In [1]:

```
!pip install torchsummary
```

```
Requirement already satisfied: torchsummary in /opt/conda/lib/python3.7/site-packages (1.
5.1)
```

In [2]:

```
import numpy as np

import scipy.io
import os
from numpy.linalg import norm,det,inv,svd
from scipy.linalg import rq
import math
import matplotlib.pyplot as plt
import numpy as np
import math
import random
import sys
from scipy import ndimage,spatial
from tqdm.notebook import trange,tqdm
import torch
import torch.nn as nn
import torch.optim as optim
from torch.optim import lr_scheduler
from torch.autograd import Variable
import torchvision
from torchvision import datasets,models,transforms
from  torch.utils.data import Dataset,DataLoader,ConcatDataset
from skimage import io,transform,data
from torchvision import transforms,utils
import  os
import sklearn.svm
import cv2
from os.path import exists
import pandas as pd
import PIL
from sklearn.metrics.cluster import completeness_score
from sklearn.cluster import KMeans
from tqdm import tqdm,tqdm_notebook
from functools import partial
from  torchsummary import summary
from torchvision.datasets import ImageFolder
from torch.utils.data.sampler import SubsetRandomSampler
```

In [3]:

```
class Image:
    def __init__(self,img,position):
        self.img = img
        self.position  = position

inliner_matchset = []
def features_matching(a,keypointlength,threshold):
    bestmatch = np.empty((keypointlength), dtype=np.int16)
    img1index = np.empty((keypointlength),dtype=np.init16)
    distance = np.empty((keypointlength))
    index =0
    for j in range(0,keypointlength):
        x=a[j]
        listx = x.tolist()
        x.sort()
        minval1=x[0]
        minval2=x[1]
        itemindex1 = listx.index(minval1)
        itemindex2 = listx.index(minval2)
```

```
        ratio = minval1/minval2

        if ratio < threshold:
            bestmatch[index] = itemindex1
            distance[index] = minval1
            img1index[index] = j
            index = index + 1
    return [cv2.DMatch(img1index[i],bestmatch[i].astype(int),distance[i]) for i in range
(0,index)]

def compute_Hmography(im1_pts,im2_pts):
    num_matches=len(im1_pts)
    num_rows = 2*num_matches
    num_cols = 9
    A_matrix_shape = (num_rows,num_cols)
    A = np.zeros(A_matrix_shape)
    a_index = 0
    for i in range(0,num_matches):
        (a_x,a_y) = im1_pts[i]
        (b_x,b_y) = im2_pts[i]
        row1 = [a_x,a_y,1,0,0,0,-b_x*a_x,-b_x*a_y,-b_x]
        row2 = [0,0,0,a_x,a_y,1,-b_y*a_x,-b_y*a_y,-b_y]
        A[a_index] = row1

        A[a_index+1] = row2
        a_index += 2

    U,s,Vt = np.linalg.svd(A)
    H = np.eye(3)
    H = Vt[-1].reshape(3,3)
    return H

def displayplot(img,title):
    plt.figure(figsize=(15,15))
    plt.title(title)
    plt.imshow(cv2.cvtColor(img,cv2.COLOR_BGR2RGB))
    plt.show()

def RANSAC_alg(f1,f2,matches,nRANSAC,RANSACthresh):
    minMatches = 4
    nBest = 0
    best_inliners = []
    H_estimate = np.eye(3,3)
    global inliner_matchset
    inliner_matchset = []
    for iteration in range(nRANSAC):
        matchSimple = random.sample(matches,minMatches)
        im1_pts = np.empty((minMatches,2))
        im2_pts = np.empty((minMatches,2))
        for i in range(0,minMatches):
            m  = matchSimple[i]
            im1_pts[i] = f1[m.queryIdx].pt
            im2_pts[i] = f2[m.trainIdx].pt

        H_estimate = compute_Hmography(im1_pts,im2_pts)
        inliners = get_inliners(f1,f2,matches,H_estimate,RANSACthresh)
        if len(inliners) > nBest:
            nBest = len(inliners)
            best_inliners= inliners

    print("Number of best inliners", len(best_inliners))
    for i in range(len(best_inliners)):
        inliner_matchset.append(matches[best_inliners[i]])
    im1_pts = np.empty((len(best_inliners),2))
    im2_pts = np.empty((len(best_inliners),2))
    for i in range(0,len(best_inliners)):
        m = inliner_matchset[i]
        im1_pts[i] = f1[m.queryIdx].pt
        im2_pts[i] = f2[m.trainIdx].pt
    M = compute_Hmography(im1_pts,im2_pts)
    return M, len(best_inliners)
```

In [4]:

```
!pip install opencv-python==3.4.2.17
!pip install opencv-contrib-python==3.4.2.17
```

```
Requirement already satisfied: opencv-python==3.4.2.17 in /opt/conda/lib/python3.7/site-p
ackages (3.4.2.17)
Requirement already satisfied: numpy>=1.14.5 in /opt/conda/lib/python3.7/site-packages (f
rom opencv-python==3.4.2.17) (1.19.5)
Requirement already satisfied: opencv-contrib-python==3.4.2.17 in /opt/conda/lib/python3.
7/site-packages (3.4.2.17)
Requirement already satisfied: numpy>=1.14.5 in /opt/conda/lib/python3.7/site-packages (f
rom opencv-contrib-python==3.4.2.17) (1.19.5)
```

In [5]:

```
import cv2
```

In [6]:

```
files_all = os.listdir('../input/uni-campus-dataset/RGB-img/img/')
files_all.sort()

folder_path = '../input/uni-campus-dataset/RGB-img/img/'
left_files_path_rev = []
right_files_path = []
for file in files_all[:61]:
    left_files_path_rev.append(folder_path + file)


left_files_path = left_files_path_rev[::-1]


for file in files_all[61:100]:
    right_files_path.append(folder_path + file)
```

In [7]:

```
gridsize = 8
clahe = cv2.createCLAHE(clipLimit=2.0,tileGridSize=(gridsize,gridsize))
images_left_bgr = []
images_right_bgr = []
images_left = []
images_right = []


for file in tqdm(left_files_path):
    left_image_sat= cv2.imread(file)
    lab = cv2.cvtColor(left_image_sat, cv2.COLOR_BGR2LAB)
    lab[...,0] = clahe.apply(lab[...,0])
    left_image_sat = cv2.cvtColor(lab, cv2.COLOR_LAB2BGR)
    left_img = cv2.resize(left_image_sat,None,fx=0.35, fy=0.35, interpolation = cv2.INTE
R_CUBIC)
    images_left.append(cv2.cvtColor(left_img, cv2.COLOR_BGR2GRAY).astype('float32')/255.)
    images_left_bgr.append(left_img)


for file in tqdm(right_files_path):
    right_image_sat= cv2.imread(file)
    lab = cv2.cvtColor(right_image_sat, cv2.COLOR_BGR2LAB)
    lab[...,0] = clahe.apply(lab[...,0])
    right_image_sat = cv2.cvtColor(lab, cv2.COLOR_LAB2BGR)
    right_img = cv2.resize(right_image_sat,None,fx=0.35,fy=0.35, interpolation = cv2.INT
ER_CUBIC)
    images_right.append(cv2.cvtColor(right_img, cv2.COLOR_BGR2GRAY).astype('float32')/255
.)
    images_right_bgr.append(right_img)
```

```
100%|██████████| 61/61 [01:06<00:00,  1.09s/it]
100%|██████████| 39/39 [00:41<00:00,  1.07s/it]
```

In [8]:

```
images_left_bgr_no_enhance = []
images_right_bgr_no_enhance = []

for file in tqdm(left_files_path):
    left_image_sat= cv2.imread(file)
    left_img = cv2.resize(left_image_sat,None,fx=0.35, fy=0.35, interpolation = cv2.INTE
R_CUBIC)
    images_left_bgr_no_enhance.append(left_img)

for file in tqdm(right_files_path):
    right_image_sat= cv2.imread(file)
    right_img = cv2.resize(right_image_sat,None,fx=0.35,fy=0.35, interpolation = cv2.INT
ER_CUBIC)
    images_right_bgr_no_enhance.append(right_img)
```

```
100%|██████████| 61/61 [00:24<00:00,  2.46it/s]
100%|██████████| 39/39 [00:15<00:00,  2.50it/s]
```

In [9]:

```
Threshl=60;
Octaves=8;
#PatternScales=1.0f;
brisk = cv2.BRISK_create(Threshl,Octaves)

keypoints_all_left_brisk = []
descriptors_all_left_brisk = []
points_all_left_brisk=[]


keypoints_all_right_brisk = []
descriptors_all_right_brisk = []
points_all_right_brisk=[]


for imgs in tqdm(images_left_bgr):
    kpt = brisk.detect(imgs,None)
    kpt,descrip = brisk.compute(imgs, kpt)
    keypoints_all_left_brisk.append(kpt)
    descriptors_all_left_brisk.append(descrip)
    points_all_left_brisk.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr):
    kpt = brisk.detect(imgs,None)
    kpt,descrip = brisk.compute(imgs, kpt)
    keypoints_all_right_brisk.append(kpt)
    descriptors_all_right_brisk.append(descrip)
    points_all_right_brisk.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

```
100%|██████████| 61/61 [00:56<00:00,  1.08it/s]
100%|██████████| 39/39 [00:35<00:00,  1.09it/s]
```

In [29]:

```
orb = cv2.ORB_create(5000)
keypoints_all_left_orb = []
descriptors_all_left_orb = []
points_all_left_orb=[]

keypoints_all_right_orb = []
descriptors_all_right_orb = []
points_all_right_orb=[]

for imgs in tqdm(images_left_bgr):
    kpt = orb.detect(imgs,None)
    kpt,descrip = orb.compute(imgs, kpt)
    keypoints_all_left_orb.append(kpt)
    descriptors_all_left_orb.append(descrip)
    points_all_left_orb.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

```
for imgs in tqdm(images_right_bgr):
    kpt = orb.detect(imgs,None)
    kpt,descrip = orb.compute(imgs, kpt)
    keypoints_all_right_orb.append(kpt)
    descriptors_all_right_orb.append(descrip)
    points_all_right_orb.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

```
100%|██████████| 61/61 [00:09<00:00,  6.37it/s]
100%|██████████| 39/39 [00:05<00:00,  6.70it/s]
```

In [9]:

```
kaze = cv2.KAZE_create()
keypoints_all_left_kaze = []
descriptors_all_left_kaze = []
points_all_left_kaze=[]

keypoints_all_right_kaze = []
descriptors_all_right_kaze = []
points_all_right_kaze=[]

for imgs in tqdm(images_left_bgr):
    kpt = kaze.detect(imgs,None)
    kpt,descrip = kaze.compute(imgs, kpt)
    keypoints_all_left_kaze.append(kpt)
    descriptors_all_left_kaze.append(descrip)
    points_all_left_kaze.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr):
    kpt = kaze.detect(imgs,None)
    kpt,descrip = kaze.compute(imgs, kpt)
    keypoints_all_right_kaze.append(kpt)
    descriptors_all_right_kaze.append(descrip)
    points_all_right_kaze.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

```
100%|██████████| 61/61 [07:28<00:00,  7.34s/it]
100%|██████████| 39/39 [04:46<00:00,  7.33s/it]
```

In [9]:

```
tqdm = partial(tqdm, position=0, leave=True)
```

In [10]:

```
akaze = cv2.AKAZE_create()
keypoints_all_left_akaze = []
descriptors_all_left_akaze = []
points_all_left_akaze=[]

keypoints_all_right_akaze = []
descriptors_all_right_akaze = []
points_all_right_akaze=[]

for imgs in tqdm(images_left_bgr):
    kpt = akaze.detect(imgs,None)
    kpt,descrip = akaze.compute(imgs, kpt)
    keypoints_all_left_akaze.append(kpt)
    descriptors_all_left_akaze.append(descrip)
    points_all_left_akaze.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
for imgs in tqdm(images_right_bgr):
    kpt = akaze.detect(imgs,None)
    kpt,descrip = akaze.compute(imgs, kpt)
    keypoints_all_right_akaze.append(kpt)
    descriptors_all_right_akaze.append(descrip)
    points_all_right_akaze.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

```
100%|██████████| 61/61 [01:25<00:00,  1.40s/it]
100%|██████████| 39/39 [00:51<00:00,  1.33s/it]
```

In [ ]:

