1Width] = img1

plt.figure(12)
plt.imshow(stitchedImage4321, cmap='gray')
plt.show()

stitchedImage4321_cropped = stitchedImage4321[0:670, 0:2100]

plt.figure(13)
plt.imshow(stitchedImage4321_cropped, cmap='gray')
plt.show()
```

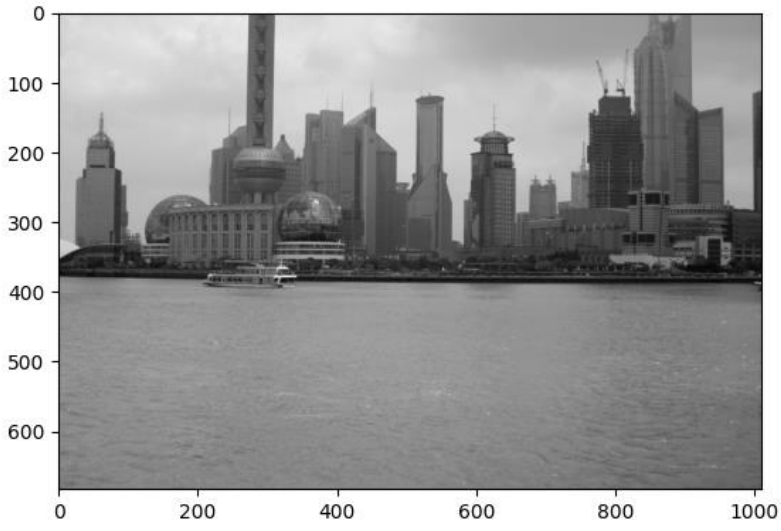
Assignment 3
Tudorache Vlad Adrian 442C

I got better results when I started from the Right part of the panorama. After every stich I did a crop to remove the black borders. Also in the script I get more plots (like the modified perspective plot for a image) so I can adjust the parameters. I did not attach all the plots in this file.

The first matches:

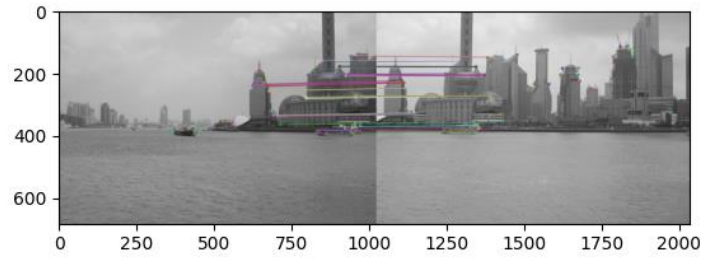


The first stich:

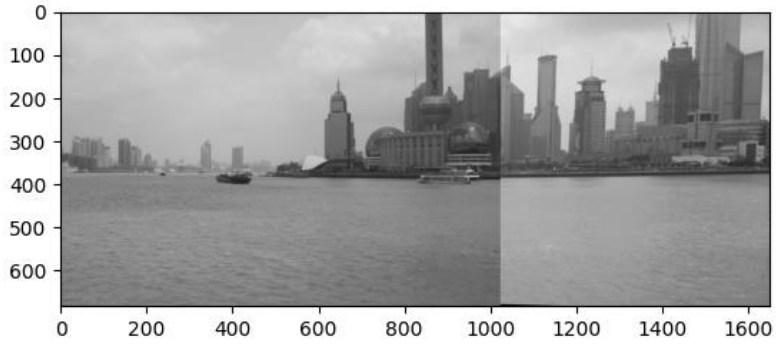


Assignment 3
Tudorache Vlad Adrian 442C

The second matches:

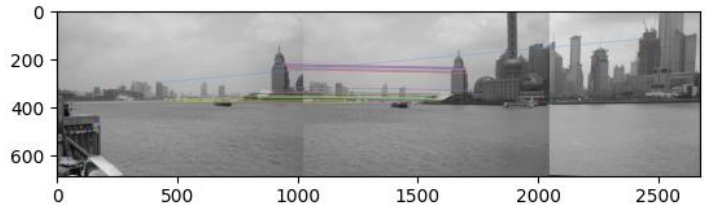


The second stich:

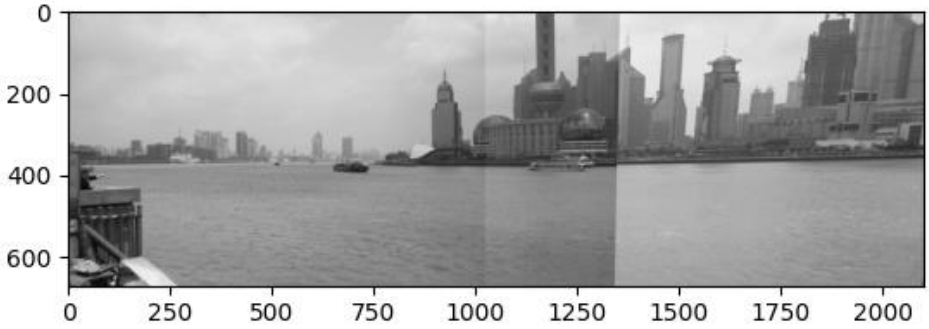


Assignment 3
Tudorache Vlad Adrian 442C

The final matches:



The final stich:



Assignment 3
Tudorache Vlad Adrian 442C

Subsection 2:

For the first section my assigned set_x was set01.

