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)
```

In [2]:

```
import cv2
cv= cv2.xfeatures2d.SIFT_create()
```

In [5]:

```
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 [6]:

```
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 [01:09<00:00,  1.13s/it]
100%|██████████| 39/39 [00:43<00:00,  1.13s/it]
```

In [7]:

```
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 [00:26<00:00,  2.32it/s]
100%|██████████| 39/39 [00:16<00:00,  2.42it/s]
```

In [ ]:

```
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]))
```

In [ ]:

```
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]))
```

In [ ]:

```
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]))
```

In [8]:

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

In [ ]:

```
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]))
```

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 [10]:

```
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 [00:51<00:00,  1.18it/s]
100%|██████████| 39/39 [00:32<00:00,  1.20it/s]
```

In [27]:

```
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]))
```

```
100%|██████████| 61/61 [05:00<00:00,  4.93s/it]
100%|██████████| 39/39 [03:21<00:00,  5.16s/it]
```

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 [9]:

```
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]))
```

```
100%|██████████| 61/61 [07:55<00:00,  7.79s/it]
100%|██████████| 39/39 [05:20<00:00,  8.21s/it]
```

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 [10]:

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

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

In [11]:

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

In [12]:

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

In [13]:

```
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 [14]:

```
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 [ ]:

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

In [ ]:

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

In [15]:

```
num_kps_fast = []
for j in tqdm(keypoints_all_left_fast + keypoints_all_right_fast):
    num_kps_fast.append(len(j))
```

```
100%|██████████| 100/100 [00:00<00:00, 451972.41it/s]
```

In [ ]:

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

In [ ]:

```
num_kps_akaze  = []



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

In [16]:

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

```
100%|██████████| 100/100 [00:00<00:00, 221335.30it/s]
```

In [33]:

```
num_kps_mser =[]
for j in tqdm(keypoints_all_left_mser + keypoints_all_right_mser):
    num_kps_mser.append(len(j))
```

```
100%|██████████| 100/100 [00:00<00:00, 393461.91it/s]
```

In [ ]:

```
num_kps_gftt =[]
for j in tqdm(keypoints_all_left_gftt + keypoints_all_left_gftt):
    num_kps_gftt.append(len(j))
```

In [ ]:

```
num_kps_daisy = []
for  j in tqdm(keypoints_all_left_daisy + keypoints_all_right_daisy):
    num_kps_daisy.append(j)
```

In [16]:

```
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 [17]:

```
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 [18]:

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

In [ ]:

```
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)
```

In [ ]:

```
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)
```

In [ ]:

```
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)
```

In [ ]:

```
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)
```

In [20]:

```
H_left_freak = []
H_right_freak = []

num_matches_freak = []
num_good_matches_freak = []

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
_freak[j:j+2][::-1],points_all_left_freak[j:j+2][::-1],descriptors_all_left_freak[j:j+2]
[::-1])
    H_left_freak.append(H_a)
    num_matches_freak.append(matches)
    num_good_matches_freak.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_freak[j:j+2][::-1],points_all_right_freak[j:j+2][::-1],descriptors_all_right_freak[j:
j+2][::-1])
    H_right_freak.append(H_a)
    num_matches_freak.append(matches)
    num_good_matches_freak.append(gd_matches)
```

```
  2%|▏         | 1/61 [00:01<01:43,  1.73s/it]
```

```
Number of matches 23038
Number of matches After Lowe's Ratio 827
Number of Robust matches 178
```

```
  3%|▎         | 2/61 [00:03<01:59,  2.02s/it]
```

```
Number of matches 29091
Number of matches After Lowe's Ratio 896
Number of Robust matches 162
```

```
  5%|▍         | 3/61 [00:06<02:06,  2.17s/it]
```

```
Number of matches 23985
Number of matches After Lowe's Ratio 545
Number of Robust matches 7
```

```
  7%|▋         | 4/61 [00:08<02:00,  2.11s/it]
```

```
Number of matches 21791
Number of matches After Lowe's Ratio 1273
Number of Robust matches 544
```

```
  8%|▊         | 5/61 [00:10<01:51,  1.98s/it]
```

```
Number of matches 26179
Number of matches After Lowe's Ratio 1578
Number of Robust matches 585
```

```
 10%|▉         | 6/61 [00:12<01:50,  2.00s/it]
```

```
Number of matches 24534
Number of matches After Lowe's Ratio 1309
Number of Robust matches 431
```

```
 11%|█▏        | 7/61 [00:14<01:52,  2.09s/it]
```

```
Number of matches 29762
Number of matches After Lowe's Ratio 1770
Number of Robust matches 649
```

```
 13%|█▎        | 8/61 [00:16<01:57,  2.22s/it]
```

```
Number of matches 20799
Number of matches After Lowe's Ratio 987
Number of Robust matches 326
```

```
 15%|█▍        | 9/61 [00:18<01:46,  2.05s/it]
```

```
Number of matches 28885
Number of matches After Lowe's Ratio 1448
Number of Robust matches 726
```

```
 16%|█▋        | 10/61 [00:20<01:49,  2.15s/it]
```

```
Number of matches 24714
Number of matches After Lowe's Ratio 1147
Number of Robust matches 493
```

```
 18%|█▊        | 11/61 [00:23<01:47,  2.16s/it]
```

```
Number of matches 30963
Number of matches After Lowe's Ratio 1992
Number of Robust matches 1010
```

```
 20%|█▉        | 12/61 [00:26<01:57,  2.39s/it]
```

```
Number of matches 30334
Number of matches After Lowe's Ratio 1968
Number of Robust matches 1157
```

```
 21%|██▏       | 13/61 [00:28<01:59,  2.49s/it]
```

```
Number of matches 35225
Number of matches After Lowe's Ratio 2013
Number of Robust matches 1053
```

```
 23%|██▎       | 14/61 [00:32<02:16,  2.90s/it]
```

```
Number of matches 34588
Number of matches After Lowe's Ratio 2639
Number of Robust matches 1809
```

```
 25%|██▍       | 15/61 [00:35<02:15,  2.94s/it]
```

```
Number of matches 32541
Number of matches After Lowe's Ratio 2190
Number of Robust matches 1262
```

```
 26%|██▌       | 16/61 [00:38<02:13,  2.96s/it]
```

```
Number of matches 28721
Number of matches After Lowe's Ratio 2142
Number of Robust matches 1432
```

```
 28%|██▊       | 17/61 [00:41<02:05,  2.85s/it]
```

```
Number of matches 30225
Number of matches After Lowe's Ratio 2178
Number of Robust matches 1314
```

```
 30%|██▉       | 18/61 [00:43<01:58,  2.76s/it]
```

```
Number of matches 30991
Number of matches After Lowe's Ratio 2575
Number of Robust matches 1393
```

```
 31%|███       | 19/61 [00:46<01:54,  2.73s/it]
```

```
Number of matches 29265
Number of matches After Lowe's Ratio 2813
Number of Robust matches 1813
Number of Robust matches 1015
```

```
 33%|███▎      | 20/61 [00:49<01:53,  2.77s/it]
```

```
Number of matches 29356
Number of matches After Lowe's Ratio 2194
Number of Robust matches 1149
```

```
 34%|███▍      | 21/61 [00:51<01:47,  2.70s/it]
```

```
Number of matches 30010
Number of matches After Lowe's Ratio 1721
Number of Robust matches 827
```

```
 36%|███▌      | 22/61 [00:54<01:43,  2.65s/it]
```

```
Number of matches 29570
Number of matches After Lowe's Ratio 1888
Number of Robust matches 830
```

```
 38%|███▊      | 23/61 [00:57<01:40,  2.65s/it]
```

```
Number of matches 30831
Number of matches After Lowe's Ratio 2105
Number of Robust matches 1042
```

```
 39%|███▉      | 24/61 [00:59<01:41,  2.74s/it]
```

```
Number of matches 33305
Number of matches After Lowe's Ratio 1807
Number of Robust matches 781
```

```
 41%|████      | 25/61 [01:03<01:47,  2.98s/it]
```

```
Number of matches 41880
Number of matches After Lowe's Ratio 1812
Number of Robust matches 599
```

```
 43%|████▎     | 26/61 [01:07<01:54,  3.28s/it]
```

```
Number of matches 35904
Number of matches After Lowe's Ratio 1772
Number of Robust matches 496
```

```
 44%|████▍     | 27/61 [01:10<01:52,  3.31s/it]
```

```
Number of matches 30389
Number of matches After Lowe's Ratio 1701
Number of Robust matches 567
```

```
 46%|████▌     | 28/61 [01:13<01:41,  3.09s/it]
```

```
Number of matches 28704
Number of matches After Lowe's Ratio 1352
Number of Robust matches 326
```