```
star = cv2.xfeatures2d.StarDetector_create()
```

```
brief = cv2.xfeatures2d.BriefDescriptorExtractor_create()
keypoints_all_left_star = []
descriptors_all_left_brief = []
points_all_left_star=[]


keypoints_all_right_star = []
descriptors_all_right_brief = []
points_all_right_star=[]

for imgs in tqdm(images_left_bgr):
    kpt = star.detect(imgs,None)
    kpt,descrip = brief.compute(imgs, kpt)
    keypoints_all_left_star.append(kpt)
    descriptors_all_left_brief.append(descrip)
    points_all_left_star.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr):
    kpt = star.detect(imgs,None)
    kpt,descrip = brief.compute(imgs, kpt)
    keypoints_all_right_star.append(kpt)
    descriptors_all_right_brief.append(descrip)
    points_all_right_star.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [32]:

```
Threshl=60;
Octaves=8;
#PatternScales=1.0f;
brisk = cv2.BRISK_create(Threshl,Octaves)
freak = cv2.xfeatures2d.FREAK_create()
keypoints_all_left_freak = []
descriptors_all_left_freak = []
points_all_left_freak=[]


keypoints_all_right_freak = []
descriptors_all_right_freak = []
points_all_right_freak=[]

for imgs in tqdm(images_left_bgr):
    kpt = brisk.detect(imgs)
    kpt,descrip = freak.compute(imgs, kpt)
    keypoints_all_left_freak.append(kpt)
    descriptors_all_left_freak.append(descrip)
    points_all_left_freak.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr):
    kpt = brisk.detect(imgs,None)
    kpt,descrip = freak.compute(imgs, kpt)
    keypoints_all_right_freak.append(kpt)
    descriptors_all_right_freak.append(descrip)
    points_all_right_freak.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

```
100%|██████████| 61/61 [00:49<00:00,  1.23it/s]
100%|██████████| 39/39 [00:30<00:00,  1.26it/s]
```

In [ ]:

```
mser = cv2.MSER_create()
sift = cv2.xfeatures2d.SIFT_create()
keypoints_all_left_mser = []
descriptors_all_left_mser = []
points_all_left_mser=[]

keypoints_all_right_mser = []
descriptors_all_right_mser = []
points_all_right_mser=[]
for imgs in tqdm(images_left_bgr_no_enhance):
    kpt = mser.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
```

```
    keypoints_all_left_mser.append(kpt)
    descriptors_all_left_mser.append(descrip)
    points_all_left_mser.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))


for imgs in tqdm(images_right_bgr_no_enhance):
    kpt = mser.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_right_mser.append(kpt)
    descriptors_all_right_mser.append(descrip)
    points_all_right_mser.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [ ]:

```
agast = cv2.AgastFeatureDetector_create()
sift = cv2.xfeatures2d.SIFT_create()
keypoints_all_left_agast = []
descriptors_all_left_agast = []
points_all_left_agast=[]

keypoints_all_right_agast = []
descriptors_all_right_agast = []
points_all_right_agast=[]


for imgs in tqdm(images_left_bgr_no_enhance):
    kpt = agast.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_left_agast.append(kpt)
    descriptors_all_left_agast.append(descrip)
    points_all_left_agast.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))


for imgs in tqdm(images_right_bgr_no_enhance):
    kpt = agast.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_right_agast.append(kpt)
    descriptors_all_right_agast.append(descrip)
    points_all_right_agast.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [ ]:

```
fast = cv2.FastFeatureDetector_create()
sift = cv2.xfeatures2d.SIFT_create()
keypoints_all_left_fast = []
descriptors_all_left_fast = []
points_all_left_fast=[]

keypoints_all_right_fast = []
descriptors_all_right_fast = []
points_all_right_fast=[]
for imgs in tqdm(images_left_bgr_no_enhance):
    kpt = fast.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_left_fast.append(kpt)
    descriptors_all_left_fast.append(descrip)
    points_all_left_fast.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr_no_enhance):
    kpt = fast.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_right_fast.append(kpt)
    descriptors_all_right_fast.append(descrip)
    points_all_right_fast.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [ ]:

```
gftt = cv2.GFTTDetector_create()
sift = cv2.xfeatures2d.SIFT_create()
keypoints_all_left_gftt = []
descriptors_all_left_gftt = []
```

```
points_all_left_gftt=[]

keypoints_all_right_gftt = []
descriptors_all_right_gftt = []
points_all_right_gftt=[]
for imgs in tqdm(images_left_bgr_no_enhance):
    kpt = gftt.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_left_gftt.append(kpt)
    descriptors_all_left_gftt.append(descrip)
    points_all_left_gftt.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr_no_enhance):
    kpt = gftt.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_right_gftt.append(kpt)
    descriptors_all_right_gftt.append(descrip)
    points_all_right_gftt.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [ ]:

```
daisy = cv2.xfeatures2d.DAISY_create()
sift = cv2.xfeatures2d.SIFT_create()
keypoints_all_left_daisy = []
descriptors_all_left_daisy = []
points_all_left_daisy=[]

keypoints_all_right_daisy = []
descriptors_all_right_daisy = []
points_all_right_daisy=[]

for imgs in tqdm(images_left_bgr_no_enhance):
    kpt = sift.detect(imgs,None)
    kpt,descrip = daisy.compute(imgs, kpt)
    keypoints_all_left_daisy.append(kpt)
    descriptors_all_left_daisy.append(descrip)
    points_all_left_daisy.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr_no_enhance):
    kpt = sift.detect(imgs,None)
    kpt,descrip = daisy.compute(imgs, kpt)
    keypoints_all_right_daisy.append(kpt)
    descriptors_all_right_daisy.append(descrip)
    points_all_right_daisy.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [ ]:

```
surf = cv2.xfeatures2d.SURF_create()
sift = cv2.xfeatures2d.SIFT_create()
keypoints_all_left_surfsift = []
descriptors_all_left_surfsift = []
points_all_left_surfsift=[]

keypoints_all_right_surfsift = []
descriptors_all_right_surfsift = []
points_all_right_surfsift=[]

for imgs in tqdm(images_left_bgr_no_enhance):
    kpt = surf.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_left_surfsift.append(kpt)
    descriptors_all_left_surfsift.append(descrip)
    points_all_left_surfsift.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr_no_enhance):
    kpt = surf.detect(imgs,None)

    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_right_surfsift.append(kpt)
    descriptors_all_right_surfsift.append(descrip)
    points_all_right_surfsift.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

```
    `
```

In [ ]:

```
sift = cv2.xfeatures2d.SIFT_create()
keypoints_all_left_sift = []
descriptors_all_left_sift = []
points_all_left_sift=[]

keypoints_all_right_sift = []
descriptors_all_right_sift = []
points_all_right_sift=[]

for imgs in tqdm(images_left_bgr_no_enhance):
    kpt = sift.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_left_sift.append(kpt)
    descriptors_all_left_sift.append(descrip)
    points_all_left_sift.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr_no_enhance):
    kpt = sift.detect(imgs,None)
    kpt,descrip = sift.compute(imgs, kpt)
    keypoints_all_right_sift.append(kpt)
    descriptors_all_right_sift.append(descrip)
    points_all_right_sift.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [ ]:

```
surf = cv2.xfeatures2d.SURF_create()
keypoints_all_left_surf = []
descriptors_all_left_surf = []
points_all_left_surf=[]

keypoints_all_right_surf = []
descriptors_all_right_surf = []
points_all_right_surf=[]
for imgs in tqdm(images_left_bgr):
    kpt = surf.detect(imgs,None)
    kpt,descrip = surf.compute(imgs, kpt)
    keypoints_all_left_surf.append(kpt)
    descriptors_all_left_surf.append(descrip)
    points_all_left_surf.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))

for imgs in tqdm(images_right_bgr):
    kpt = surf.detect(imgs,None)
    kpt,descrip = surf.compute(imgs, kpt)
    keypoints_all_right_surf.append(kpt)
    descriptors_all_right_surf.append(descrip)
    points_all_right_surf.append(np.asarray([[p.pt[0],p.pt[1]] for p in kpt]))
```

In [ ]:

```
# sift  = cv2.xfeatures2d.SURF_Create()
# keypoints_all_left_surf = []
# descriptor_all_left_surf = []
# points_all_left_surf = []

# keypoints_all_right_surf = []
# descriptor_all_right_surf = []
# points_all_right_surf = []

#  for images in tqdm(left_images_bgr):
#  kpt = surf.detect(imgs,None)
# kpt, descrip = surf.compute(imgs,kpt)
# keypoints_all_left_surf.append(kpt)
#descriptor_all_left_surf.append(descrip)
#points_all_left_surf.append(np.asarray([[p.pt[0],p.pt[1]] for p in kpt])
#  points_all_left_surf.append(np.asarray([[p.pt[0], p.pt[1] for p in  kpt]]))
```

In [ ]:

```
class RootSIFT:
    def __init__(self):
        # initialize the SIFT feature extractor
        #self.extractor = cv2.DescriptorExtractor_create("SIFT")
        self.sift = cv2.xfeatures2d.SIFT_create()
    def compute(self, image, kps, eps=1e-7):
        # compute SIFT descriptors
        (kps, descs) = self.sift.compute(image, kps)
        # if there are no keypoints or descriptors, return an empty tuple
        if len(kps) == 0:
            return ([], None)
        # apply the Hellinger kernel by first L1-normalizing, taking the
        # square-root, and then L2-normalizing
        descs /= (np.linalg.norm(descs, axis=0, ord=2) + eps)
        descs /= (descs.sum(axis=0) + eps)
        descs = np.sqrt(descs)
        #descs /= (np.linalg.norm(descs, axis=0, ord=2) + eps)
        # return a tuple of the keypoints and descriptors
        return (kps, descs)
```

In [ ]:

```
sift = cv2.xfeatures2d.SIFT_create()
rootsift = RootSIFT()
keypoints_all_left_rootsift = []
descriptors_all_left_rootsift = []
points_all_left_rootsift=[]

keypoints_all_right_rootsift = []
descriptors_all_right_rootsift = []
points_all_right_rootsift=[]

for imgs in tqdm(images_left_bgr):
    kpt = sift.detect(imgs,None)
    kpt,descrip = rootsift.compute(imgs, kpt)
    keypoints_all_left_rootsift.append(kpt)
    descriptors_all_left_rootsift.append(descrip)
    points_all_left_rootsift.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
for imgs in tqdm(images_right_bgr):
    kpt = sift.detect(imgs,None)
    kpt,descrip = rootsift.compute(imgs, kpt)
    keypoints_all_right_rootsift.append(kpt)
    descriptors_all_right_rootsift.append(descrip)
    points_all_right_rootsift.append(np.asarray([[p.pt[0], p.pt[1]] for p in kpt]))
```

In [11]:

```
!git clone https://github.com/magicleap/SuperPointPretrainedNetwork.git
```

```
fatal: destination path 'SuperPointPretrainedNetwork' already exists and is not an empty
directory.
```

In [12]:

```
weights_path = 'SuperPointPretrainedNetwork/superpoint_v1.pth'
cuda = 'True'
```

In [14]:

```
def to_kpts(pts,size=1):
    return [cv2.KeyPoint(pt[0],pt[1],size) for pt in pts]
```

