In [1]:
```
import numpy as np
import cv2
import scipy.io
import os
from numpy.linalg import norm
from matplotlib import pyplot as plt
from numpy.linalg import det
from numpy.linalg import inv
from scipy.linalg import rq
from numpy.linalg import svd
import matplotlib.pyplot as plt
import numpy as np
import math
import random
import sys
from scipy import ndimage, spatial
from tqdm.notebook import tqdm, trange

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 numpy as np
import math
import glob
import matplotlib.pyplot as plt
import time
import os
import copy
import sklearn.svm
import cv2
from matplotlib import pyplot as plt
import numpy as np
from os.path import exists
import pandas as pd
import PIL
import random
from google.colab import drive
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
import h5py as h5
```

In [2]:
```
from google.colab import drive

# This will prompt for authorization.
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

In [3]:
```
!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 /usr/local/lib/python3.7/dist-packages (3.4.2.17)
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python==3.4.2.17) (1.19.5)
Requirement already satisfied: opencv-contrib-python==3.4.2.17 in /usr/local/lib/python3.7/dist-packages (3.4.2.17)
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-contrib-python==3.4.2.17) (1.19.5)
```

In [4]:
```
class Image:
    def __init__(self, img, position):

        self.img = img
        self.position = position

inlier_matchset = []
def features_matching(a,keypointlength,threshold):
  #threshold=0.2
  bestmatch=np.empty((keypointlength),dtype= np.int16)
  img1index=np.empty((keypointlength),dtype=np.int16)
  distance=np.empty((keypointlength))
  index=0
  for j in range(0,keypointlength):
    #For a descriptor fa in Ia, take the two closest descriptors fb1 and fb2 in Ib
    x=a[j]
    listx=x.tolist()
    x.sort()
    minval1=x[0]                                # min
    minval2=x[1]                                # 2nd min
    itemindex1 = listx.index(minval1)           #index of min val
    itemindex2 = listx.index(minval2)           #index of second min value
    ratio=minval1/minval2                       #Ratio Test

    if ratio<threshold:
      #Low distance ratio: fb1 can be a good match
      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_Homography(im1_pts,im2_pts):
  """
  im1_pts and im2_pts are 2×n matrices with
  4 point correspondences from the two images
  """
  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] # First row
    row2 = [0, 0, 0, a_x, a_y, 1, -b_y*a_x, -b_y*a_y, -b_y] # Second row

    # place the rows in the matrix
    A[a_index] = row1
```

```
    A[a_index+1] = row2

    a_index += 2

  U, s, Vt = np.linalg.svd(A)

  #s is a 1-D array of singular values sorted in descending order
  #U, Vt are unitary matrices
  #Rows of Vt are the eigenvectors of A^TA.
  #Columns of U are the eigenvectors of AA^T.
  H = np.eye(3)
  H = Vt[-1].reshape(3,3) # take the last row of the Vt matrix
  return H


def displayplot(img,title):

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

In [5]:
```
def get_inliers(f1, f2, matches, H, RANSACthresh):

  inlier_indices = []
  for i in range(len(matches)):
    queryInd = matches[i].queryIdx
    trainInd = matches[i].trainIdx

    #queryInd = matches[i][0]
    #trainInd = matches[i][1]

    queryPoint = np.array([f1[queryInd].pt[0],  f1[queryInd].pt[1], 1]).T
    trans_query = H.dot(queryPoint)


    comp1 = [trans_query[0]/trans_query[2], trans_query[1]/trans_query[2]] # normalize with respect to z
    comp2 = np.array(f2[trainInd].pt)[:2]


    if(np.linalg.norm(comp1-comp2) <= RANSACthresh): # check against threshold
      inlier_indices.append(i)
  return inlier_indices


def RANSAC_alg(f1, f2, matches, nRANSAC, RANSACthresh):


    minMatches = 4
    nBest = 0
    best_inliers = []
    H_estimate = np.eye(3,3)
    global inlier_matchset
    inlier_matchset=[]
    for iteration in range(nRANSAC):

        #Choose a minimal set of feature matches.
        matchSample = random.sample(matches, minMatches)

        #Estimate the Homography implied by these matches
        im1_pts=np.empty((minMatches,2))
        im2_pts=np.empty((minMatches,2))
        for i in range(0,minMatches):
          m = matchSample[i]
          im1_pts[i] = f1[m.queryIdx].pt
          im2_pts[i] = f2[m.trainIdx].pt
          #im1_pts[i] = f1[m[0]].pt
          #im2_pts[i] = f2[m[1]].pt

        H_estimate=compute_Homography(im1_pts,im2_pts)


        # Calculate the inliers for the H
        inliers = get_inliers(f1, f2, matches, H_estimate, RANSACthresh)

        # if the number of inliers is higher than previous iterations, update the best estimates
        if len(inliers) > nBest:
            nBest= len(inliers)
            best_inliers = inliers

    print("Number of best inliers",len(best_inliers))
    for i in range(len(best_inliers)):
      inlier_matchset.append(matches[best_inliers[i]])

    # compute a homography given this set of matches
    im1_pts=np.empty((len(best_inliers),2))
    im2_pts=np.empty((len(best_inliers),2))
    for i in range(0,len(best_inliers)):
      m = inlier_matchset[i]
      im1_pts[i] = f1[m.queryIdx].pt
      im2_pts[i] = f2[m.trainIdx].pt
      #im1_pts[i] = f1[m[0]].pt
      #im2_pts[i] = f2[m[1]].pt

    M=compute_Homography(im1_pts,im2_pts)
    return M, best_inliers
```