```
 48%|████▊     | 29/61 [01:16<01:34,  2.95s/it]
 48%|████▊     | 29/61 [01:16<01:34,  2.95s/it]
```

```
Number of matches 31542
Number of matches After Lowe's Ratio 971
Number of Robust matches 237
```

```
 49%|████▉     | 30/61 [01:18<01:29,  2.89s/it]
```

```
Number of matches 31853
Number of matches After Lowe's Ratio 1294
Number of Robust matches 392
```

```
 51%|█████     | 31/61 [01:21<01:29,  2.97s/it]
```

```
Number of matches 32825
Number of matches After Lowe's Ratio 839
Number of Robust matches 136
```

```
 52%|█████▏    | 32/61 [01:24<01:25,  2.95s/it]
```

```
Number of matches 22605
Number of matches After Lowe's Ratio 522
Number of Robust matches 8
```

```
 54%|█████▍    | 33/61 [01:26<01:12,  2.60s/it]
```

```
Number of matches 23139
Number of matches After Lowe's Ratio 1327
Number of Robust matches 443
```

```
 56%|█████▌    | 34/61 [01:28<01:03,  2.34s/it]
```

```
Number of matches 19638
Number of matches After Lowe's Ratio 1224
Number of Robust matches 511
```

```
 57%|█████▋    | 35/61 [01:29<00:54,  2.10s/it]
```

```
Number of matches 24280
Number of matches After Lowe's Ratio 1272
Number of Robust matches 460
```

```
 59%|█████▉    | 36/61 [01:32<00:54,  2.19s/it]
```

```
Number of matches 29286
Number of matches After Lowe's Ratio 1611
Number of Robust matches 552
```

```
 61%|██████    | 37/61 [01:35<00:59,  2.48s/it]
```

```
Number of matches 41948
Number of matches After Lowe's Ratio 1637
Number of Robust matches 461
```

```
 62%|██████▏   | 38/61 [01:39<01:09,  3.02s/it]
```

```
Number of matches 45777
Number of matches After Lowe's Ratio 2084
Number of Robust matches 511

 64%|██████▍   | 39/61 [01:44<01:17,  3.53s/it]
Number of matches 41498
Number of matches After Lowe's Ratio 1852
Number of Robust matches 564

 66%|██████▌   | 40/61 [01:48<01:15,  3.58s/it]
Number of matches 32765
Number of matches After Lowe's Ratio 1906
Number of Robust matches 627

 67%|██████▋   | 41/61 [01:51<01:08,  3.40s/it]
Number of matches 30811
Number of matches After Lowe's Ratio 2094
Number of Robust matches 961

 69%|██████▉   | 42/61 [01:54<01:01,  3.23s/it]
Number of matches 28921
Number of matches After Lowe's Ratio 2126
Number of Robust matches 928

 70%|███████   | 43/61 [01:56<00:53,  2.98s/it]
Number of matches 28541
Number of matches After Lowe's Ratio 2295
Number of Robust matches 1301

 72%|███████▏  | 44/61 [01:59<00:48,  2.87s/it]
Number of matches 34315
Number of matches After Lowe's Ratio 2266
Number of Robust matches 1099

 74%|███████▍  | 45/61 [02:02<00:46,  2.94s/it]
Number of matches 37159
Number of matches After Lowe's Ratio 2669
Number of Robust matches 1024

 75%|███████▌  | 46/61 [02:05<00:47,  3.15s/it]
Number of matches 35439
Number of matches After Lowe's Ratio 2547
Number of Robust matches 1201

 77%|███████▋  | 47/61 [02:09<00:47,  3.39s/it]
Number of matches 36939
Number of matches After Lowe's Ratio 2591
Number of Robust matches 1167
```

```
 79%|███████▉  | 48/61 [02:12<00:43,  3.35s/it]
```

```
Number of matches 30760
Number of matches After Lowe's Ratio 1671
Number of Robust matches 676
```

```
 80%|████████  | 49/61 [02:15<00:38,  3.21s/it]
```

```
Number of matches 29514
Number of matches After Lowe's Ratio 2857
Number of Robust matches 1650
```

```
 82%|████████▏ | 50/61 [02:18<00:33,  3.01s/it]
```

```
Number of matches 28402
Number of matches After Lowe's Ratio 2460
Number of Robust matches 1500
```

```
 84%|████████▎ | 51/61 [02:20<00:27,  2.80s/it]
```

```
Number of matches 25855
Number of matches After Lowe's Ratio 1467
Number of Robust matches 688
```

```
 85%|████████▌ | 52/61 [02:22<00:23,  2.58s/it]
```

```
Number of matches 25442
Number of matches After Lowe's Ratio 1488
Number of Robust matches 707
```

```
 87%|████████▋ | 53/61 [02:24<00:19,  2.45s/it]
```

```
Number of matches 25540
Number of matches After Lowe's Ratio 1988
Number of Robust matches 1163
```

```
 89%|████████▊ | 54/61 [02:27<00:17,  2.44s/it]
```

```
Number of matches 31109
Number of matches After Lowe's Ratio 1909
Number of Robust matches 859
```

```
 90%|█████████ | 55/61 [02:29<00:14,  2.46s/it]
```

```
Number of matches 25420
Number of matches After Lowe's Ratio 1764
Number of Robust matches 1062
```

```
 92%|█████████▏| 56/61 [02:32<00:11,  2.39s/it]
```

```
Number of matches 27131
Number of matches After Lowe's Ratio 1710
Number of Robust matches 637
```

```
 93%|█████████▎| 57/61 [02:34<00:09,  2.36s/it]
```

```
Number of matches 30777
Number of matches After Lowe's Ratio 2514
Number of Robust matches 752

 95%|█████████▌| 58/61 [02:37<00:07,  2.51s/it]

Number of matches 31215
Number of matches After Lowe's Ratio 1722
Number of Robust matches 512

 97%|█████████▋| 59/61 [02:40<00:05,  2.79s/it]

Number of matches 34016
Number of matches After Lowe's Ratio 2526
Number of Robust matches 717

 98%|█████████▊| 60/61 [02:43<00:02,  2.73s/it]
  0%|          | 0/39 [00:00<?, ?it/s]

Number of matches 23270
Number of matches After Lowe's Ratio 863
Number of Robust matches 145

  3%|▎         | 1/39 [00:01<01:13,  1.94s/it]

Number of matches 33002
Number of matches After Lowe's Ratio 1734
Number of Robust matches 915

  5%|▌         | 2/39 [00:04<01:34,  2.55s/it]

Number of matches 26873
Number of matches After Lowe's Ratio 1827
Number of Robust matches 1125

  8%|▊         | 3/39 [00:07<01:27,  2.44s/it]

Number of matches 24439
Number of matches After Lowe's Ratio 1139
Number of Robust matches 487

 10%|█         | 4/39 [00:09<01:17,  2.21s/it]

Number of matches 22762
Number of matches After Lowe's Ratio 707
Number of Robust matches 184

 13%|█▎        | 5/39 [00:10<01:09,  2.04s/it]

Number of matches 19743
Number of matches After Lowe's Ratio 1447
Number of Robust matches 896

 15%|█▌        | 6/39 [00:12<01:02,  1.88s/it]

Number of matches 29263
Number of matches After Lowe's Ratio 1177
Number of Robust matches 466
```

```
 18%|█▊        | 7/39 [00:15<01:07,  2.12s/it]
```

```
Number of matches 29662
Number of matches After Lowe's Ratio 2324
Number of Robust matches 1498
```

```
 21%|██        | 8/39 [00:17<01:12,  2.34s/it]
```

```
Number of matches 31340
Number of matches After Lowe's Ratio 2381
Number of Robust matches 1700
```

```
 23%|██▎       | 9/39 [00:20<01:12,  2.43s/it]
```

```
Number of matches 28069
Number of matches After Lowe's Ratio 1945
Number of Robust matches 1199
```

```
 26%|██▌       | 10/39 [00:22<01:10,  2.44s/it]
```

```
Number of matches 30957
Number of matches After Lowe's Ratio 2200
Number of Robust matches 1454
```

```
 28%|██▊       | 11/39 [00:25<01:10,  2.51s/it]
```

```
Number of matches 30565
Number of matches After Lowe's Ratio 1722
Number of Robust matches 920
```

```
 31%|███       | 12/39 [00:29<01:15,  2.80s/it]
```

```
Number of matches 32974
Number of matches After Lowe's Ratio 2112
Number of Robust matches 1181
```

```
 33%|███▎      | 13/39 [00:32<01:15,  2.91s/it]
```

```
Number of matches 36549
Number of matches After Lowe's Ratio 2012
Number of Robust matches 1014
```