In [15]:

```
torch.cuda.empty_cache()
class SuperPointNet(nn.Module):
    def __init__(self):
        super(SuperPointNet,self).__init__()
        self.relu = nn.ReLU(inplace=True)
        self.pool = nn.MaxPool2d(kernel_size=2, stride=2)
```

```
        c1,c2,c3,c4,c5,d1 = 64,64,128,128,256,256
        self.conv1a = nn.Conv2d(1,c1,kernel_size=3,stride=1,padding=1)
        self.conv1b = nn.Conv2d(c1,c1,kernel_size=3,stride=1,padding=1)
        self.conv2a = nn.Conv2d(c1,c2,kernel_size=3,stride=1,padding=1)
        self.conv2b = nn.Conv2d(c2,c2,kernel_size=3,stride=1,padding=1)
        self.conv3a = nn.Conv2d(c2,c3,kernel_size=3,stride=1,padding=1)
        self.conv3b = nn.Conv2d(c3,c3,kernel_size=3,stride=1,padding=1)
        self.conv4a = nn.Conv2d(c3,c4,kernel_size=3,stride=1,padding=1)
        self.conv4b = nn.Conv2d(c4,c4,kernel_size=3,stride=1,padding=1)
        self.convPa = nn.Conv2d(c4,c5,kernel_size=3,stride=1,padding=1)
        self.convPb = nn.Conv2d(c5,65,kernel_size=1,stride=1,padding=0)
        self.convDa = nn.Conv2d(c4,c5,kernel_size=3,stride=1,padding=1)


        self.convDb = nn.Conv2d(c5,d1,kernel_size=1,stride=1,padding=0)

    def forward(self,x):
        x = self.relu(self.conv1a(x))
        x = self.relu(self.conv1b(x))
        x = self.pool(x)
        x = self.relu(self.conv2a(x))
        x = self.relu(self.conv2b(x))
        x = self.pool(x)
        x  = self.relu(self.conv3a(x))
        x = self.relu(self.conv3b(x))
        x = self.pool(x)
        x = self.relu(self.conv4a(x))
        x = self.relu(self.conv4b(x))
        cPa = self.relu(self.convPa(x))
        semi = self.convPb(cPa)
        cDa = self.relu(self.convDa(x))
        desc = self.convDb(cDa)
        dn = torch.norm(desc,p=2,dim=1)
        desc = desc.div(torch.unsqueeze(dn,1))
        return semi,desc


class SuperPointFrontend(object):
    def __init__(self,weights_path,nms_dist,conf_thresh, nn_thresh,cuda=True):
        self.name = 'SuperPoint'
        self.cuda = cuda
        self.nms_dist = nms_dist
        self.conf_thresh = conf_thresh
        self.nn_thresh = nn_thresh
        self.cell = 8
        self.border_remove = 4

        self.net = SuperPointNet()
        if cuda:
            self.net.load_state_dict(torch.load(weights_path))
            self.net = self.net.cuda()
        else:
            self.net.load_state_dict(torch.load(weights_path,map_location=lambda storage
, loc: storage))
        self.net.eval()

    def nms_fast(self,in_corners,H,W,dist_thresh):
        grid = np.zeros((H,W)).astype(int)
        inds = np.zeros((H,W)).astype(int)
        inds1 = np.argsort(-in_corners[2,:])
        corners  = in_corners[:,inds1]
        rcorners = corners[:2,:].round().astype(int)
        if rcorners.shape[1] == 0:
            return np.zeros((3,0)).astype(int), np.zeros(0).astype(int)
        if rcorners.shape[1] == 1:
            out =  np.vstack((rcorners,in_corners[2])).reshape(3,1)
            return out,np.zeros((1)).astype(int)
        for i, rc in enumerate(rcorners.T):
            grid[rcorners[1,i],rcorners[0,i]] =1
            inds[rcorners[1,i],rcorners[0,i]] =i
        pad = dist_thresh
        grid = np.pad(grid,((pad,pad),(pad,pad)),mode='constant')
```

```
        count = 0
        for i,rc  in enumerate(rcorners.T):
            pt = (rc[0]+pad, rc[1]+pad)
            if grid[pt[1], pt[0]] == 1:
                grid[pt[1]-pad:pt[1]+pad+1, pt[0]-pad:pt[0]+pad+1]=0


                grid[pt[1], pt[0]] = -1
                count += 1

        keepy, keepx = np.where(grid==-1)
        keepy,keepx = keepy-pad , keepx-pad
        inds_keep = inds[keepy, keepx]
        out  = corners[:,inds_keep]
        values = out[-1,:]
        inds2 = np.argsort(-values)
        out = out[:,inds2]
        out_inds = inds1[inds_keep[inds2]]
        return out, out_inds

    def run(self,img):
        assert img.ndim == 2
        assert img.dtype == np.float32
        H,W = img.shape[0], img.shape[1]
        inp = img.copy()
        inp = (inp.reshape(1,H,W))
        inp = torch.from_numpy(inp)
        inp = torch.autograd.Variable(inp).view(1,1,H,W)
        if self.cuda:
            inp = inp.cuda()
        outs = self.net.forward(inp)
        semi,coarse_desc = outs[0],outs[1]
        semi = semi.data.cpu().numpy().squeeze()


        dense = np.exp(semi)
        dense = dense / (np.sum(dense,axis=0)+.00001)
        nodust = dense[:-1,:,:]
        Hc = int(H / self.cell)
        Wc = int(W  / self.cell)
        nodust = np.transpose(nodust,[1,2,0])
        heatmap = np.reshape(nodust,[Hc,Wc,self.cell,self.cell])
        heatmap = np.transpose(heatmap,[0,2,1,3])
        heatmap = np.reshape(heatmap,[Hc*self.cell, Wc*self.cell])
        prob_map  = heatmap/np.sum(np.sum(heatmap))

        return heatmap,coarse_desc

    def key_pt_sampling(self,img,heat_map,coarse_desc,sampled):
        H,W = img.shape[0], img.shape[1]
        xs,ys = np.where(heat_map >= self.conf_thresh)
        if len(xs) == 0:
            return np.zeros((3,0)),None,None
        print("Number of pts selected:",len(xs))

        pts = np.zeros((3,len(xs)))
        pts[0,:] = ys
        pts[1,:] = xs
        pts[2,:] = heat_map[xs,ys]
        pts,_ = self.nms_fast(pts,H,W,dist_thresh=self.nms_dist)
        inds = np.argsort(pts[2,:])
        pts = pts[:,inds[::-1]]
        bord = self.border_remove
        toremoveW = np.logical_or(pts[0,:] < bord, pts[0,:] >= (W-bord))
        toremoveH = np.logical_or(pts[1,:] < bord, pts[0,:] >= (H-bord))
        toremove = np.logical_or(toremoveW, toremoveH)
        pts = pts[:,~toremove]
        pts = pts[:,0:sampled]
        D = coarse_desc.shape[1]
        if pts.shape[1] == 0:
            desc = np.zeros((D,0))
        else:
```

```
            samp_pts = torch.from_numpy(pts[:2,:].copy())
            samp_pts[0,:] = (samp_pts[0,:] / (float(W)/2.))-1.
            samp_pts[1,:] = (samp_pts[1,:] / (float(W)/2.))-1.
            samp_pts = samp_pts.transpose(0,1).contiguous()
            samp_pts = samp_pts.view(1,1,-1,2)
            samp_pts = samp_pts.float()
            if self.cuda:
                samp_pts = samp_pts.cuda()
            desc = nn.functional.grid_sample(coarse_desc, samp_pts)
            desc = desc.data.cpu().numpy().reshape(D,-1)
            desc /= np.linalg.norm(desc,axis=0)[np.newaxis,:]
        return pts,desc
```

In [16]:

```
print('Load pre trained network')
fe = SuperPointFrontend(weights_path = weights_path, nms_dist = 4, conf_thresh  = 0.015,
nn_thresh=0.7,
                        cuda = cuda)
print('Successfully  loaded pretrained network')
```

```
Load pre trained network
Successfully  loaded pretrained network
```

In [ ]:

```
keypoint_all_left_superpoint = []
descriptor_all_left_superpoint = []
point_all_left_superpoint = []

keypoints_all_right_superpoint = []
descriptors_all_right_superpoint = []
points_all_right_superpoint = []

for ifpth in tqdm(images_left):
    heatmap1, coarse_desc1 = fe.run(ifpth)
    pts_1, desc_1 = fe.key_pt_sampling(ifpth,heatmap1,coarse_desc1,2000)

    keypoint_all_left_superpoint.append(to_kpts(pts_1.T))
    descriptor_all_left_superpoint.append(desc_1.T)
    point_all_left_superpoint.append(pts_1.T)





for rfpth in tqdm(images_right):
    heatmap1, coarse_desc1 = fe.run(rfpth)
    pts_1, desc_1 = fe.key_pt_sampling(rfpth,heatmap1,coarse_desc1,2000)

    keypoints_all_right_superpoint.append(to_kpts(pts_1.T))
    descriptors_all_right_superpoint.append(desc_1.T)
    points_all_right_superpoint.append(pts_1.T)
```

In [16]:

```
num_kps_brisk = []
for j in tqdm(keypoints_all_left_brisk + keypoints_all_right_brisk):
    num_kps_brisk.append(len(j))
```

```
100%|██████████| 100/100 [00:00<00:00, 456895.86it/s]
```

In [30]:

```
num_kps_orb = []
for j  in tqdm(keypoints_all_left_orb + keypoints_all_right_orb):
    num_kps_orb.append(len(j))
```

```
100%|██████████| 100/100 [00:00<00:00, 404465.19it/s]
```

In [17]:

```
num_kps_kaze = []
for j  in tqdm(keypoints_all_left_kaze + keypoints_all_right_kaze):
    num_kps_kaze.append(len(j))
```

```
100%|██████████| 100/100 [00:00<00:00, 101630.82it/s]
```

In [18]:

```
num_kps_akaze  = []




for j in tqdm(keypoints_all_left_akaze + keypoints_all_right_akaze):
    num_kps_akaze.append(len(j))
```

```
100%|██████████| 100/100 [00:00<00:00, 415689.20it/s]
```

In [ ]:

```
num_kps_freak = []
for j in tqdm(keypoints_all_left_freak + keypoints_all_right_freak):
    num_kps_freak.append(len(j))
```

In [ ]:

```
def compute_homography_fast(matched_pts1, matched_pts2,thresh=4):
    #matched_pts1 = cv2.KeyPoint_convert(matched_kp1)
    #matched_pts2 = cv2.KeyPoint_convert(matched_kp2)
    # Estimate the homography between the matches using RANSAC
    H, inliers = cv2.findHomography(matched_pts1,matched_pts2,cv2.RANSAC, ransacReprojTh
reshold =thresh)
    inliers = inliers.flatten()
    return H, inliers
```

In [20]:

```
def get_Hmatrix(imgs,keypts,pts,descripts,ratio=0.8,thresh=4,disp=False):
    FLANN_INDEX_KDTREE = 2
    index_params = dict(algorithm=FLANN_INDEX_KDTREE, trees=5)
    search_params = dict(checks=50)
    flann = cv2.FlannBasedMatcher(index_params, search_params)
    #flann = cv2.BFMatcher()
    lff1 = np.float32(descripts[0])
    lff = np.float32(descripts[1])
    matches_lf1_lf = flann.knnMatch(lff1, lff, k=2)
    print("\nNumber of matches",len(matches_lf1_lf))
    matches_4 = []
    ratio = ratio
    # loop over the raw matches
    for m in matches_lf1_lf:
        # ensure the distance is within a certain ratio of each
        # other (i.e. Lowe’s ratio test)
        if len(m) == 2 and m[0].distance < m[1].distance * ratio:

            matches_4.append(m[0])
    print("Number of matches After Lowe's Ratio",len(matches_4))
    matches_idx = np.array([m.queryIdx for m in matches_4])
    imm1_pts = np.array([keypts[0][idx].pt for idx in matches_idx])
    matche_idx = np.array([m.trainIdx for m in matches_4])
    imm2_pts = np.array([keypts[1][idx].pt for idx in matche_idx])

    '''
    # Estimate homography 1
    #Compute H1
    # Estimate homography 1
    #Compute H1
    imm1_pts=np.empty((len(matches_4),2))
    imm2_pts=np.empty((len(matches_4),2))
    for i in range(0,len(matches_4)):
    m = matches_4[i]
```

```
    (a_x, a_y) = keypts[0][m.queryIdx].pt
    (b_x, b_y) = keypts[1][m.trainIdx].pt
    imm1_pts[i]=(a_x, a_y)
    imm2_pts[i]=(b_x, b_y)
    H=compute_Homography(imm1_pts,imm2_pts)
    #Robustly estimate Homography 1 using RANSAC
    Hn, best_inliers=RANSAC_alg(keypts[0] ,keypts[1], matches_4, nRANSAC=1000, RANSACthre
sh=6)
    '''
    Hn,inliers = compute_homography_fast(imm1_pts,imm2_pts)

    inlier_matchset = np.array(matches_4)[inliers.astype(bool)].tolist()
    print("Number of Robust matches",len(inlier_matchset))
    print("\n")
    '''
    if len(inlier_matchset)<50:
        matches_4 = []
        ratio = 0.67
        # loop over the raw matches
        for m in matches_lf1_lf:
           # ensure the distance is within a certain ratio of each
           # other (i.e. Lowe’s ratio test)
           if len(m) == 2 and m[0].distance < m[1].distance * ratio:
           #matches_1.append((m[0].trainIdx, m[0].queryIdx))
           matches_4.append(m[0])
        print("Number of matches After Lowe's Ratio New",len(matches_4))
        matches_idx = np.array([m.queryIdx for m in matches_4])
        imm1_pts = np.array([keypts[0][idx].pt for idx in matches_idx])
        matches_idx = np.array([m.trainIdx for m in matches_4])
        imm2_pts = np.array([keypts[1][idx].pt for idx in matches_idx])
        Hn,inliers = compute_homography_fast_other(imm1_pts,imm2_pts)
        inlier_matchset = np.array(matches_4)[inliers.astype(bool)].tolist()
        print("Number of Robust matches New",len(inlier_matchset))
        print("\n")
       '''
    #H=compute_Homography(imm1_pts,imm2_pts)
    #Robustly estimate Homography 1 using RANSAC
    #Hn=RANSAC_alg(keypts[0] ,keypts[1], matches_4, nRANSAC=1500, RANSACthresh=6)
    #global inlier_matchset
    if disp==True:
        dispimg1=cv2.drawMatches(imgs[0], keypts[0], imgs[1], keypts[1], inlier_matchset
, None,flags=2)
        displayplot(dispimg1,'Robust Matching between Reference Image and Right Image ')
    return Hn/Hn[2,2], len(matches_lf1_lf), len(inlier_matchset)
```

In [21]:

```
from functools import partial
from tqdm import tqdm
tqdm = partial(tqdm, position=0, leave=True)
```