In [6]:
```
tqdm = partial(tqdm, position=0, leave=True)
```

In [7]:
```
files_all=[]
for file in os.listdir("/content/drive/My Drive/Aerial"):
    if file.endswith(".JPG"):
      files_all.append(file)


files_all.sort()
folder_path = '/content/drive/My Drive/Aerial/'

#centre_file = folder_path + files_all[50]
left_files_path_rev = []
right_files_path = []


#Change this according to your dataset split

for file in files_all[:int(len(files_all)/2)+1]:
  left_files_path_rev.append(folder_path + file)

left_files_path = left_files_path_rev[::-1]
```

```
for file in files_all[int(len(files_all)/2):]:
  right_files_path.append(folder_path + file)
```

In [8]:
```
import multiprocessing
print(multiprocessing.cpu_count())
```

```
2
```

In [9]:
```
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.INTER_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.INTER_CUBIC )
  images_right.append(cv2.cvtColor(right_img, cv2.COLOR_BGR2GRAY).astype('float32')/255.)
  images_right_bgr.append(right_img)
```

```
100%|████████| 51/51 [00:59<00:00,  1.17s/it]
100%|████████| 50/50 [00:59<00:00,  1.18s/it]
```

In [10]:
```
f=h5.File('drive/MyDrive/all_images_bgr_sift_40.h5','w')
t0=time.time()
f.create_dataset('data',data=images_left_bgr + images_right_bgr)
f.close()
print('HDF5  w/o comp.:',time.time()-t0,'[s] ... size',os.path.getsize('drive/MyDrive/all_images_bgr_sift_40.h5')/1.e6,'MB')
```

```
HDF5  w/o comp.: 8.098325729370117 [s] ... size 890.822048 MB
```

In [11]:
```
f=h5.File('drive/MyDrive/all_images_gray_sift_40.h5','w')
t0=time.time()
f.create_dataset('data',data=images_left + images_right)
f.close()
print('HDF5  w/o comp.:',time.time()-t0,'[s] ... size',os.path.getsize('drive/MyDrive/all_images_gray_sift_40.h5')/1.e6,'MB')
```

```
HDF5  w/o comp.: 15.97645378112793 [s] ... size 1187.762048 MB
```

In [12]:
```
del images_left_bgr,images_right_bgr
```

In [13]:
```
from timeit import default_timer as timer
time_all = []
```

In [14]:
```
num_kps_sift = []
num_kps_brisk = []
num_kps_agast = []
num_kps_kaze = []
num_kps_akaze = []
num_kps_orb = []
num_kps_mser = []
num_kps_daisy = []
num_kps_surfsift = []
num_kps_fast = []
num_kps_freak = []
num_kps_gftt = []
num_kps_briefstar = []
num_kps_surf = []
num_kps_rootsift = []
num_kps_superpoint = []
```

In [15]:
```
Threshl=50;
Octaves=6;
```

In [17]:
```
start = timer()

mser = cv2.MSER_create()
sift = cv2.xfeatures2d.SIFT_create(Threshl,Octaves)

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 cnt in tqdm(range(len(left_files_path))):
  f=h5.File('drive/MyDrive/all_images_bgr_sift_40.h5','r')
  imgs = f['data'][cnt]
  f.close()
  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 cnt in tqdm(range(len(right_files_path))):
  f=h5.File('drive/MyDrive/all_images_bgr_sift_40.h5','r')
  imgs = f['data'][cnt+len(left_files_path)]
  f.close()
  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]))

end = timer()

time_all.append(end-start)
```

```
100%|████████| 51/51 [05:30<00:00,  6.48s/it]
100%|████████| 50/50 [06:04<00:00,  7.30s/it]
```

In [18]:
```
for j in tqdm(keypoints_all_left_mser + keypoints_all_right_mser[1:]):
```

```
    num_kps_mser.append(len(j))
```

```
100%|██████████| 100/100 [00:00<00:00, 380954.04it/s]
```