```
 36%|███▌      | 14/39 [00:35<01:15,  3.03s/it]
```

```
Number of matches 35159
Number of matches After Lowe's Ratio 2162
Number of Robust matches 1012
```

```
 38%|███▊      | 15/39 [00:39<01:16,  3.21s/it]
```

```
Number of matches 38169
Number of matches After Lowe's Ratio 2419
Number of Robust matches 1113
```

```
 41%|████      | 16/39 [00:42<01:14,  3.26s/it]
```

```
Number of matches 33493
Number of matches 33493
Number of matches After Lowe's Ratio 2232
Number of Robust matches 906

 44%|████      | 17/39 [00:45<01:08,  3.13s/it]
Number of matches 27254
Number of matches After Lowe's Ratio 1748
Number of Robust matches 725

 46%|████▌     | 18/39 [00:47<01:00,  2.89s/it]
Number of matches 29921
Number of matches After Lowe's Ratio 2041
Number of Robust matches 716

 49%|████▊     | 19/39 [00:50<00:57,  2.89s/it]
Number of matches 27914
Number of matches After Lowe's Ratio 1863
Number of Robust matches 535

 51%|█████▏    | 20/39 [00:52<00:51,  2.69s/it]
Number of matches 22382
Number of matches After Lowe's Ratio 1349
Number of Robust matches 437

 54%|█████▍    | 21/39 [00:54<00:43,  2.43s/it]
Number of matches 26806
Number of matches After Lowe's Ratio 1383
Number of Robust matches 482

 56%|█████▋    | 22/39 [00:57<00:42,  2.48s/it]
Number of matches 43407
Number of matches After Lowe's Ratio 929
Number of Robust matches 68

 59%|█████▉    | 23/39 [01:02<00:51,  3.23s/it]
Number of matches 39639
Number of matches After Lowe's Ratio 1076
Number of Robust matches 296

 62%|██████▏   | 24/39 [01:06<00:52,  3.49s/it]
Number of matches 44415
Number of matches After Lowe's Ratio 759
Number of Robust matches 6

 64%|██████▍   | 25/39 [01:10<00:51,  3.67s/it]
Number of matches 34575
Number of matches After Lowe's Ratio 999
Number of Robust matches 188
```

```
 67%|██████▋   | 26/39 [01:13<00:45,  3.53s/it]
```

```
Number of matches 32318
Number of matches After Lowe's Ratio 1688
Number of Robust matches 510
```

```
 69%|██████▉   | 27/39 [01:16<00:40,  3.35s/it]
```

```
Number of matches 30468
Number of matches After Lowe's Ratio 1795
Number of Robust matches 498
```

```
 72%|███████▏  | 28/39 [01:19<00:34,  3.10s/it]
```

```
Number of matches 26838
Number of matches After Lowe's Ratio 1523
Number of Robust matches 429
```

```
 74%|███████▍  | 29/39 [01:21<00:28,  2.82s/it]
```

```
Number of matches 24184
Number of matches After Lowe's Ratio 1222
Number of Robust matches 298
```

```
 77%|███████▋  | 30/39 [01:23<00:23,  2.63s/it]
```

```
Number of matches 25941
Number of matches After Lowe's Ratio 1205
Number of Robust matches 311
```

```
 79%|███████▉  | 31/39 [01:25<00:20,  2.54s/it]
```

```
Number of matches 27985
Number of matches After Lowe's Ratio 2133
Number of Robust matches 634
```

```
 82%|████████▏ | 32/39 [01:28<00:17,  2.49s/it]
```

```
Number of matches 29906
Number of matches After Lowe's Ratio 1417
Number of Robust matches 379
```

```
 85%|████████▍ | 33/39 [01:30<00:14,  2.49s/it]
```

```
Number of matches 26647
Number of matches After Lowe's Ratio 1699
Number of Robust matches 652
```

```
 87%|████████▋ | 34/39 [01:33<00:13,  2.72s/it]
```

```
Number of matches 30823
Number of matches After Lowe's Ratio 1837
Number of Robust matches 613
```

```
 90%|████████▉ | 35/39 [01:36<00:10,  2.64s/it]
```

```
Number of matches 23861
Number of matches After Lowe's Ratio 1301
Number of Robust matches 519
```

```
 92%|█████████▏| 36/39 [01:38<00:07,  2.42s/it]
```

```
Number of matches 26390
Number of matches After Lowe's Ratio 1039
Number of Robust matches 460
```

```
 95%|█████████▍| 37/39 [01:40<00:04,  2.33s/it]
```

```
Number of matches 26937
Number of matches After Lowe's Ratio 1315
Number of Robust matches 663
```

```
 97%|█████████▋| 38/39 [01:42<00:02,  2.70s/it]
```

```
Number of matches 26134
Number of matches After Lowe's Ratio 1256
Number of Robust matches 576
```

In [37]:

```python
H_left_mser = []
H_right_mser = []

num_matches_mser = []
num_good_matches_mser = []

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
_mser[j:j+2][::-1],points_all_left_mser[j:j+2][::-1],descriptors_all_left_mser[j:j+2][::
-1])
    H_left_mser.append(H_a)
    num_matches_mser.append(matches)
    num_good_matches_mser.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_mser[j:j+2][::-1],points_all_right_mser[j:j+2][::-1],descriptors_all_right_mser[j:j+2
][::-1])
    H_right_mser.append(H_a)
    num_matches_mser.append(matches)
    num_good_matches_mser.append(gd_matches)
```

```
  2%|▏         | 1/61 [00:00<00:21,  2.82it/s]
```

```
Number of matches 2649
Number of matches After Lowe's Ratio 227
Number of Robust matches 44
```

```
  3%|▎         | 2/61 [00:00<00:20,  2.82it/s]
```

```
Number of matches 3077
Number of matches After Lowe's Ratio 138
Number of Robust matches 34
```

```
  5%|▌         | 3/61 [00:01<00:20,  2.81it/s]
```

```
Number of matches 2731
Number of matches After Lowe's Ratio 23
Number of Robust matches 9
```

```
  7%|▋         | 4/61 [00:01<00:20,  2.79it/s]
```

```
Number of matches 2847
Number of matches After Lowe's Ratio 559
Number of Robust matches 163
```

```
  8%|▊         | 5/61 [00:01<00:20,  2.79it/s]
```

```
Number of matches 2761
Number of matches After Lowe's Ratio 86
Number of Robust matches 25
```

```
 10%|▉         | 6/61 [00:02<00:19,  2.85it/s]
```

```
Number of matches 2570
Number of matches After Lowe's Ratio 283
Number of Robust matches 78
```

```
 11%|█▏        | 7/61 [00:02<00:18,  2.91it/s]
```

```
Number of matches 2642
Number of matches After Lowe's Ratio 134
Number of Robust matches 42
```

```
 13%|█▎        | 8/61 [00:02<00:17,  3.01it/s]
```

```
Number of matches 2122
Number of matches After Lowe's Ratio 260
Number of Robust matches 81
```

```
 15%|█▍        | 9/61 [00:03<00:16,  3.19it/s]
```

```
Number of matches 2614
Number of matches After Lowe's Ratio 232
Number of Robust matches 90
```

```
 18%|█▊        | 11/61 [00:03<00:13,  3.73it/s]
```

```
Number of matches 1521
Number of matches After Lowe's Ratio 112
Number of Robust matches 52
```

```
Number of matches 1844
Number of matches After Lowe's Ratio 334
Number of Robust matches 142
```

```
 21%|██▏       | 13/61 [00:03<00:11,  4.33it/s]
```

```
Number of matches 1636
Number of matches 1050
Number of matches After Lowe's Ratio 49
Number of Robust matches 19



Number of matches 1977
Number of matches After Lowe's Ratio 169
Number of Robust matches 80
```

```
 23%|██▎       | 14/61 [00:04<00:10,  4.41it/s]
```

```
Number of matches 2022
Number of matches After Lowe's Ratio 293
Number of Robust matches 160
```

```
 25%|██▍       | 15/61 [00:04<00:10,  4.37it/s]
```

```
Number of matches 1919
Number of matches After Lowe's Ratio 322
Number of Robust matches 153
```

```
 26%|██▌       | 16/61 [00:04<00:10,  4.38it/s]
```

```
Number of matches 2101
Number of matches After Lowe's Ratio 338
Number of Robust matches 174
```

```
 28%|██▊       | 17/61 [00:04<00:10,  4.25it/s]
```

```
Number of matches 2304
Number of matches After Lowe's Ratio 449
Number of Robust matches 236
```