In [22]:

```
H_left_brisk = []
H_right_brisk = []

num_matches_brisk = []
num_good_matches_brisk = []

for j in tqdm(range(len(images_left))):
    if j==len(images_left)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_left_bgr[j:j+2][::-1],keypoints_all_left
_brisk[j:j+2][::-1],points_all_left_brisk[j:j+2][::-1],descriptors_all_left_brisk[j:j+2]
[::-1])
    H_left_brisk.append(H_a)
    num_matches_brisk.append(matches)
    num_good_matches_brisk.append(gd_matches)

for j in tqdm(range(len(images_right))):
```

```
    if j==len(images_right)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_right_bgr[j:j+2][::-1],keypoints_all_rig
ht_brisk[j:j+2][::-1],points_all_right_brisk[j:j+2][::-1],descriptors_all_right_brisk[j:
j+2][::-1])
    H_right_brisk.append(H_a)
    num_matches_brisk.append(matches)
    num_good_matches_brisk.append(gd_matches)
```

```
  0%|          | 0/61 [00:00<?, ?it/s]
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-22-6e02912d5178> in <module>
      9         break
     10
---> 11     H_a,matches,gd_matches = get_Hmatrix(images_left_bgr[j:j+2][::-1],keypoints_
all_left_brisk[j:j+2][::-1],points_all_left_brisk[j:j+2][::-1],descriptors_all_left_bris
k[j:j+2][::-1])
     12     H_left_brisk.append(H_a)
     13     num_matches_brisk.append(matches)

NameError: name 'keypoints_all_left_brisk' is not defined
```

In [31]:

```
H_left_orb = []
H_right_orb = []

num_matches_orb = []
num_good_matches_orb = []

for j in tqdm(range(len(images_left))):
    if j==len(images_left)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_left_bgr[j:j+2][::-1],keypoints_all_left
_orb[j:j+2][::-1],points_all_left_orb[j:j+2][::-1],descriptors_all_left_orb[j:j+2][::-1]
)
    H_left_orb.append(H_a)
    num_matches_orb.append(matches)
    num_good_matches_orb.append(gd_matches)

for j in tqdm(range(len(images_right))):
    if j==len(images_right)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_right_bgr[j:j+2][::-1],keypoints_all_rig
ht_orb[j:j+2][::-1],points_all_right_orb[j:j+2][::-1],descriptors_all_right_orb[j:j+2][:
:-1])
    H_right_orb.append(H_a)
    num_matches_orb.append(matches)
    num_good_matches_orb.append(gd_matches)
```

```
  2%|          | 1/61 [00:00<00:12,  4.96it/s]
```

```
Number of matches 5000
Number of matches After Lowe's Ratio 200
Number of Robust matches 33



Number of matches 5000
Number of matches After Lowe's Ratio 186
```

```
  5%|          | 3/61 [00:00<00:11,  5.01it/s]
```

```
Number of Robust matches 33



Number of matches 5000
```

```
Number of matches After Lowe's Ratio 144
Number of Robust matches 8


  8%|          | 5/61 [00:00<00:09,  5.67it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 377
Number of Robust matches 186



Number of matches 5000
Number of matches After Lowe's Ratio 392
Number of Robust matches 216


 11%|          | 7/61 [00:01<00:09,  5.92it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 355
Number of Robust matches 140



Number of matches 5000
Number of matches After Lowe's Ratio 383
Number of Robust matches 193


 15%|          | 9/61 [00:01<00:08,  5.98it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 264
Number of Robust matches 112



Number of matches 5000
Number of matches After Lowe's Ratio 409
Number of Robust matches 221


 18%|          | 11/61 [00:01<00:08,  5.86it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 286
Number of Robust matches 157



Number of matches 5000
Number of matches After Lowe's Ratio 367
Number of Robust matches 221


 21%|          | 13/61 [00:02<00:07,  6.02it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 410
Number of Robust matches 246



Number of matches 5000
Number of matches After Lowe's Ratio 399
Number of Robust matches 255
```

```
 25%|██        | 15/61 [00:02<00:07,  5.93it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 467
Number of Robust matches 312



Number of matches 5000
Number of matches After Lowe's Ratio 394
Number of Robust matches 269


 28%|██        | 17/61 [00:02<00:07,  6.02it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 502
Number of Robust matches 351



Number of matches 5000
Number of matches After Lowe's Ratio 431
Number of Robust matches 267


 31%|└└└       | 19/61 [00:03<00:06,  6.04it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 472
Number of Robust matches 280



Number of matches 5000
Number of matches After Lowe's Ratio 565
Number of Robust matches 333


 34%|└└└└      | 21/61 [00:03<00:07,  5.70it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 420
Number of Robust matches 241



Number of matches 5000
Number of matches After Lowe's Ratio 302
Number of Robust matches 176


 38%|└└└└      | 23/61 [00:04<00:09,  4.15it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 374
Number of Robust matches 216



Number of matches 5000
Number of matches After Lowe's Ratio 397
Number of Robust matches 277


 41%|└└└└      | 25/61 [00:04<00:07,  4.95it/s]

Number of matches 5000
```

```
Number of matches After Lowe's Ratio 336
Number of Robust matches 208


Number of matches 5000
Number of matches After Lowe's Ratio 328
Number of Robust matches 181

 44%|[redacted]      | 27/61 [00:04<00:06,  5.53it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 349
Number of Robust matches 146


Number of matches 5000
Number of matches After Lowe's Ratio 395
Number of Robust matches 233

 48%|[redacted]      | 29/61 [00:05<00:05,  5.73it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 333
Number of Robust matches 140


Number of matches 5000
Number of matches After Lowe's Ratio 264
Number of Robust matches 85

 51%|[redacted]      | 31/61 [00:05<00:05,  5.60it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 284
Number of Robust matches 93


Number of matches 5000
Number of matches After Lowe's Ratio 184
Number of Robust matches 48

 52%|[redacted]      | 32/61 [00:05<00:05,  5.40it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 166
Number of Robust matches 15


Number of matches 5000
Number of matches After Lowe's Ratio 321
 56%|[redacted]      | 34/61 [00:06<00:04,  5.73it/s]
Number of Robust matches 150


Number of matches 5000
Number of matches After Lowe's Ratio 414
Number of Robust matches 199
```

```
 59%|▊▊▊▊▊▊▌     | 36/61 [00:06<00:04,  5.81it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 320
Number of Robust matches 129



Number of matches 5000
Number of matches After Lowe's Ratio 310
Number of Robust matches 123


 61%|▊▊▊▊▊▊▊     | 37/61 [00:06<00:04,  5.54it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 262
Number of Robust matches 68



Number of matches 5000
Number of matches After Lowe's Ratio 324
 64%|▊▊▊▊▊▊▊▌    | 39/61 [00:07<00:04,  5.39it/s]

Number of Robust matches 72



Number of matches 5000
Number of matches After Lowe's Ratio 267
Number of Robust matches 70


 67%|▊▊▊▊▊▊▊▊    | 41/61 [00:07<00:03,  5.73it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 330
Number of Robust matches 164



Number of matches 5000
Number of matches After Lowe's Ratio 416
Number of Robust matches 275


 70%|▊▊▊▊▊▊▊▊▊   | 43/61 [00:07<00:03,  5.87it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 486
Number of Robust matches 311



Number of matches 5000
Number of matches After Lowe's Ratio 514
Number of Robust matches 368


 74%|▊▊▊▊▊▊▊▊▌   | 45/61 [00:08<00:02,  5.94it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 504
Number of Robust matches 296
```

```
Number of matches 5000
Number of matches After Lowe's Ratio 489
Number of Robust matches 303


 77%|▉▉▉▉▉▉▉▉  | 47/61 [00:08<00:02,  6.00it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 413
Number of Robust matches 219


Number of matches 5000
Number of matches After Lowe's Ratio 358
Number of Robust matches 170


 80%|▉▉▉▉▉▉▉▉▉  | 49/61 [00:08<00:01,  6.08it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 320
Number of Robust matches 167


Number of matches 5000
Number of matches After Lowe's Ratio 588
Number of Robust matches 446


 84%|▉▉▉▉▉▉▉▉▉  | 51/61 [00:09<00:01,  6.05it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 518
Number of Robust matches 382


Number of matches 5000
Number of matches After Lowe's Ratio 345
Number of Robust matches 201


 87%|▉▉▉▉▉▉▉▉▉▉ | 53/61 [00:09<00:01,  6.09it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 319
Number of Robust matches 172


Number of matches 5000
Number of matches After Lowe's Ratio 448
Number of Robust matches 287


 90%|▉▉▉▉▉▉▉▉▉▉ | 55/61 [00:09<00:00,  6.07it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 300
Number of Robust matches 145


Number of matches 5000
Number of matches After Lowe's Ratio 364
Number of Robust matches 234
```

```
 93%|██████████| 57/61 [00:10<00:00,  6.08it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 349
Number of Robust matches 157



Number of matches 5000
Number of matches After Lowe's Ratio 388
Number of Robust matches 169


 97%|██████████| 59/61 [00:10<00:00,  5.94it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 308
Number of Robust matches 91



Number of matches 5000
Number of matches After Lowe's Ratio 403
Number of Robust matches 141


 98%|██████████| 60/61 [00:10<00:00,  5.66it/s]
  0%|          | 0/39 [00:00<?, ?it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 191
Number of Robust matches 24



Number of matches 5000
Number of matches After Lowe's Ratio 505
  5%|▌         | 2/39 [00:00<00:06,  6.09it/s]
Number of Robust matches 326



Number of matches 5000
Number of matches After Lowe's Ratio 503
Number of Robust matches 356


 10%|▌         | 4/39 [00:00<00:05,  5.92it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 343
Number of Robust matches 193



Number of matches 5000
Number of matches After Lowe's Ratio 216
Number of Robust matches 62


 15%|▌▌        | 6/39 [00:01<00:05,  5.98it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 410
Number of Robust matches 278
```

```
Number of matches 5000
Number of matches After Lowe's Ratio 303
Number of Robust matches 162


 21%|██        | 8/39 [00:01<00:06,  5.08it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 522
Number of Robust matches 342



Number of matches 5000
Number of matches After Lowe's Ratio 631
Number of Robust matches 452


 26%|└└└        | 10/39 [00:01<00:05,  5.53it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 415
Number of Robust matches 263



Number of matches 5000
Number of matches After Lowe's Ratio 474
Number of Robust matches 342


 31%|└└└        | 12/39 [00:02<00:04,  5.84it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 332
Number of Robust matches 185



Number of matches 5000
Number of matches After Lowe's Ratio 506
Number of Robust matches 337


 36%|└└└└       | 14/39 [00:02<00:04,  5.94it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 410
Number of Robust matches 275



Number of matches 5000
Number of matches After Lowe's Ratio 457
Number of Robust matches 278


 41%|└└└└       | 16/39 [00:02<00:03,  5.97it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 402
Number of Robust matches 226



Number of matches 5000
Number of matches After Lowe's Ratio 364
```

```
Number of Robust matches 176


 46%|[redacted]      | 18/39 [00:03<00:03,  5.99it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 384
Number of Robust matches 190



Number of matches 5000
Number of matches After Lowe's Ratio 397
Number of Robust matches 188


 51%|[redacted]      | 20/39 [00:03<00:03,  5.97it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 424
Number of Robust matches 178



Number of matches 5000
Number of matches After Lowe's Ratio 412
Number of Robust matches 204


 54%|[redacted]      | 21/39 [00:03<00:03,  6.00it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 406
Number of Robust matches 247



Number of matches 5000
Number of matches After Lowe's Ratio 150
 56%|[redacted]      | 22/39 [00:03<00:03,  5.57it/s]

Number of Robust matches 11



Number of matches 5000
Number of matches After Lowe's Ratio 221
 59%|[redacted]      | 23/39 [00:04<00:03,  5.04it/s]

Number of Robust matches 44


 62%|[redacted]      | 24/39 [00:04<00:03,  4.12it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 139
Number of Robust matches 5


 64%|[redacted]      | 25/39 [00:04<00:03,  4.07it/s]

Number of matches 5000
Number of matches After Lowe's Ratio 230
Number of Robust matches 50


 69%|[redacted]      | 27/39 [00:05<00:02,  4.84it/s]
```

```
Number of matches 5000
Number of matches After Lowe's Ratio 401
Number of Robust matches 197


Number of matches 5000
Number of matches After Lowe's Ratio 396
Number of Robust matches 170

 74%|[redacted]  | 29/39 [00:05<00:01,  5.30it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 371
Number of Robust matches 164


Number of matches 5000
Number of matches After Lowe's Ratio 311
Number of Robust matches 90

 79%|[redacted]  | 31/39 [00:05<00:01,  5.53it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 291
Number of Robust matches 77


Number of matches 5000
Number of matches After Lowe's Ratio 417
Number of Robust matches 158

 82%|[redacted]  | 32/39 [00:05<00:01,  5.60it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 241
Number of Robust matches 75

 87%|[redacted]  | 34/39 [00:06<00:00,  5.62it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 395
Number of Robust matches 205


Number of matches 5000
Number of matches After Lowe's Ratio 329
Number of Robust matches 137

 92%|[redacted]  | 36/39 [00:06<00:00,  5.88it/s]
Number of matches 5000
Number of matches After Lowe's Ratio 318
Number of Robust matches 138


Number of matches 5000
Number of matches After Lowe's Ratio 309
Number of Robust matches 177
```

```
 97%|██████████| 38/39 [00:06<00:00,  5.51it/s]
```

```
Number of matches 5000
Number of matches After Lowe's Ratio 555
Number of Robust matches 375



Number of matches 5000
Number of matches After Lowe's Ratio 420
Number of Robust matches 276
```