```
# SUBSECTION 2 ------------------------------------------------------------
--------

Obj =
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_y\set01\object.jpg", 1)
Cl1=
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_y\set01\clutter01.jpg",
1)
Cl2=
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_y\set01\clutter02.jpg",
1)
Cl3=
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_y\set01\clutter03.jpg",
1)
Cl4=
cv2.imread(r"C:\Users\Vladuts\Desktop\IPIVA\TEMA3\set_y\set01\clutter04.jpg",
1)

plt.figure(14)
plt.subplot(151);plt.imshow(Obj[:, :, ::-1]);plt.title("Object")
plt.subplot(152);plt.imshow(Cl1[:, :, ::-1]);plt.title("Clutter 1")
plt.subplot(153);plt.imshow(Cl2[:, :, ::-1]);plt.title("Clutter 2")
plt.subplot(154);plt.imshow(Cl3[:, :, ::-1]);plt.title("Clutter 3")
plt.subplot(155);plt.imshow(Cl4[:, :, ::-1]);plt.title("Clutter 4")
plt.show()


def SIFT_MATCH(imgCl, obj):

    siftObject = cv2.SIFT_create()
    ks, des = siftObject.detectAndCompute(obj, None)
    ks1, des1 = siftObject.detectAndCompute(imgCl, None)

    FLANN_INDEX_KDTREE = 1
    index_params = dict(algorithm=FLANN_INDEX_KDTREE, trees=5)
    search_params = dict(checks=100)
    flann = cv2.FlannBasedMatcher(index_params, search_params)
    knn_matches = flann.knnMatch(des1, des, k=2)
    # Filter matches using the Lowe's ratio test
    ratio_thresh = 0.73
    good_matches = []
    for m, n in knn_matches:
        if m.distance < ratio_thresh * n.distance:
            good_matches.append(m)

        # Draw matches
    img_matches = np.empty((max(imgCl.shape[0], obj.shape[0]), imgCl.shape[1]
+ obj.shape[1], 3), dtype=np.uint8)
    cv2.drawMatches(imgCl, ks1, obj, ks, good_matches, img_matches,
flags=cv2.DrawMatchesFlags_NOT_DRAW_SINGLE_POINTS)

    plt.figure()
```

Assignment 3
Tudorache Vlad Adrian 442C

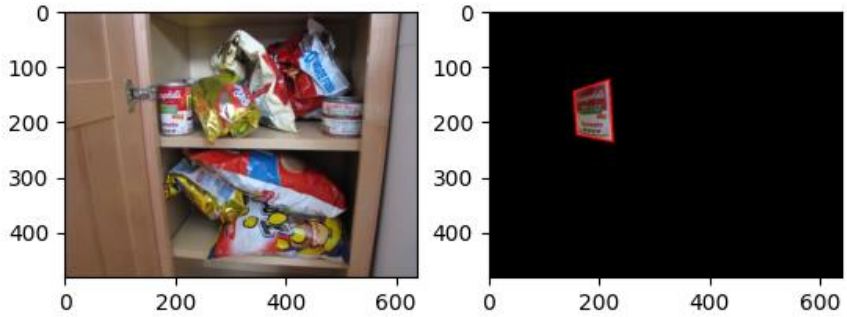
```
    plt.imshow(img_matches[:, :, ::-1])
    plt.show()