```
 30%|██▉       | 18/61 [00:05<00:10,  4.09it/s]
```

```
Number of matches 2185
Number of matches After Lowe's Ratio 632
Number of Robust matches 350
```

```
 31%|███       | 19/61 [00:05<00:10,  4.01it/s]
```

```
Number of matches 2208
Number of matches After Lowe's Ratio 653
Number of Robust matches 334
```

```
 33%|███▎      | 20/61 [00:05<00:10,  3.94it/s]
```

```
Number of matches 2367
Number of matches After Lowe's Ratio 480
Number of Robust matches 204
```

```
 34%|███▍      | 21/61 [00:05<00:10,  3.80it/s]
```

```
Number of matches 2393
Number of matches After Lowe's Ratio 262
Number of Robust matches 108
```

```
 36%|███▋      | 22/61 [00:06<00:10,  3.70it/s]
Number of matches 2359
Number of matches After Lowe's Ratio 537
Number of Robust matches 222


 38%|███▊      | 23/61 [00:06<00:10,  3.56it/s]
Number of matches 2461
Number of matches After Lowe's Ratio 304
Number of Robust matches 137


 39%|███▉      | 24/61 [00:06<00:10,  3.55it/s]
Number of matches 2249
Number of matches After Lowe's Ratio 472
Number of Robust matches 186


 41%|████      | 25/61 [00:07<00:09,  3.63it/s]
Number of matches 2372
Number of matches After Lowe's Ratio 7
Number of Robust matches 5


 43%|████▎     | 26/61 [00:07<00:09,  3.64it/s]
Number of matches 2069
Number of matches After Lowe's Ratio 41
Number of Robust matches 16


 44%|████▍     | 27/61 [00:07<00:10,  3.09it/s]
Number of matches 2359
Number of matches After Lowe's Ratio 495
Number of Robust matches 128


 46%|████▌     | 28/61 [00:08<00:11,  2.93it/s]
Number of matches 2766
Number of matches After Lowe's Ratio 53
Number of Robust matches 18


 48%|████▊     | 29/61 [00:08<00:10,  2.92it/s]
Number of matches 3058
Number of matches After Lowe's Ratio 47
Number of Robust matches 13


 49%|████▉     | 30/61 [00:08<00:10,  2.84it/s]
Number of matches 3085
Number of matches After Lowe's Ratio 422
Number of Robust matches 148


 51%|█████     | 31/61 [00:09<00:10,  2.78it/s]
Number of matches 3255
Number of matches After Lowe's Ratio 264
Number of Robust matches 89


 52%|█████▏    | 32/61 [00:09<00:10,  2.71it/s]

Number of matches 2861
Number of matches After Lowe's Ratio 7
Number of Robust matches 6


 54%|█████▍    | 33/61 [00:09<00:10,  2.72it/s]

Number of matches 2662
Number of matches After Lowe's Ratio 463
Number of Robust matches 138


 56%|█████▌    | 34/61 [00:10<00:09,  2.87it/s]

Number of matches 2577
Number of matches After Lowe's Ratio 528
Number of Robust matches 190


 57%|█████▋    | 35/61 [00:10<00:08,  2.93it/s]

Number of matches 2927
Number of matches After Lowe's Ratio 442
Number of Robust matches 143


 59%|█████▉    | 36/61 [00:10<00:08,  2.81it/s]

Number of matches 3331
Number of matches After Lowe's Ratio 537
Number of Robust matches 162


 61%|██████    | 37/61 [00:11<00:09,  2.62it/s]

Number of matches 3499
Number of matches After Lowe's Ratio 497
Number of Robust matches 146


 62%|██████▏   | 38/61 [00:11<00:09,  2.41it/s]

Number of matches 3895
Number of matches After Lowe's Ratio 731
Number of Robust matches 211


 64%|██████▍   | 39/61 [00:12<00:09,  2.23it/s]

Number of matches 3345
Number of matches After Lowe's Ratio 636
Number of Robust matches 203


 66%|██████▌   | 40/61 [00:13<00:10,  2.03it/s]

Number of matches 2826
Number of matches After Lowe's Ratio 658
Number of Robust matches 225
```

```
 67%|██████▋   | 41/61 [00:13<00:08,  2.22it/s]
```

```
Number of matches 2607
Number of matches After Lowe's Ratio 611
Number of Robust matches 260
```

```
 69%|██████▉   | 42/61 [00:13<00:07,  2.45it/s]
```

```
Number of matches 2230
Number of matches After Lowe's Ratio 529
Number of Robust matches 242
```

```
 70%|███████   | 43/61 [00:13<00:06,  2.73it/s]
```

```
Number of matches 2359
Number of matches After Lowe's Ratio 590
Number of Robust matches 273
```

```
 72%|███████▏  | 44/61 [00:14<00:05,  2.93it/s]
```

```
Number of matches 2556
Number of matches After Lowe's Ratio 623
Number of Robust matches 250
```

```
 74%|███████▍  | 45/61 [00:14<00:05,  2.99it/s]
```

```
Number of matches 2370
Number of matches After Lowe's Ratio 625
Number of Robust matches 230
```

```
 75%|███████▌  | 46/61 [00:14<00:04,  3.14it/s]
```

```
Number of matches 2175
Number of matches After Lowe's Ratio 627
Number of Robust matches 238
```

```
 77%|███████▋  | 47/61 [00:15<00:04,  3.37it/s]
```

```
Number of matches 1943
Number of matches After Lowe's Ratio 491
Number of Robust matches 196
```

```
 80%|████████  | 49/61 [00:15<00:02,  4.02it/s]
```

```
Number of matches 1749
Number of matches After Lowe's Ratio 258
Number of Robust matches 137
```

```
Number of matches 1809
Number of matches After Lowe's Ratio 510
Number of Robust matches 254
```

```
 82%|████████▏ | 50/61 [00:15<00:02,  4.20it/s]
```

```
Number of matches 1908
Number of matches After Lowe's Ratio 447
Number of Robust matches 216
```

```
 84%|████████▍ | 51/61 [00:15<00:02,  4.31it/s]
```

```
Number of matches 1778
Number of matches After Lowe's Ratio 371
Number of Robust matches 170
```

```
 85%|████████▌ | 52/61 [00:16<00:02,  4.38it/s]
```

```
Number of matches 2041
Number of matches After Lowe's Ratio 422
Number of Robust matches 162
```

```
 87%|████████▋ | 53/61 [00:16<00:01,  4.19it/s]
```

```
Number of matches 2214
Number of matches After Lowe's Ratio 631
Number of Robust matches 270
```

```
 89%|████████▊ | 54/61 [00:16<00:01,  4.07it/s]
```

```
Number of matches 2180
Number of matches After Lowe's Ratio 311
Number of Robust matches 140
```

```
 90%|█████████ | 55/61 [00:16<00:01,  4.03it/s]
```

```
Number of matches 2265
Number of matches After Lowe's Ratio 284
Number of Robust matches 104
```

```
 92%|█████████▏| 56/61 [00:17<00:01,  3.88it/s]
```

```
Number of matches 2489
Number of matches After Lowe's Ratio 422
Number of Robust matches 131
```

```
 93%|█████████▎| 57/61 [00:17<00:01,  3.66it/s]
```

```
Number of matches 2634
Number of matches After Lowe's Ratio 588
Number of Robust matches 199
```

```
 95%|█████████▌| 58/61 [00:17<00:00,  3.52it/s]
```

```
Number of matches 2578
Number of matches After Lowe's Ratio 334
Number of Robust matches 101
```

```
 97%|█████████▋| 59/61 [00:18<00:00,  3.38it/s]
```

```
Number of matches 2583
Number of matches After Lowe's Ratio 632
Number of Robust matches 205
```

```
 98%|█████████▊| 60/61 [00:18<00:00,  3.24it/s]
```

```
  0%|          | 0/39 [00:00<?, ?it/s]
```

```
Number of matches 2358
Number of matches After Lowe's Ratio 81
Number of Robust matches 29
```

```
  3%|▎         | 1/39 [00:00<00:17,  2.12it/s]
```

```
Number of matches 2578
Number of matches After Lowe's Ratio 340
Number of Robust matches 125
```

```
  5%|▌         | 2/39 [00:00<00:13,  2.70it/s]
```

```
Number of matches 1782
Number of matches After Lowe's Ratio 391
Number of Robust matches 171
```

```
Number of matches 1452
Number of matches After Lowe's Ratio 176
```

```
 10%|█         | 4/39 [00:01<00:08,  3.99it/s]
```

```
Number of Robust matches 99
```

```
Number of matches 2306
Number of matches After Lowe's Ratio 136
Number of Robust matches 67
```