In [24]:

```
H_left_akaze = []
H_right_akaze = []

num_matches_akaze = []
num_good_matches_akaze = []

for j in tqdm(range(len(images_left))):
    if j==len(images_left)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_left_bgr[j:j+2][::-1],keypoints_all_left
_akaze[j:j+2][::-1],points_all_left_akaze[j:j+2][::-1],descriptors_all_left_akaze[j:j+2]
[::-1])
    H_left_akaze.append(H_a)
    num_matches_akaze.append(matches)
    num_good_matches_akaze.append(gd_matches)

for j in tqdm(range(len(images_right))):
    if j==len(images_right)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_right_bgr[j:j+2][::-1],keypoints_all_rig
ht_akaze[j:j+2][::-1],points_all_right_akaze[j:j+2][::-1],descriptors_all_right_akaze[j:
j+2][::-1])
    H_right_akaze.append(H_a)
    num_matches_akaze.append(matches)
    num_good_matches_akaze.append(gd_matches)
```

```
  2%|          | 1/61 [00:01<01:04,  1.07s/it]
```

```
Number of matches 16167
Number of matches After Lowe's Ratio 1027
Number of Robust matches 472
```

```
  3%|          | 2/61 [00:02<01:09,  1.18s/it]
```

```
Number of matches 21720
Number of matches After Lowe's Ratio 883
Number of Robust matches 283
```

```
  5%|          | 3/61 [00:03<01:16,  1.32s/it]
```

```
Number of matches 18547
Number of matches After Lowe's Ratio 422
Number of Robust matches 26
```

```
  7%|          | 4/61 [00:05<01:13,  1.28s/it]
```

```
Number of matches 18110
```

```
Number of matches 18110
Number of matches After Lowe's Ratio 2296
Number of Robust matches 1206


  8%|          | 5/61 [00:06<01:10,  1.25s/it]

Number of matches 18754
Number of matches After Lowe's Ratio 2318
Number of Robust matches 1238


 10%|          | 6/61 [00:07<01:13,  1.33s/it]

Number of matches 19049
Number of matches After Lowe's Ratio 2382
Number of Robust matches 1229


 11%|          | 7/61 [00:09<01:11,  1.33s/it]

Number of matches 20428
Number of matches After Lowe's Ratio 2502
Number of Robust matches 1488


 13%|          | 8/61 [00:10<01:10,  1.33s/it]

Number of matches 15748
Number of matches After Lowe's Ratio 1193
Number of Robust matches 684


 15%|          | 9/61 [00:11<01:04,  1.25s/it]

Number of matches 21469
Number of matches After Lowe's Ratio 2080
Number of Robust matches 1322


 16%|          | 10/61 [00:12<01:06,  1.31s/it]

Number of matches 15853
Number of matches After Lowe's Ratio 1110
Number of Robust matches 613


 18%|          | 11/61 [00:13<01:01,  1.24s/it]

Number of matches 20465
Number of matches After Lowe's Ratio 2253
Number of Robust matches 1560


 20%|          | 12/61 [00:15<01:04,  1.32s/it]

Number of matches 19280
Number of matches After Lowe's Ratio 2561
Number of Robust matches 1879


 21%|          | 13/61 [00:16<01:04,  1.35s/it]

Number of matches 21349
Number of matches After Lowe's Ratio 2759
Number of Robust matches 2022
```

```
 23%|[redacted]        | 14/61 [00:19<01:14,  1.58s/it]

Number of matches 21345
Number of matches After Lowe's Ratio 4201
Number of Robust matches 2742


 25%|[redacted]        | 15/61 [00:20<01:11,  1.54s/it]

Number of matches 20541
Number of matches After Lowe's Ratio 3169
Number of Robust matches 2464


 26%|[redacted]        | 16/61 [00:22<01:09,  1.54s/it]

Number of matches 19544
Number of matches After Lowe's Ratio 3430
Number of Robust matches 2717


 28%|[redacted]        | 17/61 [00:23<01:05,  1.48s/it]

Number of matches 19557
Number of matches After Lowe's Ratio 3276
Number of Robust matches 2551


 30%|[redacted]        | 18/61 [00:24<01:01,  1.44s/it]

Number of matches 19398
Number of matches After Lowe's Ratio 4104
Number of Robust matches 3182


 31%|[redacted]        | 19/61 [00:26<00:59,  1.41s/it]

Number of matches 19838
Number of matches After Lowe's Ratio 4527
Number of Robust matches 3136


 33%|[redacted]        | 20/61 [00:27<00:58,  1.43s/it]

Number of matches 19744
Number of matches After Lowe's Ratio 4034
Number of Robust matches 2946


 34%|[redacted]        | 21/61 [00:28<00:56,  1.42s/it]

Number of matches 20624
Number of matches After Lowe's Ratio 3333
Number of Robust matches 2497


 36%|[redacted]        | 22/61 [00:30<00:58,  1.49s/it]

Number of matches 19950
Number of matches After Lowe's Ratio 3080
Number of Robust matches 1862


 38%|[redacted]        | 23/61 [00:31<00:55,  1.47s/it]
```

```
Number of matches 20566
Number of matches After Lowe's Ratio 2834
Number of Robust matches 1833

 39%|[redacted]| 24/61 [00:33<00:54,  1.49s/it]

Number of matches 20559
Number of matches After Lowe's Ratio 2283
Number of Robust matches 1611

 41%|[redacted]| 25/61 [00:34<00:53,  1.48s/it]

Number of matches 24258
Number of matches After Lowe's Ratio 2044
Number of Robust matches 1057

 43%|[redacted]| 26/61 [00:36<00:54,  1.55s/it]

Number of matches 20958
Number of matches After Lowe's Ratio 1742
Number of Robust matches 792

 44%|[redacted]| 27/61 [00:38<00:51,  1.52s/it]

Number of matches 22246
Number of matches After Lowe's Ratio 2136
Number of Robust matches 985

 46%|[redacted]| 28/61 [00:39<00:50,  1.53s/it]

Number of matches 20947
Number of matches After Lowe's Ratio 2148
Number of Robust matches 1004

 48%|[redacted]| 29/61 [00:41<00:52,  1.63s/it]

Number of matches 24081
Number of matches After Lowe's Ratio 1473
Number of Robust matches 652

 49%|[redacted]| 30/61 [00:43<00:51,  1.65s/it]

Number of matches 22618
Number of matches After Lowe's Ratio 1573
Number of Robust matches 788

 51%|[redacted]| 31/61 [00:44<00:49,  1.64s/it]

Number of matches 23539
Number of matches After Lowe's Ratio 923
Number of Robust matches 314

 52%|[redacted]| 32/61 [00:46<00:48,  1.67s/it]

Number of matches 19832
Number of matches After Lowe's Ratio 504
Number of Robust matches 83
```

```
 54%|      | 33/61 [00:48<00:44,  1.61s/it]

Number of matches 19393
Number of matches After Lowe's Ratio 1909
Number of Robust matches 1160


 56%|      | 34/61 [00:49<00:42,  1.59s/it]

Number of matches 17976
Number of matches After Lowe's Ratio 1998
Number of Robust matches 1190


 57%|      | 35/61 [00:51<00:39,  1.53s/it]

Number of matches 19408
Number of matches After Lowe's Ratio 2039
Number of Robust matches 1310


 59%|      | 36/61 [00:52<00:38,  1.55s/it]

Number of matches 23039
Number of matches After Lowe's Ratio 2408
Number of Robust matches 1308


 61%|      | 37/61 [00:54<00:38,  1.62s/it]

Number of matches 26557
Number of matches After Lowe's Ratio 1982
Number of Robust matches 848


 62%|      | 38/61 [00:56<00:39,  1.73s/it]

Number of matches 28674
Number of matches After Lowe's Ratio 2327
Number of Robust matches 838


 64%|      | 39/61 [00:58<00:40,  1.86s/it]

Number of matches 25251
Number of matches After Lowe's Ratio 2351
Number of Robust matches 993


 66%|      | 40/61 [01:00<00:38,  1.85s/it]

Number of matches 22062
Number of matches After Lowe's Ratio 2493
Number of Robust matches 1254


 67%|      | 41/61 [01:02<00:36,  1.80s/it]

Number of matches 20521
Number of matches After Lowe's Ratio 2753
Number of Robust matches 1769


 69%|      | 42/61 [01:03<00:33,  1.75s/it]
```

```
Number of matches 19126
Number of matches After Lowe's Ratio 2988
Number of Robust matches 2072


 70%|[redacted]  | 43/61 [01:05<00:29,  1.62s/it]

Number of matches 20186
Number of matches After Lowe's Ratio 3476
Number of Robust matches 2474


 72%|[redacted]  | 44/61 [01:06<00:26,  1.55s/it]

Number of matches 21213
Number of matches After Lowe's Ratio 3204
Number of Robust matches 2093


 74%|[redacted]  | 45/61 [01:08<00:25,  1.57s/it]

Number of matches 21932
Number of matches After Lowe's Ratio 3666
Number of Robust matches 2374


 75%|[redacted]  | 46/61 [01:09<00:23,  1.56s/it]

Number of matches 21264
Number of matches After Lowe's Ratio 3930
Number of Robust matches 2868


 77%|[redacted]  | 47/61 [01:11<00:21,  1.53s/it]

Number of matches 20833
Number of matches After Lowe's Ratio 3448
Number of Robust matches 1985


 79%|[redacted]  | 48/61 [01:12<00:19,  1.49s/it]

Number of matches 18699
Number of matches After Lowe's Ratio 2103
Number of Robust matches 1491


 80%|[redacted]  | 49/61 [01:13<00:17,  1.49s/it]

Number of matches 17733
Number of matches After Lowe's Ratio 3792
Number of Robust matches 3082


 82%|[redacted]  | 50/61 [01:15<00:15,  1.44s/it]

Number of matches 18293
Number of matches After Lowe's Ratio 3550
Number of Robust matches 2894


 84%|[redacted]  | 51/61 [01:16<00:13,  1.36s/it]

Number of matches 16473
Number of matches After Lowe's Ratio 2108
Number of Robust matches 1521
```

```
 85%|▉▉▉▉▉▉▉▉▉ | 52/61 [01:17<00:11,  1.28s/it]

Number of matches 17759
Number of matches After Lowe's Ratio 2209
Number of Robust matches 1531


 87%|▉▉▉▉▉▉▉▉▉ | 53/61 [01:18<00:09,  1.24s/it]

Number of matches 18253
Number of matches After Lowe's Ratio 2937
Number of Robust matches 2200


 89%|▉▉▉▉▉▉▉▉▉ | 54/61 [01:19<00:08,  1.23s/it]

Number of matches 18717
Number of matches After Lowe's Ratio 1947
Number of Robust matches 1277


 90%|▉▉▉▉▉▉▉▉▉▉ | 55/61 [01:21<00:07,  1.27s/it]

Number of matches 18943
Number of matches After Lowe's Ratio 2328
Number of Robust matches 1777


 92%|▉▉▉▉▉▉▉▉▉▉ | 56/61 [01:22<00:06,  1.39s/it]

Number of matches 18446
Number of matches After Lowe's Ratio 2235
Number of Robust matches 1295


 93%|▉▉▉▉▉▉▉▉▉▉ | 57/61 [01:24<00:05,  1.34s/it]

Number of matches 19154
Number of matches After Lowe's Ratio 2746
Number of Robust matches 1404


 95%|▉▉▉▉▉▉▉▉▉▉▉| 58/61 [01:25<00:04,  1.42s/it]

Number of matches 20674
Number of matches After Lowe's Ratio 1749
Number of Robust matches 675


 97%|▉▉▉▉▉▉▉▉▉▉▉| 59/61 [01:27<00:02,  1.41s/it]

Number of matches 20317
Number of matches After Lowe's Ratio 2097
Number of Robust matches 812


 98%|▉▉▉▉▉▉▉▉▉▉▉| 60/61 [01:28<00:01,  1.47s/it]
  0%|          | 0/39 [00:00<?, ?it/s]

Number of matches 17535
Number of matches After Lowe's Ratio 777
Number of Robust matches 206
```

```
  3%|▌          | 1/39 [00:01<00:52,  1.37s/it]
```

```
Number of matches 20771
Number of matches After Lowe's Ratio 2619
Number of Robust matches 1919
```

```
  5%|▌          | 2/39 [00:02<00:50,  1.38s/it]
```

```
Number of matches 18297
Number of matches After Lowe's Ratio 2846
Number of Robust matches 1976
```

```
  8%|▌          | 3/39 [00:03<00:46,  1.29s/it]
```

```
Number of matches 15660
Number of matches After Lowe's Ratio 1364
Number of Robust matches 882
```

```
 10%|▌          | 4/39 [00:04<00:41,  1.19s/it]
```

```
Number of matches 19968
Number of matches After Lowe's Ratio 633
Number of Robust matches 202
```

```
 13%|▌          | 5/39 [00:06<00:41,  1.23s/it]
```

```
Number of matches 15728
Number of matches After Lowe's Ratio 2085
Number of Robust matches 1388
```

```
 15%|▌          | 6/39 [00:07<00:42,  1.27s/it]
```

```
Number of matches 21692
Number of matches After Lowe's Ratio 1805
Number of Robust matches 1140
```

```
 18%|▌          | 7/39 [00:09<00:44,  1.40s/it]
```

```
Number of matches 21865
Number of matches After Lowe's Ratio 3684
Number of Robust matches 2637
```

```
 21%|▌          | 8/39 [00:10<00:44,  1.44s/it]
```

```
Number of matches 21455
Number of matches After Lowe's Ratio 5018
Number of Robust matches 3402
```

```
 23%|▌          | 9/39 [00:12<00:43,  1.44s/it]
```

```
Number of matches 19571
Number of matches After Lowe's Ratio 4095
Number of Robust matches 3031
```

```
 26%|▌          | 10/39 [00:13<00:41,  1.45s/it]
```

```
Number of matches 20134
Number of matches After Lowe's Ratio 3944
```

```
Number of Robust matches 3181

 28%|██▌       | 11/39 [00:15<00:39,  1.43s/it]

Number of matches 21099
Number of matches After Lowe's Ratio 2855
Number of Robust matches 1999

 31%|███       | 12/39 [00:16<00:39,  1.45s/it]

Number of matches 22136
Number of matches After Lowe's Ratio 3595
Number of Robust matches 2455

 33%|▌         | 13/39 [00:18<00:40,  1.55s/it]

Number of matches 23198
Number of matches After Lowe's Ratio 2988
Number of Robust matches 1892

 36%|▌         | 14/39 [00:20<00:40,  1.63s/it]

Number of matches 24310
Number of matches After Lowe's Ratio 3010
Number of Robust matches 1586

 38%|▌         | 15/39 [00:21<00:40,  1.67s/it]

Number of matches 24654
Number of matches After Lowe's Ratio 3191
Number of Robust matches 1752

 41%|▌         | 16/39 [00:23<00:39,  1.73s/it]

Number of matches 22984
Number of matches After Lowe's Ratio 3166
Number of Robust matches 1738

 44%|▌         | 17/39 [00:25<00:40,  1.86s/it]

Number of matches 21716
Number of matches After Lowe's Ratio 2720
Number of Robust matches 1198

 46%|▌         | 18/39 [00:27<00:37,  1.81s/it]

Number of matches 22411
Number of matches After Lowe's Ratio 3365
Number of Robust matches 1581

 49%|▌         | 19/39 [00:29<00:35,  1.80s/it]

Number of matches 20504
Number of matches After Lowe's Ratio 2741
Number of Robust matches 1209
```

```
 51%|      | 20/39 [00:30<00:32,  1.69s/it]

Number of matches 20204
Number of matches After Lowe's Ratio 2764
Number of Robust matches 1301


 54%|      | 21/39 [00:32<00:28,  1.60s/it]

Number of matches 20233
Number of matches After Lowe's Ratio 2184
Number of Robust matches 1186


 56%|      | 22/39 [00:33<00:26,  1.56s/it]

Number of matches 27287
Number of matches After Lowe's Ratio 812
Number of Robust matches 220


 59%|      | 23/39 [00:35<00:27,  1.73s/it]

Number of matches 25609
Number of matches After Lowe's Ratio 1295
Number of Robust matches 387


 62%|      | 24/39 [00:37<00:27,  1.80s/it]

Number of matches 29338
Number of matches After Lowe's Ratio 484
Number of Robust matches 6


 64%|      | 25/39 [00:40<00:27,  1.95s/it]

Number of matches 24561
Number of matches After Lowe's Ratio 1234
Number of Robust matches 425


 67%|      | 26/39 [00:41<00:24,  1.92s/it]

Number of matches 22164
Number of matches After Lowe's Ratio 2321
Number of Robust matches 1016


 69%|      | 27/39 [00:43<00:22,  1.84s/it]

Number of matches 20348
Number of matches After Lowe's Ratio 2210
Number of Robust matches 885


 72%|      | 28/39 [00:45<00:18,  1.70s/it]

Number of matches 19519
Number of matches After Lowe's Ratio 1941
Number of Robust matches 888


 74%|      | 29/39 [00:46<00:15,  1.60s/it]

Number of matches 20911
Number of matches After Lowe's Ratio 1940
```

```
Number of matches After Lowe's Ratio 1949
Number of Robust matches 636

 77%|[redacted]| 30/39 [00:47<00:13,  1.55s/it]

Number of matches 20565
Number of matches After Lowe's Ratio 1753
Number of Robust matches 755

 79%|[redacted]| 31/39 [00:49<00:12,  1.56s/it]

Number of matches 20347
Number of matches After Lowe's Ratio 3708
Number of Robust matches 1725

 82%|[redacted]| 32/39 [00:50<00:10,  1.52s/it]

Number of matches 22441
Number of matches After Lowe's Ratio 1750
Number of Robust matches 754

 85%|[redacted]| 33/39 [00:52<00:09,  1.60s/it]

Number of matches 21870
Number of matches After Lowe's Ratio 3154
Number of Robust matches 1609

 87%|[redacted]| 34/39 [00:54<00:07,  1.58s/it]

Number of matches 20671
Number of matches After Lowe's Ratio 2425
Number of Robust matches 1294

 90%|[redacted]| 35/39 [00:55<00:06,  1.57s/it]

Number of matches 18732
Number of matches After Lowe's Ratio 2420
Number of Robust matches 1538

 92%|[redacted]| 36/39 [00:57<00:04,  1.56s/it]

Number of matches 18770
Number of matches After Lowe's Ratio 1827
Number of Robust matches 1201

 95%|[redacted]| 37/39 [00:58<00:03,  1.54s/it]

Number of matches 18471
Number of matches After Lowe's Ratio 1778
Number of Robust matches 1290

 97%|[redacted]| 38/39 [00:59<00:01,  1.58s/it]

Number of matches 17991
Number of matches After Lowe's Ratio 2165
Number of Robust matches 1337
```