        # Compute the homography using findHomography and RANSAC.
    points1 = np.zeros((len(good_matches), 2), dtype=np.float32)
    points2 = np.zeros((len(good_matches), 2), dtype=np.float32)
    for i, match in enumerate(good_matches):
        points2[i, :] = ks1[match.queryIdx].pt
        points1[i, :] = ks[match.trainIdx].pt
        # Find homography
    h, mask = cv2.findHomography(points1, points2, cv2.RANSAC)
    size = (imgCl.shape[1], imgCl.shape[0])
    im_dst = cv2.warpPerspective(obj, h, size)

    img_gray = cv2.cvtColor(im_dst, cv2.COLOR_BGR2GRAY)
    ret, thresh = cv2.threshold(img_gray, 10, 255, cv2.THRESH_BINARY)
    contours, hierarchy = cv2.findContours(image=thresh, mode=cv2.RETR_TREE,
method=cv2.CHAIN_APPROX_NONE)
    image_copy = im_dst.copy()
    cv2.drawContours(image=imgCl, contours=contours, contourIdx=-1, color=(0,
0, 255), thickness=2,
                     lineType=cv2.LINE_AA)
    plt.figure()
    plt.subplot(121);
    plt.imshow(imgCl[:, :, ::-1]);
    plt.subplot(122);
    plt.imshow(image_copy[:, :, ::-1]);
    plt.show()

    plt.figure()
    plt.imshow(imgCl[:, :, ::-1])
    plt.show()