```
 13%|█▎        | 5/39 [00:01<00:08,  4.00it/s]
```

```
Number of matches 1911
Number of matches After Lowe's Ratio 466
Number of Robust matches 268
```

```
 15%|█▌        | 6/39 [00:01<00:08,  4.09it/s]
```

```
Number of matches 2142
Number of matches After Lowe's Ratio 263
Number of Robust matches 134
```

```
 18%|█▊        | 7/39 [00:01<00:07,  4.10it/s]
```

```
Number of matches 2191
Number of matches After Lowe's Ratio 613
Number of Robust matches 343
```

```
 21%|██        | 8/39 [00:02<00:07,  4.03it/s]
```

```
Number of matches 2029
Number of matches After Lowe's Ratio 538
Number of Robust matches 289
```

```
 23%|██▎       | 9/39 [00:02<00:07,  4.07it/s]
```

```
Number of matches 2512
Number of matches After Lowe's Ratio 588
Number of Robust matches 333
```

```
 26%|██▌       | 10/39 [00:02<00:07,  3.86it/s]
```

```
Number of matches 2412
Number of matches After Lowe's Ratio 670
Number of Robust matches 367
```

```
 28%|██▊       | 11/39 [00:02<00:07,  3.64it/s]
```

```
Number of matches 2826
Number of matches After Lowe's Ratio 368
Number of Robust matches 158
```

```
 31%|███       | 12/39 [00:03<00:07,  3.43it/s]
```

```
Number of matches 2981
Number of matches After Lowe's Ratio 571
Number of Robust matches 224
```

```
 33%|███▎      | 13/39 [00:03<00:08,  3.21it/s]
```

```
Number of matches 3041
Number of matches After Lowe's Ratio 596
Number of Robust matches 234
```

```
 36%|███▌      | 14/39 [00:04<00:08,  3.05it/s]
```

```
Number of matches 3227
Number of matches After Lowe's Ratio 686
Number of Robust matches 271
```

```
 38%|███▊      | 15/39 [00:04<00:08,  2.93it/s]
```

```
Number of matches 3127
Number of matches After Lowe's Ratio 517
Number of Robust matches 214
```

```
 41%|████      | 16/39 [00:04<00:08,  2.84it/s]
```

```
Number of matches 3082
Number of matches After Lowe's Ratio 764
Number of Robust matches 277
```

```
 44%|████▎     | 17/39 [00:05<00:07,  2.78it/s]
```

```
Number of matches 2828
Number of matches After Lowe's Ratio 573
Number of Robust matches 177
```

```
 46%|████▌     | 18/39 [00:05<00:07,  2.82it/s]
```

```
Number of matches 2804
Number of matches After Lowe's Ratio 739
Number of Robust matches 207
```

```
 49%|████▉     | 19/39 [00:05<00:07,  2.79it/s]
```

```
Number of matches 3047
Number of matches After Lowe's Ratio 701
Number of Robust matches 189
```

```
 51%|█████▏    | 20/39 [00:06<00:06,  2.82it/s]
```

```
Number of matches 3292
Number of matches After Lowe's Ratio 391
Number of Robust matches 141
```

```
 54%|█████▍    | 21/39 [00:06<00:06,  2.80it/s]
```

```
Number of matches 2865
Number of matches After Lowe's Ratio 325
Number of Robust matches 122
```

```
 56%|█████▋    | 22/39 [00:06<00:06,  2.82it/s]
```

```
Number of matches 3522
Number of matches After Lowe's Ratio 133
Number of Robust matches 39
```

```
 59%|█████▉    | 23/39 [00:07<00:06,  2.62it/s]
```

```
Number of matches 3409
Number of matches After Lowe's Ratio 528
Number of Robust matches 140
```

```
 62%|██████▏   | 24/39 [00:07<00:06,  2.50it/s]
```

```
Number of matches 3648
Number of matches After Lowe's Ratio 16
Number of Robust matches 4
```

```
 64%|██████▍   | 25/39 [00:08<00:05,  2.40it/s]
```

```
Number of matches 3290
Number of matches After Lowe's Ratio 371
Number of Robust matches 110
```

```
 67%|██████▋   | 26/39 [00:08<00:05,  2.46it/s]
```

```
Number of matches 3162
Number of matches After Lowe's Ratio 415
Number of Robust matches 129
```

```
 69%|██████▉   | 27/39 [00:09<00:04,  2.56it/s]
```

```
Number of matches 2667
Number of matches After Lowe's Ratio 582
Number of Robust matches 198
```

```
 72%|███████▏  | 28/39 [00:09<00:04,  2.68it/s]
```

```
Number of matches 3045
Number of matches After Lowe's Ratio 329
Number of Robust matches 99
```

```
 74%|███████▍  | 29/39 [00:09<00:03,  2.71it/s]
```

```
Number of matches 2955
Number of matches After Lowe's Ratio 362
Number of Robust matches 105
```

```
 77%|███████▋  | 30/39 [00:10<00:03,  2.34it/s]
```

```
Number of matches 2928
Number of matches After Lowe's Ratio 505
Number of Robust matches 194
```

```
 79%|███████▉  | 31/39 [00:10<00:03,  2.10it/s]
```

```
Number of matches 2889
Number of matches After Lowe's Ratio 841
Number of Robust matches 263
```

```
 82%|████████▏ | 32/39 [00:11<00:03,  1.86it/s]
```

```
Number of matches 2877
Number of matches After Lowe's Ratio 528
Number of Robust matches 193
```

```
 85%|████████▌ | 33/39 [00:11<00:02,  2.08it/s]
```

```
Number of matches 2724
Number of matches After Lowe's Ratio 68
Number of Robust matches 23
```

```
 87%|████████▋ | 34/39 [00:12<00:02,  2.33it/s]
```

```
Number of matches 2688
Number of matches After Lowe's Ratio 450
Number of Robust matches 173
```

```
 90%|████████▉ | 35/39 [00:12<00:01,  2.54it/s]
```

```
Number of matches 2695
Number of matches After Lowe's Ratio 306
Number of Robust matches 105
```

```
 92%|█████████▏| 36/39 [00:12<00:01,  2.75it/s]
```

```
Number of matches 2283
Number of matches After Lowe's Ratio 162
Number of Robust matches 84
```

```
 95%|█████████▍| 37/39 [00:13<00:00,  2.99it/s]
```

```
Number of matches 2330
Number of matches After Lowe's Ratio 343
Number of Robust matches 167
```

```
 97%|█████████▋| 38/39 [00:13<00:00,  2.85it/s]
```

```
Number of matches 2095
Number of matches After Lowe's Ratio 346
Number of Robust matches 136
```

In [ ]:

```
H_left_gftt = []
H_right_gftt = []

num_matches_gftt = []
num_good_matches_gftt = []

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
_gftt[j:j+2][::-1],points_all_left_gftt[j:j+2][::-1],descriptors_all_left_gftt[j:j+2][::
-1])
    H_left_gftt.append(H_a)
    num_matches_gftt.append(matches)
    num_good_matches_gftt.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_gftt[j:j+2][::-1],points_all_right_gftt[j:j+2][::-1],descriptors_all_right_gftt[j:j+2
][::-1])
    H_right_gftt.append(H_a)
    num_matches_gftt.append(matches)
    num_good_matches_gftt.append(gd_matches)
```

In [ ]:

```
H_left_daisy = []
H_right_daisy = []

num_matches_daisy = []
num_good_matches_daisy = []

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
_daisy[j:j+2][::-1],points_all_left_daisy[j:j+2][::-1],descriptors_all_left_daisy[j:j+2]
[::-1])
    H_left_daisy.append(H_a)
    num_matches_daisy.append(matches)
    num_good_matches_daisy.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_daisy[j:j+2][::-1],points_all_right_daisy[j:j+2][::-1],descriptors_all_right_daisy[j:
j+2][::-1])
    H_right_daisy.append(H_a)
    num_matches_daisy.append(matches)
    num_good_matches_daisy.append(gd_matches)
```

In [19]:

```
H_left_fast = []
```

```
H_right_fast = []

num_matches_fast = []
num_good_matches_fast = []