In [21]:

```
H_left_kaze = []
H_right_kaze = []

num_matches_kaze = []
num_good_matches_kaze = []

for j in tqdm(range(len(images_left))):
    if j==len(images_left)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_left_bgr[j:j+2][::-1],keypoints_all_left
_kaze[j:j+2][::-1],points_all_left_kaze[j:j+2][::-1],descriptors_all_left_kaze[j:j+2][::
-1])
    H_left_kaze.append(H_a)
    num_matches_kaze.append(matches)
    num_good_matches_kaze.append(gd_matches)

for j in tqdm(range(len(images_right))):
    if j==len(images_right)-1:
        break

    H_a,matches,gd_matches = get_Hmatrix(images_right_bgr[j:j+2][::-1],keypoints_all_rig
ht_kaze[j:j+2][::-1],points_all_right_kaze[j:j+2][::-1],descriptors_all_right_kaze[j:j+2
][::-1])
    H_right_kaze.append(H_a)
    num_matches_kaze.append(matches)
    num_good_matches_kaze.append(gd_matches)
```

```
  2%|          | 1/61 [00:01<01:17,  1.29s/it]
```

```
Number of matches 17737
Number of matches After Lowe's Ratio 2181
Number of Robust matches 1031
```

```
  3%|          | 2/61 [00:02<01:19,  1.35s/it]
```

```
Number of matches 23066
Number of matches After Lowe's Ratio 1974
Number of Robust matches 746
```

```
  5%|          | 3/61 [00:04<01:32,  1.60s/it]
```

```
Number of matches 20398
Number of matches After Lowe's Ratio 662
Number of Robust matches 155
```

```
  7%|          | 4/61 [00:06<01:32,  1.62s/it]
```

```
Number of matches 19469
Number of matches After Lowe's Ratio 5129
Number of Robust matches 2523
```

```
  8%|          | 5/61 [00:07<01:30,  1.62s/it]
```

```
Number of matches 20021
Number of matches After Lowe's Ratio 5533
Number of Robust matches 2936
```

```
 10%|          | 6/61 [00:09<01:37,  1.77s/it]
```

```
Number of matches 20426
```

```
Number of matches 20426
Number of matches After Lowe's Ratio 5250
Number of Robust matches 2674


 11%|▉         | 7/61 [00:11<01:32,  1.72s/it]

Number of matches 22778
Number of matches After Lowe's Ratio 5894
Number of Robust matches 3703


 13%|▉         | 8/61 [00:13<01:30,  1.71s/it]

Number of matches 17408
Number of matches After Lowe's Ratio 2954
Number of Robust matches 1589


 15%|▉         | 9/61 [00:14<01:23,  1.62s/it]

Number of matches 23670
Number of matches After Lowe's Ratio 4657
Number of Robust matches 2889


 16%|▉         | 10/61 [00:16<01:27,  1.71s/it]

Number of matches 18203
Number of matches After Lowe's Ratio 2739
Number of Robust matches 1938


 18%|▉         | 11/61 [00:18<01:25,  1.71s/it]

Number of matches 22285
Number of matches After Lowe's Ratio 5440
Number of Robust matches 3699


 20%|▉         | 12/61 [00:19<01:24,  1.71s/it]

Number of matches 22018
Number of matches After Lowe's Ratio 6488
Number of Robust matches 4588


 21%|▉         | 13/61 [00:21<01:23,  1.74s/it]

Number of matches 24330
Number of matches After Lowe's Ratio 6340
Number of Robust matches 3890


 23%|▉         | 14/61 [00:23<01:27,  1.85s/it]

Number of matches 24904
Number of matches After Lowe's Ratio 9111
Number of Robust matches 6213


 25%|▉         | 15/61 [00:25<01:27,  1.89s/it]

Number of matches 23410
Number of matches After Lowe's Ratio 6899
Number of Robust matches 5111
```

```
 26%|[redacted]        | 16/61 [00:27<01:24,  1.87s/it]

Number of matches 21871
Number of matches After Lowe's Ratio 7042
Number of Robust matches 5262


 28%|[redacted]        | 17/61 [00:29<01:25,  1.94s/it]

Number of matches 21387
Number of matches After Lowe's Ratio 6429
Number of Robust matches 4739


 30%|[redacted]        | 18/61 [00:31<01:19,  1.86s/it]

Number of matches 21095
Number of matches After Lowe's Ratio 7156
Number of Robust matches 5133


 31%|[redacted]        | 19/61 [00:33<01:16,  1.82s/it]

Number of matches 21423
Number of matches After Lowe's Ratio 8127
Number of Robust matches 6000


 33%|[redacted]        | 20/61 [00:34<01:12,  1.76s/it]

Number of matches 20665
Number of matches After Lowe's Ratio 6675
Number of Robust matches 5102


 34%|[redacted]        | 21/61 [00:36<01:10,  1.75s/it]

Number of matches 21776
Number of matches After Lowe's Ratio 5895
Number of Robust matches 3946


 36%|[redacted]        | 22/61 [00:38<01:07,  1.74s/it]

Number of matches 21075
Number of matches After Lowe's Ratio 5746
Number of Robust matches 3765


 38%|[redacted]        | 23/61 [00:40<01:07,  1.77s/it]

Number of matches 22361
Number of matches After Lowe's Ratio 5830
Number of Robust matches 3541


 39%|[redacted]        | 24/61 [00:42<01:12,  1.95s/it]

Number of matches 21739
Number of matches After Lowe's Ratio 4885
Number of Robust matches 3099


 41%|[redacted]        | 25/61 [00:44<01:08,  1.90s/it]
```

```
Number of matches 24424
Number of matches After Lowe's Ratio 4477
Number of Robust matches 2505


 43%|[redacted]| 26/61 [00:46<01:06,  1.89s/it]

Number of matches 21469
Number of matches After Lowe's Ratio 4739
Number of Robust matches 2609


 44%|[redacted]| 27/61 [00:47<01:02,  1.83s/it]

Number of matches 22552
Number of matches After Lowe's Ratio 4907
Number of Robust matches 2582


 46%|[redacted]| 28/61 [00:49<01:01,  1.86s/it]

Number of matches 21879
Number of matches After Lowe's Ratio 4762
Number of Robust matches 1892


 48%|[redacted]| 29/61 [00:51<01:01,  1.92s/it]

Number of matches 25013
Number of matches After Lowe's Ratio 3136
Number of Robust matches 1119


 49%|[redacted]| 30/61 [00:53<01:00,  1.95s/it]

Number of matches 24466
Number of matches After Lowe's Ratio 4455
Number of Robust matches 2060


 51%|[redacted]| 31/61 [00:55<01:00,  2.00s/it]

Number of matches 24798
Number of matches After Lowe's Ratio 2438
Number of Robust matches 1019


 52%|[redacted]| 32/61 [00:58<00:59,  2.05s/it]

Number of matches 21670
Number of matches After Lowe's Ratio 1129
Number of Robust matches 302


 54%|[redacted]| 33/61 [00:59<00:54,  1.94s/it]

Number of matches 21083
Number of matches After Lowe's Ratio 4217
Number of Robust matches 2098


 56%|[redacted]| 34/61 [01:01<00:52,  1.94s/it]

Number of matches 19041
Number of matches After Lowe's Ratio 4723
Number of Robust matches 2624
```

```
 57%|▌     | 35/61 [01:03<00:47,  1.82s/it]

Number of matches 21135
Number of matches After Lowe's Ratio 4207
Number of Robust matches 2197


 59%|▌     | 36/61 [01:05<00:46,  1.86s/it]

Number of matches 25378
Number of matches After Lowe's Ratio 5543
Number of Robust matches 2790


 61%|▌     | 37/61 [01:07<00:47,  2.00s/it]

Number of matches 28928
Number of matches After Lowe's Ratio 5192
Number of Robust matches 2185


 62%|▌     | 38/61 [01:10<00:51,  2.22s/it]

Number of matches 31135
Number of matches After Lowe's Ratio 7036
Number of Robust matches 3376


 64%|▌     | 39/61 [01:13<00:55,  2.54s/it]

Number of matches 28082
Number of matches After Lowe's Ratio 6800
Number of Robust matches 2867


 66%|▌     | 40/61 [01:16<00:52,  2.51s/it]

Number of matches 24578
Number of matches After Lowe's Ratio 6470
Number of Robust matches 3690


 67%|▌     | 41/61 [01:18<00:47,  2.39s/it]

Number of matches 22760
Number of matches After Lowe's Ratio 6519
Number of Robust matches 3667


 69%|▌     | 42/61 [01:20<00:43,  2.27s/it]

Number of matches 21201
Number of matches After Lowe's Ratio 6230
Number of Robust matches 3953


 70%|▌     | 43/61 [01:21<00:37,  2.09s/it]

Number of matches 21619
Number of matches After Lowe's Ratio 7175
Number of Robust matches 5282


 72%|▌     | 44/61 [01:23<00:34,  2.04s/it]
```

```
Number of matches 23323
Number of matches After Lowe's Ratio 6338
Number of Robust matches 4132


 74%|[redacted]  | 45/61 [01:25<00:32,  2.02s/it]

Number of matches 24274
Number of matches After Lowe's Ratio 7460
Number of Robust matches 4877


 75%|[redacted]  | 46/61 [01:27<00:29,  1.99s/it]

Number of matches 23964
Number of matches After Lowe's Ratio 7777
Number of Robust matches 4608


 77%|[redacted]  | 47/61 [01:29<00:27,  1.96s/it]

Number of matches 23881
Number of matches After Lowe's Ratio 7514
Number of Robust matches 4350


 79%|[redacted]  | 48/61 [01:31<00:25,  1.94s/it]

Number of matches 22006
Number of matches After Lowe's Ratio 4750
Number of Robust matches 2771


 80%|[redacted]  | 49/61 [01:33<00:22,  1.91s/it]

Number of matches 21045
Number of matches After Lowe's Ratio 7775
Number of Robust matches 5496


 82%|[redacted]  | 50/61 [01:35<00:21,  1.92s/it]

Number of matches 21574
Number of matches After Lowe's Ratio 7640
Number of Robust matches 5600


 84%|[redacted]  | 51/61 [01:36<00:18,  1.87s/it]

Number of matches 19699
Number of matches After Lowe's Ratio 4747
Number of Robust matches 3387


 85%|[redacted]  | 52/61 [01:38<00:16,  1.79s/it]

Number of matches 20924
Number of matches After Lowe's Ratio 4990
Number of Robust matches 3455


 87%|[redacted]  | 53/61 [01:40<00:14,  1.79s/it]

Number of matches 21133
Number of matches After Lowe's Ratio 6750
Number of Robust matches 4003
```

```
 89%|▉▉▉▉▉▉▉▉▉ | 54/61 [01:42<00:12,  1.77s/it]

Number of matches 21286
Number of matches After Lowe's Ratio 4918
Number of Robust matches 3207


 90%|▉▉▉▉▉▉▉▉▉▉ | 55/61 [01:43<00:10,  1.74s/it]

Number of matches 20795
Number of matches After Lowe's Ratio 5671
Number of Robust matches 3611


Number of matches 20754
Number of matches After Lowe's Ratio 4884
 92%|▉▉▉▉▉▉▉▉▉▉ | 56/61 [01:46<00:09,  1.95s/it]

Number of Robust matches 2543


 93%|▉▉▉▉▉▉▉▉▉▉ | 57/61 [01:47<00:07,  1.87s/it]

Number of matches 21466
Number of matches After Lowe's Ratio 6503
Number of Robust matches 3376


 95%|▉▉▉▉▉▉▉▉▉▉▉ | 58/61 [01:49<00:05,  1.82s/it]

Number of matches 22210
Number of matches After Lowe's Ratio 4511
Number of Robust matches 1757


 97%|▉▉▉▉▉▉▉▉▉▉▉ | 59/61 [01:51<00:03,  1.79s/it]

Number of matches 22050
Number of matches After Lowe's Ratio 5929
Number of Robust matches 2515


 98%|▉▉▉▉▉▉▉▉▉▉▉ | 60/61 [01:53<00:01,  1.88s/it]
  0%|          | 0/39 [00:00<?, ?it/s]

Number of matches 19110
Number of matches After Lowe's Ratio 1846
Number of Robust matches 581


  3%|▉         | 1/39 [00:01<00:58,  1.54s/it]

Number of matches 22253
Number of matches After Lowe's Ratio 5366
Number of Robust matches 3356


  5%|▉         | 2/39 [00:03<01:06,  1.80s/it]

Number of matches 19990
Number of matches After Lowe's Ratio 5722
Number of Robust matches 3571
```

```
  8%|▉         | 3/39 [00:05<01:00,  1.67s/it]

Number of matches 17969
Number of matches After Lowe's Ratio 2872
Number of Robust matches 1930


 10%|▉         | 4/39 [00:06<00:55,  1.60s/it]

Number of matches 21377
Number of matches After Lowe's Ratio 1764
Number of Robust matches 1052


 13%|▉         | 5/39 [00:08<00:54,  1.61s/it]

Number of matches 18219
Number of matches After Lowe's Ratio 4637
Number of Robust matches 2813


 15%|▉         | 6/39 [00:09<00:51,  1.55s/it]

Number of matches 23113
Number of matches After Lowe's Ratio 3598
Number of Robust matches 2340


 18%|▉         | 7/39 [00:11<00:52,  1.64s/it]

Number of matches 23306
Number of matches After Lowe's Ratio 8476
Number of Robust matches 6435


 21%|▉         | 8/39 [00:13<00:53,  1.74s/it]

Number of matches 22231
Number of matches After Lowe's Ratio 8734
Number of Robust matches 6829


 23%|▉         | 9/39 [00:15<00:54,  1.82s/it]

Number of matches 20707
Number of matches After Lowe's Ratio 6880
Number of Robust matches 5019


 26%|▉         | 10/39 [00:17<00:51,  1.77s/it]

Number of matches 21002
Number of matches After Lowe's Ratio 7096
Number of Robust matches 5626


 28%|▉         | 11/39 [00:18<00:48,  1.72s/it]

Number of matches 22557
Number of matches After Lowe's Ratio 5492
Number of Robust matches 3822


 31%|▉         | 12/39 [00:20<00:48,  1.80s/it]

Number of matches 22744
```

```
Number of matches After Lowe's Ratio 6920
Number of Robust matches 4789


 33%|[redacted]       | 13/39 [00:22<00:47,  1.83s/it]

Number of matches 23731
Number of matches After Lowe's Ratio 6316
Number of Robust matches 3625


 36%|[redacted]       | 14/39 [00:24<00:47,  1.90s/it]

Number of matches 25368
Number of matches After Lowe's Ratio 7207
Number of Robust matches 4211


 38%|[redacted]       | 15/39 [00:27<00:54,  2.26s/it]

Number of matches 25661
Number of matches After Lowe's Ratio 7440
Number of Robust matches 4020


 41%|[redacted]       | 16/39 [00:29<00:50,  2.20s/it]

Number of matches 24425
Number of matches After Lowe's Ratio 7852
Number of Robust matches 3703


 44%|[redacted]       | 17/39 [00:31<00:46,  2.13s/it]

Number of matches 22841
Number of matches After Lowe's Ratio 6375
Number of Robust matches 2666


 46%|[redacted]       | 18/39 [00:33<00:42,  2.02s/it]

Number of matches 22352
Number of matches After Lowe's Ratio 7216
Number of Robust matches 3196


 49%|[redacted]       | 19/39 [00:35<00:39,  1.97s/it]

Number of matches 21771
Number of matches After Lowe's Ratio 5782
Number of Robust matches 2195


 51%|[redacted]       | 20/39 [00:37<00:37,  1.96s/it]

Number of matches 20959
Number of matches After Lowe's Ratio 5512
Number of Robust matches 2156


 54%|[redacted]       | 21/39 [00:38<00:33,  1.85s/it]

Number of matches 20882
Number of matches After Lowe's Ratio 4422
Number of Robust matches 2135
```

```
 56%|      | 22/39 [00:40<00:31,  1.83s/it]
Number of matches 28555
Number of matches After Lowe's Ratio 2124
Number of Robust matches 912

 59%|      | 23/39 [00:43<00:32,  2.02s/it]
Number of matches 26455
Number of matches After Lowe's Ratio 4133
Number of Robust matches 1730

 62%|      | 24/39 [00:45<00:31,  2.08s/it]
Number of matches 31041
Number of matches After Lowe's Ratio 509
Number of Robust matches 46

 64%|      | 25/39 [00:48<00:32,  2.30s/it]
Number of matches 25532
Number of matches After Lowe's Ratio 3326
Number of Robust matches 1253

 67%|      | 26/39 [00:50<00:29,  2.24s/it]
Number of matches 23237
Number of matches After Lowe's Ratio 5693
Number of Robust matches 2534

 69%|      | 27/39 [00:52<00:25,  2.10s/it]
Number of matches 21219
Number of matches After Lowe's Ratio 4965
Number of Robust matches 2195

 72%|      | 28/39 [00:53<00:21,  1.95s/it]
Number of matches 20480
Number of matches After Lowe's Ratio 4380
Number of Robust matches 1643

 74%|      | 29/39 [00:55<00:18,  1.85s/it]
Number of matches 22293
Number of matches After Lowe's Ratio 4715
Number of Robust matches 1789

 77%|      | 30/39 [00:57<00:16,  1.85s/it]
Number of matches 21755
Number of matches After Lowe's Ratio 4668
Number of Robust matches 1911

 79%|      | 31/39 [00:59<00:16,  2.04s/it]
Number of matches 22020
```

```
Number of matches 22020
Number of matches After Lowe's Ratio 7729
Number of Robust matches 3220
```

```
 82%|▉▉▉▉▉▉▉▉  | 32/39 [01:01<00:13,  1.95s/it]
```

```
Number of matches 23245
Number of matches After Lowe's Ratio 4187
Number of Robust matches 1682
```

```
 85%|▉▉▉▉▉▉▉▉  | 33/39 [01:03<00:11,  1.95s/it]
```

```
Number of matches 22554
Number of matches After Lowe's Ratio 7133
Number of Robust matches 3407
```

```
 87%|▉▉▉▉▉▉▉▉  | 34/39 [01:04<00:09,  1.88s/it]
```

```
Number of matches 21359
Number of matches After Lowe's Ratio 5467
Number of Robust matches 2708
```

```
 90%|▉▉▉▉▉▉▉▉  | 35/39 [01:06<00:07,  1.81s/it]
```

```
Number of matches 20079
Number of matches After Lowe's Ratio 4656
Number of Robust matches 2854
```

```
 92%|▉▉▉▉▉▉▉▉  | 36/39 [01:08<00:05,  1.72s/it]
```

```
Number of matches 20334
Number of matches After Lowe's Ratio 3726
Number of Robust matches 2892
```

```
 95%|▉▉▉▉▉▉▉▉  | 37/39 [01:10<00:03,  1.78s/it]
```

```
Number of matches 20129
Number of matches After Lowe's Ratio 4739
Number of Robust matches 3174
```

```
 97%|▉▉▉▉▉▉▉▉  | 38/39 [01:11<00:01,  1.88s/it]
```

```
Number of matches 19798
Number of matches After Lowe's Ratio 4579
Number of Robust matches 2920
```