SIFT_MATCH(Cl1, Obj)
SIFT_MATCH(Cl2, Obj)
SIFT_MATCH(Cl3, Obj)
SIFT_MATCH(Cl4, Obj)
```

I have created a function, so I do not have to repeat the same script for every picture. I have tried the ORB as well, but I did not get good results.

Assignment 3
Tudorache Vlad Adrian 442C

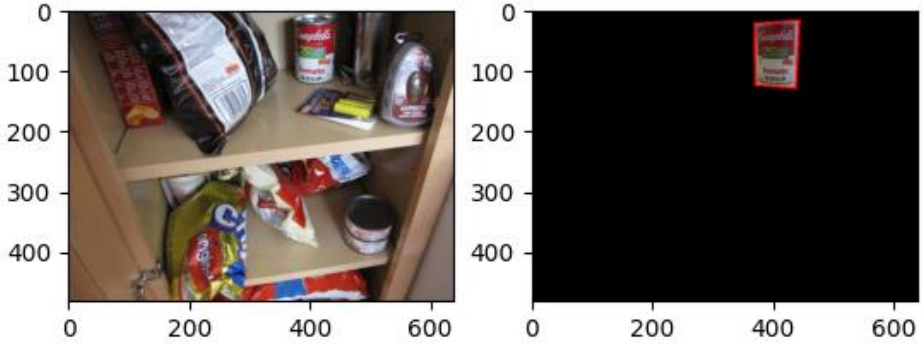
First Clutter:

Assignment 3
Tudorache Vlad Adrian 442C



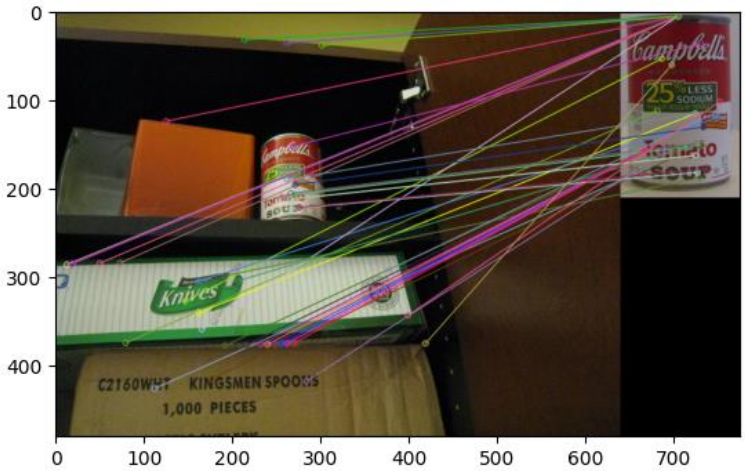
Assignment 3
Tudorache Vlad Adrian 442C

Second Clutter:

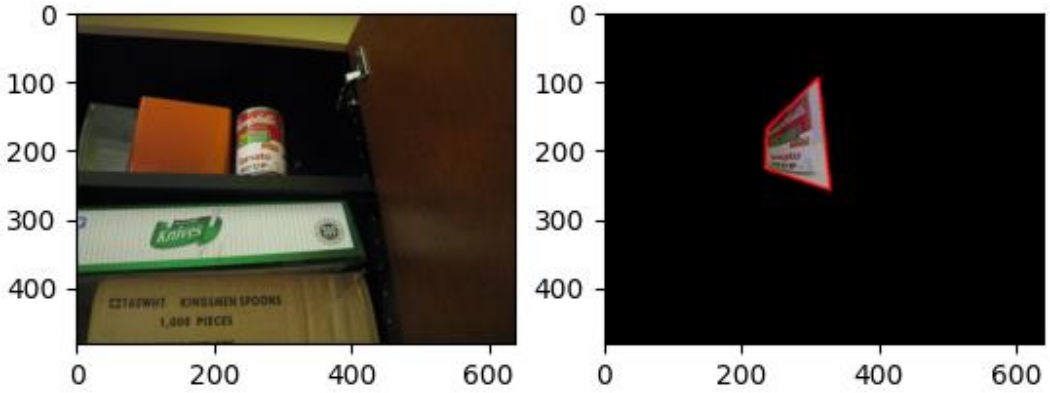


Assignment 3
Tudorache Vlad Adrian 442C

Third Clutter:

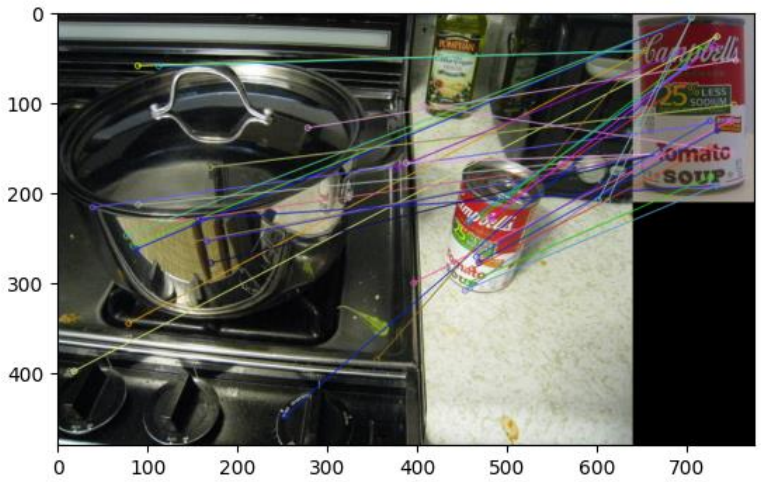


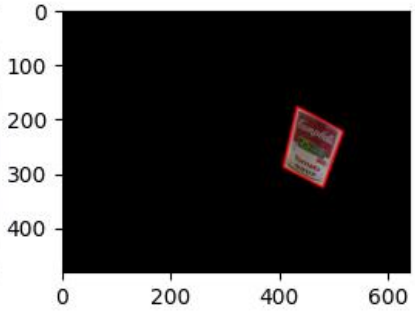
Assignment 3
Tudorache Vlad Adrian 442C



Assignment 3
Tudorache Vlad Adrian 442C

Last Clutter





Assignment 3
Tudorache Vlad Adrian 442C