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
_fast[j:j+2][::-1],points_all_left_fast[j:j+2][::-1],descriptors_all_left_fast[j:j+2][::
-1])
    H_left_fast.append(H_a)
    num_matches_fast.append(matches)
    num_good_matches_fast.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_fast[j:j+2][::-1],points_all_right_fast[j:j+2][::-1],descriptors_all_right_fast[j:j+2
][::-1])
    H_right_fast.append(H_a)
    num_matches_fast.append(matches)
    num_good_matches_fast.append(gd_matches)
```

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

Number of matches 109090
Number of matches After Lowe's Ratio 4766

  2%|▏         | 1/61 [00:22<22:35, 22.59s/it]

Number of Robust matches 1912

  3%|▎         | 2/61 [00:45<22:22, 22.75s/it]

Number of matches 121549
Number of matches After Lowe's Ratio 1831
Number of Robust matches 712

  5%|▍         | 3/61 [01:09<22:43, 23.50s/it]

Number of matches 106987
Number of matches After Lowe's Ratio 137
Number of Robust matches 45

  7%|▋         | 4/61 [01:32<22:01, 23.18s/it]

Number of matches 107417
Number of matches After Lowe's Ratio 15270
Number of Robust matches 8119

  8%|▊         | 5/61 [01:55<21:28, 23.02s/it]

Number of matches 109972
Number of matches After Lowe's Ratio 1191
Number of Robust matches 395

 10%|▉         | 6/61 [02:17<20:51, 22.75s/it]

Number of matches 105684
Number of matches After Lowe's Ratio 4632
Number of Robust matches 2291
```

```
 11%|█         | 7/61 [02:40<20:29, 22.77s/it]
```

```
Number of matches 113643
Number of matches After Lowe's Ratio 2028
Number of Robust matches 963
```

```
 13%|█▎        | 8/61 [03:02<20:01, 22.67s/it]
```

```
Number of matches 85258
Number of matches After Lowe's Ratio 6971
Number of Robust matches 4022
```

```
 15%|█▍        | 9/61 [03:21<18:34, 21.42s/it]
```

```
Number of matches 104129
Number of matches After Lowe's Ratio 6386
Number of Robust matches 3492
```

```
 16%|█▋        | 10/61 [03:41<17:50, 21.00s/it]
```

```
Number of matches 63147
Number of matches After Lowe's Ratio 2650
Number of Robust matches 1762
```

```
 18%|█▊        | 11/61 [03:55<15:46, 18.94s/it]
```

```
Number of matches 87536
Number of matches After Lowe's Ratio 10361
Number of Robust matches 7029
```

```
 20%|█▉        | 12/61 [04:13<15:15, 18.69s/it]
```

```
Number of matches 75857
Number of matches After Lowe's Ratio 605
Number of Robust matches 315
```

```
 21%|██▏       | 13/61 [04:30<14:23, 18.00s/it]
```

```
Number of matches 97150
Number of matches After Lowe's Ratio 3374
Number of Robust matches 2121
```

```
 23%|██▎       | 14/61 [04:50<14:37, 18.67s/it]
```

```
Number of matches 98403
Number of matches After Lowe's Ratio 7946
Number of Robust matches 5397
```

```
 25%|██▍       | 15/61 [05:10<14:42, 19.19s/it]
```

```
Number of matches 94577
Number of matches After Lowe's Ratio 10228
Number of Robust matches 7233
```

```
 26%|██▌       | 16/61 [05:30<14:23, 19.19s/it]
Number of matches 94935
Number of matches After Lowe's Ratio 7277
Number of Robust matches 4884


 28%|██▊       | 17/61 [05:50<14:19, 19.54s/it]
Number of matches 101108
Number of matches After Lowe's Ratio 12159
Number of Robust matches 9516


Number of matches 103611
Number of matches After Lowe's Ratio 25781
 30%|██▉       | 18/61 [06:12<14:29, 20.22s/it]
Number of Robust matches 21015


Number of matches 106411
Number of matches After Lowe's Ratio 22920
Number of Robust matches 19207


 31%|███       | 19/61 [06:34<14:34, 20.83s/it]
Number of matches 115231
Number of matches After Lowe's Ratio 14043
 33%|███▎      | 20/61 [06:57<14:40, 21.47s/it]
Number of Robust matches 10428


 34%|███▍      | 21/61 [07:21<14:47, 22.19s/it]
Number of matches 117947
Number of matches After Lowe's Ratio 4341
Number of Robust matches 2468


Number of matches 111829
Number of matches After Lowe's Ratio 24979
 38%|███▊      | 23/61 [08:09<14:36, 23.07s/it]
Number of Robust matches 16768


Number of matches 112111
Number of matches After Lowe's Ratio 4906
Number of Robust matches 2160


 39%|███▉      | 24/61 [08:33<14:22, 23.31s/it]
Number of matches 116212
Number of matches After Lowe's Ratio 19093
Number of Robust matches 12753


 41%|████      | 25/61 [08:57<14:05, 23.50s/it]
```

```
Number of matches 123205
Number of matches After Lowe's Ratio 100
Number of Robust matches 14


 43%|████▎     | 26/61 [09:22<13:58, 23.96s/it]

Number of matches 109372
Number of matches After Lowe's Ratio 535
Number of Robust matches 118


Number of matches 112914
Number of matches After Lowe's Ratio 12079

 44%|████▍     | 27/61 [09:45<13:26, 23.73s/it]

Number of Robust matches 6852


 46%|████▌     | 28/61 [10:08<12:56, 23.54s/it]

Number of matches 109913
Number of matches After Lowe's Ratio 378
Number of Robust matches 122


 48%|████▊     | 29/61 [10:31<12:27, 23.35s/it]

Number of matches 120423
Number of matches After Lowe's Ratio 233
Number of Robust matches 61


 49%|████▉     | 30/61 [10:56<12:18, 23.83s/it]

Number of matches 121925
Number of matches After Lowe's Ratio 6840
Number of Robust matches 2979


 51%|█████     | 31/61 [11:20<12:01, 24.04s/it]

Number of matches 116705
Number of matches After Lowe's Ratio 4524
Number of Robust matches 1852


 52%|█████▏    | 32/61 [11:44<11:34, 23.94s/it]

Number of matches 109716
Number of matches After Lowe's Ratio 50
Number of Robust matches 8


Number of matches 110779
Number of matches After Lowe's Ratio 20487

 54%|█████▍    | 33/61 [12:07<10:58, 23.51s/it]

Number of Robust matches 12743


 56%|█████▌    | 34/61 [12:29<10:28, 23.28s/it]

Number of matches 109349
Number of matches After Lowe's Ratio 19428
Number of Robust matches 9815
```

```
 57%|█████▋    | 35/61 [12:52<10:02, 23.18s/it]
```

```
Number of matches 115938
Number of matches After Lowe's Ratio 17149
Number of Robust matches 11331
```

```
 59%|█████▉    | 36/61 [13:16<09:43, 23.34s/it]
```

```
Number of matches 122855
Number of matches After Lowe's Ratio 14610
Number of Robust matches 7904
```

```
 61%|██████    | 37/61 [13:42<09:35, 24.00s/it]
```

```
Number of matches 133153
Number of matches After Lowe's Ratio 19119
Number of Robust matches 9940


Number of matches 139272
Number of matches After Lowe's Ratio 22039
Number of Robust matches 9809
```

```
 64%|██████▍   | 39/61 [14:37<09:32, 26.02s/it]
```

```
Number of matches 132310
Number of matches After Lowe's Ratio 20319
Number of Robust matches 10387
```

```
 66%|██████▌   | 40/61 [15:04<09:09, 26.18s/it]
```

```
Number of matches 117037
Number of matches After Lowe's Ratio 22242
Number of Robust matches 13359


Number of matches 112444
Number of matches After Lowe's Ratio 25726
```

```
 67%|██████▋   | 41/61 [15:28<08:28, 25.44s/it]
```

```
Number of Robust matches 16939


Number of matches 105817
Number of matches After Lowe's Ratio 25639
Number of Robust matches 19889
```

```
 70%|███████   | 43/61 [16:13<07:11, 23.97s/it]
```

```
Number of matches 103385
Number of matches After Lowe's Ratio 22610
Number of Robust matches 15518
```

```
 72%|███████▏  | 44/61 [16:35<06:35, 23.27s/it]
```

```
Number of matches 109002
Number of matches After Lowe's Ratio 21739
Number of Robust matches 16322


Number of matches 112071
Number of matches After Lowe's Ratio 23284
Number of Robust matches 14378


 74%|███████▍  | 45/61 [16:58<06:14, 23.41s/it]

Number of matches 106802
Number of matches After Lowe's Ratio 24957
Number of Robust matches 18034


 75%|███████▌  | 46/61 [17:22<05:52, 23.50s/it]

Number of matches 101920
Number of matches After Lowe's Ratio 18097

 77%|███████▋  | 47/61 [17:44<05:24, 23.19s/it]

Number of Robust matches 10849


 79%|███████▊  | 48/61 [18:05<04:51, 22.44s/it]

Number of matches 85971
Number of matches After Lowe's Ratio 10283
Number of Robust matches 6982


 80%|████████  | 49/61 [18:24<04:14, 21.23s/it]

Number of matches 81836
Number of matches After Lowe's Ratio 23625
Number of Robust matches 19842


 82%|████████▏ | 50/61 [18:40<03:39, 19.93s/it]

Number of matches 87852