In [26]:

```
def warpnImages(images_left, images_right,H_left,H_right):
    #img1-centre,img2-left,img3-right

    h, w = images_left[0].shape[:2]

    pts_left = []
    pts_right = []

    pts_centre = np.float32([[0, 0], [0, h], [w, h], [w, 0]]).reshape(-1, 1, 2)
```

```
    for j in range(len(H_left)):
      pts = np.float32([[0, 0], [0, h], [w, h], [w, 0]]).reshape(-1, 1, 2)
      pts_left.append(pts)

    for j in range(len(H_right)):
      pts = np.float32([[0, 0], [0, h], [w, h], [w, 0]]).reshape(-1, 1, 2)
      pts_right.append(pts)

    pts_left_transformed=[]
    pts_right_transformed=[]

    for j,pts in enumerate(pts_left):
      if j==0:
        H_trans = H_left[j]
      else:
        H_trans = H_trans@H_left[j]
      pts_ = cv2.perspectiveTransform(pts, H_trans)
      pts_left_transformed.append(pts_)

    for j,pts in enumerate(pts_right):
      if j==0:
        H_trans = H_right[j]
      else:
        H_trans = H_trans@H_right[j]
      pts_ = cv2.perspectiveTransform(pts, H_trans)
      pts_right_transformed.append(pts_)


    print('Step1:Done')


    #pts = np.concatenate((pts1, pts2_), axis=0)

    pts_concat = np.concatenate((pts_centre,np.concatenate(np.array(pts_left_transformed
),axis=0),np.concatenate(np.array(pts_right_transformed),axis=0)), axis=0)

    [xmin, ymin] = np.int32(pts_concat.min(axis=0).ravel() - 0.5)
    [xmax, ymax] = np.int32(pts_concat.max(axis=0).ravel() + 0.5)
    t = [-xmin, -ymin]
    Ht = np.array([[1, 0, t[0]], [0, 1, t[1]], [0, 0, 1]])  # translate

    print('Step2:Done')


    return xmax,xmin,ymax,ymin,t,h,w,Ht
```