Number of matches After Lowe's Ratio 18370
Number of Robust matches 12411


Number of matches 90007
Number of matches After Lowe's Ratio 18016

 84%|████████▎ | 51/61 [18:59<03:16, 19.65s/it]

Number of Robust matches 11149


 85%|████████▌ | 52/61 [19:18<02:52, 19.22s/it]

Number of matches 89119
Number of matches After Lowe's Ratio 17660
Number of Robust matches 11873


Number of matches 93962
Number of matches After Lowe's Ratio 25943

 87%|████████▋ | 53/61 [19:37<02:33, 19.19s/it]
```

```
Number of Robust matches 17703
```

```
 89%|████████▉ | 54/61 [19:55<02:12, 18.94s/it]
```

```
Number of matches 90558
Number of matches After Lowe's Ratio 7685
Number of Robust matches 4960
```

```
 90%|█████████ | 55/61 [20:14<01:53, 18.90s/it]
```

```
Number of matches 89788
Number of matches After Lowe's Ratio 5182
Number of Robust matches 3268
```

```
Number of matches 94690
Number of matches After Lowe's Ratio 11577
```

```
 92%|█████████▏| 56/61 [20:34<01:35, 19.13s/it]
```

```
Number of Robust matches 6229
```

```
Number of matches 98372
Number of matches After Lowe's Ratio 10379
```

```
 93%|█████████▎| 57/61 [20:53<01:17, 19.31s/it]
```

```
Number of Robust matches 4528
```

```
Number of matches 97482
Number of matches After Lowe's Ratio 5891
```

```
 95%|█████████▌| 58/61 [21:14<00:59, 19.68s/it]
```

```
Number of Robust matches 2276
```

```
Number of matches 100849
Number of matches After Lowe's Ratio 12415
```

```
 97%|█████████▋| 59/61 [21:34<00:39, 19.86s/it]
```

```
Number of Robust matches 5513
```

```
 98%|█████████▊| 60/61 [21:55<00:21, 21.92s/it]
  0%|          | 0/39 [00:00<?, ?it/s]
```

```
Number of matches 92828
Number of matches After Lowe's Ratio 1142
Number of Robust matches 408
```

```
  3%|▎         | 1/39 [00:23<14:38, 23.12s/it]
```

```
Number of matches 123694
Number of matches After Lowe's Ratio 19451
Number of Robust matches 14774
```

```
  5%|▌         | 2/39 [00:46<14:24, 23.37s/it]
```

```
Number of matches 96343
Number of matches After Lowe's Ratio 16757
Number of Robust matches 12888
```

```
  8%|▊         | 3/39 [01:04<12:29, 20.83s/it]
```

```
Number of matches 54457
Number of matches After Lowe's Ratio 6040
Number of Robust matches 4527
```

```
 10%|█         | 4/39 [01:16<10:08, 17.38s/it]
```

```
Number of matches 74343
Number of matches After Lowe's Ratio 3586
Number of Robust matches 1896
```

```
 13%|█▎        | 5/39 [01:31<09:18, 16.44s/it]
```

```
Number of matches 57064
Number of matches After Lowe's Ratio 10147
Number of Robust matches 8407
```

```
 15%|█▌        | 6/39 [01:45<08:36, 15.65s/it]
```

```
Number of matches 104262
Number of matches After Lowe's Ratio 8765
Number of Robust matches 5257
```

```
 18%|█▊        | 7/39 [02:07<09:26, 17.69s/it]
```

```
Number of matches 105631
Number of matches After Lowe's Ratio 19981
Number of Robust matches 13912
```

```
 21%|██        | 8/39 [02:29<09:50, 19.06s/it]
```

```
Number of matches 108249
Number of matches After Lowe's Ratio 19885
Number of Robust matches 16663
```

```
Number of matches 106606
Number of matches After Lowe's Ratio 23774
```

```
 23%|██▎       | 9/39 [02:51<10:05, 20.18s/it]
```

```
Number of Robust matches 19694
```

```
Number of matches 120200
Number of matches After Lowe's Ratio 33378
```

```
 26%|██▌       | 10/39 [03:15<10:16, 21.27s/it]
```

```
Number of Robust matches 22311
```

```
 28%|██▊       | 11/39 [03:41<10:34, 22.68s/it]
```

```
Number of matches 125528
Number of matches After Lowe's Ratio 9524
Number of Robust matches 5956
```

```
 31%|███       | 12/39 [04:07<10:42, 23.79s/it]
```

```
Number of matches 129552
Number of matches After Lowe's Ratio 16145
Number of Robust matches 10874
```

```
 33%|███▎      | 13/39 [04:34<10:38, 24.55s/it]
```

```
Number of matches 131203
Number of matches After Lowe's Ratio 18740
Number of Robust matches 11352
```

```
 36%|███▌      | 14/39 [05:00<10:27, 25.08s/it]
```

```
Number of matches 128349
Number of matches After Lowe's Ratio 14225
Number of Robust matches 7647
```

```
 38%|███▊      | 15/39 [05:26<10:10, 25.43s/it]
```

```
Number of matches 125112
Number of matches After Lowe's Ratio 9367
Number of Robust matches 4730
```

```
 41%|████      | 16/39 [05:52<09:43, 25.38s/it]
```

```
Number of matches 115397
Number of matches After Lowe's Ratio 18511
Number of Robust matches 9079
```

```
 44%|████▎     | 17/39 [06:15<09:08, 24.92s/it]
```

```
Number of matches 107757
Number of matches After Lowe's Ratio 14445
Number of Robust matches 7262
```

```
 46%|████▌     | 18/39 [06:38<08:27, 24.16s/it]
```

```
Number of matches 108520
Number of matches After Lowe's Ratio 21486
Number of Robust matches 10914
```

```
 49%|████▊     | 19/39 [07:00<07:53, 23.65s/it]
```

```
Number of matches 105170
Number of matches After Lowe's Ratio 21720
Number of Robust matches 10387
```

```
 51%|█████▏    | 20/39 [07:22<07:17, 23.02s/it]
```

```
Number of matches 101842
Number of matches After Lowe's Ratio 6594
Number of Robust matches 3393
```

```
 54%|█████▍    | 21/39 [07:43<06:43, 22.44s/it]
```

```
Number of matches 114806
Number of matches After Lowe's Ratio 7818
Number of Robust matches 3636
```

```
 56%|█████▋    | 22/39 [08:07<06:32, 23.10s/it]
```

```
Number of matches 144158
Number of matches After Lowe's Ratio 3123
Number of Robust matches 1248
```

```
 59%|█████▉    | 23/39 [08:36<06:37, 24.85s/it]
```

```
Number of matches 129327
Number of matches After Lowe's Ratio 13154
Number of Robust matches 5324
```

```
 62%|██████▏   | 24/39 [09:04<06:23, 25.56s/it]
```

```
Number of matches 150305
Number of matches After Lowe's Ratio 40
Number of Robust matches 8
```

```
 64%|██████▍   | 25/39 [09:33<06:12, 26.62s/it]
```

```
Number of matches 125780
Number of matches After Lowe's Ratio 10542
Number of Robust matches 3808
```

```
 67%|██████▋   | 26/39 [09:58<05:40, 26.15s/it]
```

```
Number of matches 122865
Number of matches After Lowe's Ratio 6979
Number of Robust matches 2468
```

```
 69%|██████▉   | 27/39 [10:22<05:07, 25.66s/it]
```

```
Number of matches 105783
Number of matches After Lowe's Ratio 17449
Number of Robust matches 8032
```

```
 72%|███████▏  | 28/39 [10:44<04:28, 24.44s/it]
```

```
Number of matches 102138
Number of matches After Lowe's Ratio 5671
Number of Robust matches 2967
```

```
 74%|███████▍  | 29/39 [11:04<03:50, 23.07s/it]
```

```
Number of matches 89671
Number of matches After Lowe's Ratio 10570
Number of Robust matches 3917
```

```
 77%|███████▋  | 30/39 [11:23<03:16, 21.87s/it]
```

```
Number of matches 96125
Number of matches 90125
Number of matches After Lowe's Ratio 15973
Number of Robust matches 6065



Number of matches 100177
Number of matches After Lowe's Ratio 26527
```

```
 79%|███████▉  | 31/39 [11:43<02:52, 21.50s/it]
```

```
Number of Robust matches 11075
```

```
 82%|████████▏ | 32/39 [12:04<02:28, 21.27s/it]
```

```
Number of matches 107031
Number of matches After Lowe's Ratio 13083
Number of Robust matches 5719
```

```
 85%|████████▍ | 33/39 [12:27<02:09, 21.58s/it]
```

```
Number of matches 103108
Number of matches After Lowe's Ratio 547
Number of Robust matches 169
```

```
 87%|████████▋ | 34/39 [12:48<01:48, 21.62s/it]
```

```
Number of matches 107375
Number of matches After Lowe's Ratio 10283
Number of Robust matches 6356
```

```
 90%|████████▉ | 35/39 [13:10<01:26, 21.69s/it]
```

```
Number of matches 108739
Number of matches After Lowe's Ratio 7884
Number of Robust matches 4117
```

```
 92%|█████████▏| 36/39 [13:33<01:05, 21.99s/it]
```

```
Number of matches 116973
Number of matches After Lowe's Ratio 5091
Number of Robust matches 2771
```

```
 95%|█████████▍| 37/39 [13:57<00:45, 22.70s/it]
```

```
Number of matches 116549
Number of matches After Lowe's Ratio 10501
Number of Robust matches 6900
```

```
 97%|█████████▋| 38/39 [14:21<00:22, 22.67s/it]
```

```
Number of matches 107771
Number of matches After Lowe's Ratio 7950
Number of Robust matches 5517
```