In [27]:

```
def final_steps_left_union(images_left,H_left,xmax,xmin,ymax,ymin,t,h,w,Ht):
    for j,H in enumerate(H_left):
        if j==  0:
            H_trans = Ht@H
        else:
            H_trans = H_trans@H
        result =  cv2.warpPerspective(images_left[j+1],H_trans,(xmax-xmin,ymax-ymin))
        warp_img_init_curr = result

        if j == 0:
            result[t[1]:h+t[1],t[0]:w+t[0]] = images_left[0]
            warp_img_init_prev = result
            continue
        black_pixels = np.where((warp_img_init_prev[:,:,0]==0)&(warp_img_init_prev[:,:,1
]==0)&(warp_img_init_prev[:,:,2]==0))
        warp_img_init_prev[black_pixels] = warp_img_init_curr[black_pixels]

    print('step31:Done')
    return warp_img_init_prev

def final_step_right_union(warp_img_prev,images_right,H_right,xmax,xmin,ymax,ymin,t,h,w,
Ht):
    for j,H in enumerate(H_right):
```

```
        if j==  0:
            H_trans = Ht@H
        else:
            H_trans = H_trans@H
        result =  cv2.warpPerspective(images_right[j+1],H_trans,(xmax-xmin,ymax-ymin))
        warp_img_init_curr = result



        black_pixels = np.where((warp_img_prev[:,:,0]==0)&(warp_img_prev[:,:,1]==0)&(war
p_img_prev[:,:,2]==0))
        warp_img_prev[black_pixels] = warp_img_init_curr[black_pixels]

    print('step32:Done')
    return warp_img_prev
```

In [23]:

```
xmax,xmin,ymax,ymin,t,h,w,Ht = warpnImages(images_left_bgr_no_enhance, images_right_bgr_
no_enhance,H_left_brisk,H_right_brisk)
```

```
Step1:Done
Step2:Done
```

In [24]:

```
warp_imgs_left = final_steps_left_union(images_left_bgr_no_enhance,H_left_brisk,xmax,xmin
,ymax,ymin,t,h,w,Ht)
```

```
step31:Done
```

In [26]:

```
warp_imgs_all_brisk = final_step_right_union(warp_imgs_left,images_right_bgr_no_enhance,H
_right_brisk,xmax,xmin,ymax,ymin,t,h,w,Ht)
```

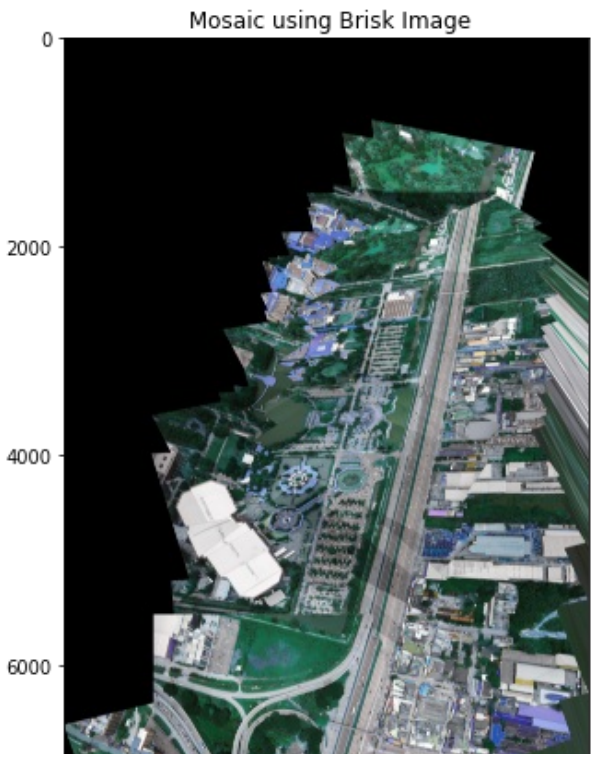
```
step32:Done
```

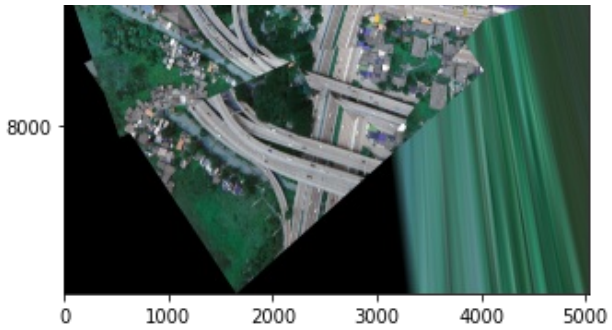
In [28]:

```
plt.figure(figsize=(20,10))
plt.imshow(warp_imgs_all_brisk)
plt.title(' Mosaic using Brisk Image')
```

Out[28]:

```
Text(0.5, 1.0, ' Mosaic using Brisk Image')
```





In [28]:

```
omax,omin,umax,umin,T,H,W,HT = warpnImages(images_left_bgr_no_enhance, images_right_bgr_
no_enhance,H_left_akaze,H_right_akaze)
```

```
Step1:Done
Step2:Done
```

In [29]:

```
warp_img_left = final_steps_left_union(images_left_bgr_no_enhance,H_left_akaze,omax,omin,
umax,umin,T,H,W,HT)
```

```
step31:Done
```

In [30]:

```
warp_imgs_all_akaze = final_step_right_union(warp_img_left,images_right_bgr_no_enhance,H_
right_akaze,omax,omin,umax,umin,T,H,W,HT)
```

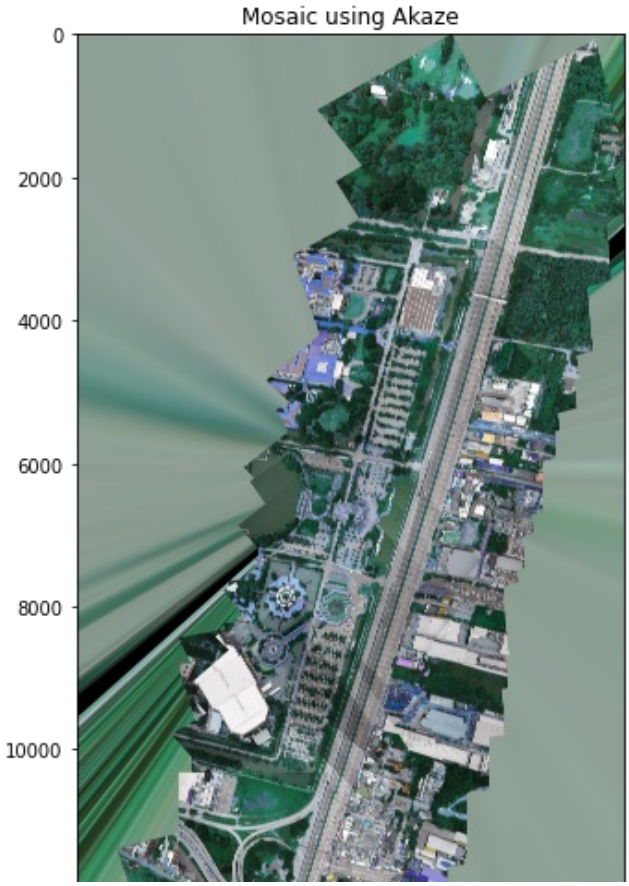
```
step32:Done
```

In [31]:

```
plt.figure(figsize=(20,10))
plt.imshow(warp_imgs_all_akaze)
plt.title('Mosaic using Akaze')
```

Out[31]:

```
Text(0.5, 1.0, 'Mosaic using Akaze')
```



In [24]:

```
amax,amin,zmax,zmin,d,i,q,ht = warpnImages(images_left_bgr_no_enhance, images_right_bgr_
no_enhance,H_left_kaze,H_right_kaze)
```

```
Step1:Done
Step2:Done
```

In [25]:

```
warp_image_left = final_steps_left_union(images_left_bgr_no_enhance,H_left_kaze,amax,amin
,zmax,zmin,d,i,q,ht)
```

```
step31:Done
```

In [27]:

```
warp_imgs_all_akaze = final_step_right_union(warp_image_left,images_right_bgr_no_enhance,
H_right_kaze,amax,amin,zmax,zmin,d,i,q,ht)
```

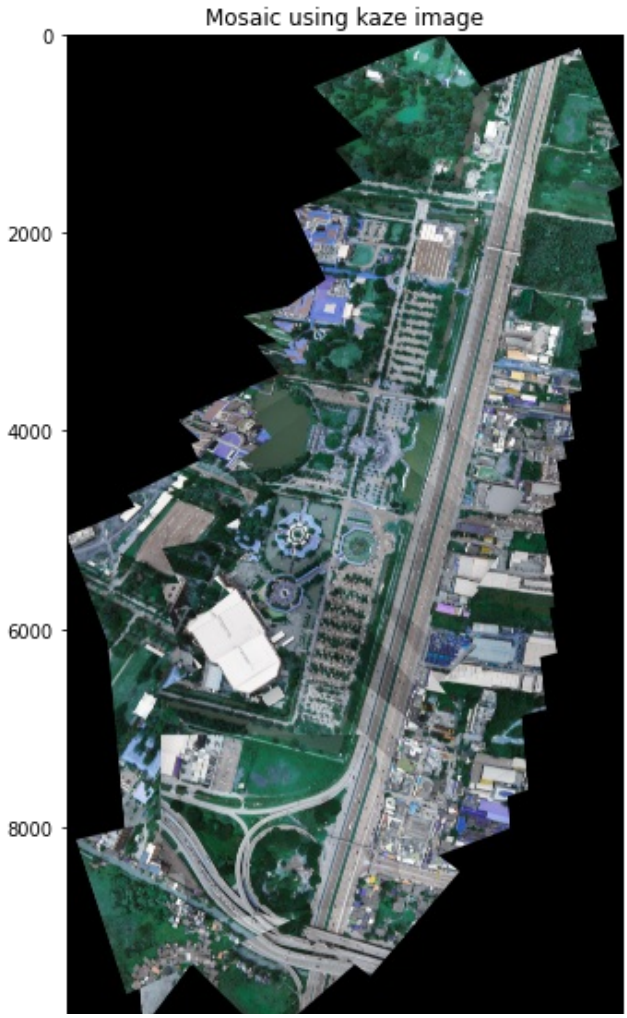
```
step32:Done
```

In [28]:

```
plt.figure(figsize=(20,10))
plt.imshow(warp_imgs_all_akaze)
plt.title('Mosaic using kaze image')
```

Out[28]:

```
Text(0.5, 1.0, 'Mosaic using kaze image')
```



In [ ]:

```

```