In [20]:

```
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 [21]:

```
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_freak,H_right_freak)
```

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

In [24]:

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

```
step31:Done
```

In [25]:

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

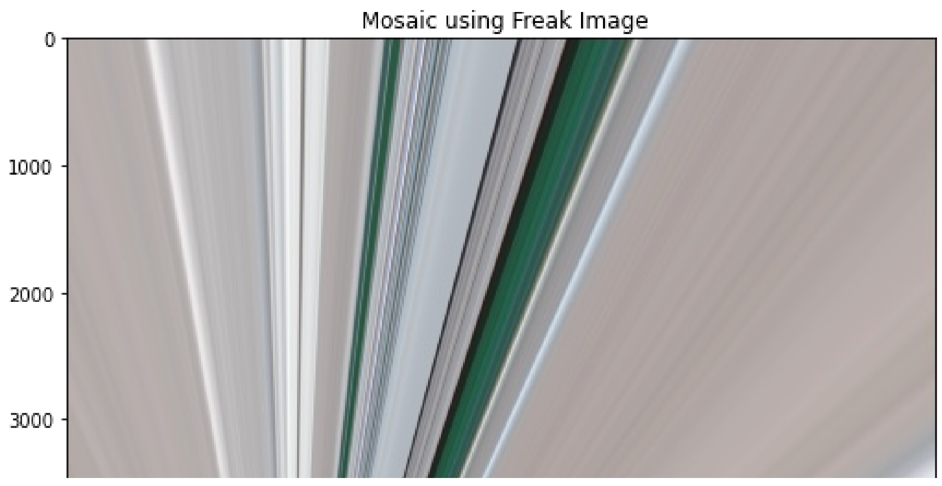
```
step32:Done
```

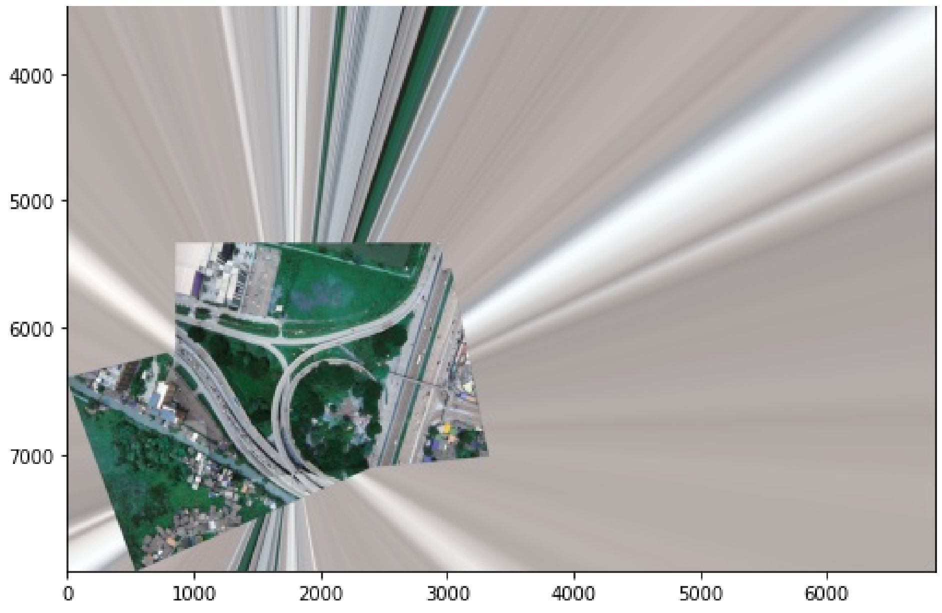
In [26]:

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

Out[26]:

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





In [22]:

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

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

In [23]:

```
warp_img_left = final_steps_left_union(images_left_bgr_no_enhance,H_left_fast,omax,omin,u
max,umin,T,H,W,HT)
```

```
step31:Done
```

In [24]:

```
warp_imgs_all_fast = final_step_right_union(warp_img_left,images_right_bgr_no_enhance,H_r
ight_fast,omax,omin,umax,umin,T,H,W,HT)
```

```
step32:Done
```

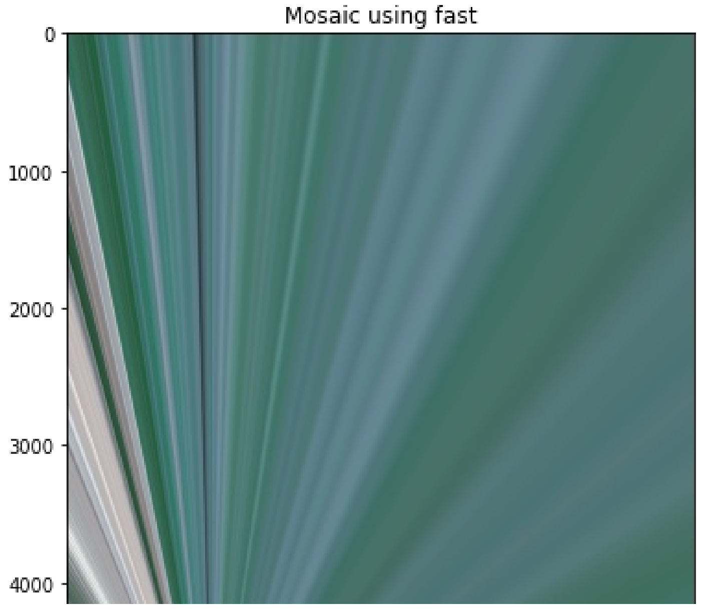
In [25]:

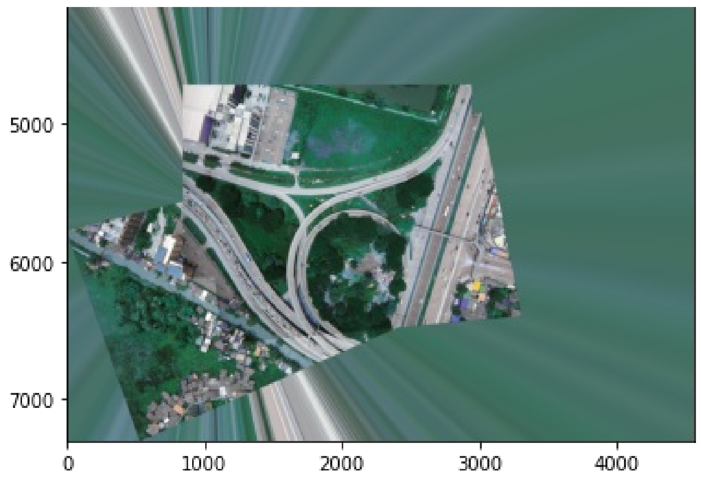
```
plt.figure(figsize=(20,10))
plt.imshow(warp_imgs_all_fast)
plt.title('Mosaic using fast')
```

Out[25]:

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

Mosaic using fast





In [ ]:

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

In [ ]:

```
warp_image_left = final_steps_left_union(images_left_bgr_no_enhance,H_left_daisy,amax,ami
n,zmax,zmin,d,i,q,ht)
```

In [ ]:

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

In [ ]:

```
plt.figure(figsize=(20,10))
plt.imshow(warp_imgs_all_daisy)
plt.title('Mosaic using Daisy image')
plt.imsave('Mosaic using Daisy Image.jpg',warp_imgs_all_daisy)
```

In [ ]:

```

```