In [19]:
```
all_feat_mser_left = []
for cnt,kpt_all in enumerate(keypoints_all_left_mser):
  all_feat_mser_left_each = []
  for cnt_each, kpt in enumerate(kpt_all):
    desc = descriptors_all_left_mser[cnt][cnt_each]
    temp = (kpt.pt, kpt.size, kpt.angle, kpt.response, kpt.octave,
        kpt.class_id, desc)
    all_feat_mser_left_each.append(temp)
  all_feat_mser_left.append(all_feat_mser_left_each)
```

In [20]:
```
all_feat_mser_right = []
for cnt,kpt_all in enumerate(keypoints_all_right_mser):
  all_feat_mser_right_each = []
  for cnt_each, kpt in enumerate(kpt_all):
    desc = descriptors_all_right_mser[cnt][cnt_each]
    temp = (kpt.pt, kpt.size, kpt.angle, kpt.response, kpt.octave,
        kpt.class_id, desc)
    all_feat_mser_right_each.append(temp)
  all_feat_mser_right.append(all_feat_mser_right_each)
```

In [21]:
```
del keypoints_all_left_mser, keypoints_all_right_mser, descriptors_all_left_mser, descriptors_all_right_mser
```

In [22]:
```
import pickle
Fdb = open('all_feat_mser_left.dat', 'wb')
pickle.dump(all_feat_mser_left,Fdb,-1)
Fdb.close()
```

In [23]:
```
import pickle
Fdb = open('all_feat_mser_right.dat', 'wb')
pickle.dump(all_feat_mser_right,Fdb,-1)
Fdb.close()
```

In [24]:
```
del Fdb, all_feat_mser_left, all_feat_mser_right
```

In [25]:
```
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, ransacReprojThreshold =thresh, maxIters=3000)
    inliers = inliers.flatten()
    return H, inliers
```

In [26]:
```
def compute_homography_fast_other(matched_pts1, matched_pts2):
    #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,
                                    0)
    inliers = inliers.flatten()
    return H, inliers
```

In [27]:
```
def get_Hmatrix(imgs,keypts,pts,descripts,ratio=0.75,thresh=4,use_lowe=True,disp=False,no_ransac=False,binary=False):
  lff1 = descripts[0]
  lff = descripts[1]

  if use_lowe==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()
    if binary==True:
      bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)

    else:
      bf = cv2.BFMatcher(cv2.NORM_L2, crossCheck=True)
      lff1 = np.float32(descripts[0])
      lff = np.float32(descripts[1])


    #matches_lf1_lf = flann.knnMatch(lff1, lff, k=2)
    matches_4 = bf.knnMatch(lff1, lff,k=2)
    matches_lf1_lf = []


    print("\nNumber of matches",len(matches_4))
    '''
    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_1.append((m[0].trainIdx, m[0].queryIdx))
      matches_4.append(m[0])
    '''
    print("Number of matches After Lowe's Ratio",len(matches_4))
  else:
    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)
    if binary==True:
      bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
      lff1 = np.float32(descripts[0])
      lff = np.float32(descripts[1])
    else:
      bf = cv2.BFMatcher(cv2.NORM_L2, crossCheck=True)
      lff1 = np.float32(descripts[0])
      lff = np.float32(descripts[1])


    matches_lf1_lf = flann.knnMatch(lff1, lff, k=2)
    #matches_lf1_lf = bf.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_1.append((m[0].trainIdx, m[0].queryIdx))
        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])
  matches_idx = np.array([m.trainIdx for m in matches_4])
  imm2_pts = np.array([keypts[1][idx].pt for idx in matches_idx])
  if no_ransac==True:
    Hn,inliers = compute_homography_fast_other(imm1_pts,imm2_pts)
  else:
    Hn,inliers = compute_homography_fast(imm1_pts,imm2_pts,thresh)

  inlier_matchset = np.array(matches_4)[inliers.astype(bool)].tolist()
  print("Number of Robust matches",len(inlier_matchset))
  print("\n")

  if len(inlier_matchset)<25:
    matches_4 = []
    ratio = 0.85
    # 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(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 [28]:

```
def get_Hmatrix_rfnet(imgs,pts,descripts,disp=True):

  des1 = descripts[0]
  des2 = descripts[1]

  kp1 = pts[0]
  kp2 = pts[1]


  predict_label, nn_kp2 = nearest_neighbor_distance_ratio_match(des1, des2, kp2, 0.7)
  idx = predict_label.nonzero().view(-1)
  mkp1 = kp1.index_select(dim=0, index=idx.long())  # predict match keypoints in I1
  mkp2 = nn_kp2.index_select(dim=0, index=idx.long())  # predict match keypoints in I2

  #img1, img2 = reverse_img(img1), reverse_img(img2)
  keypoints1 = list(map(to_cv2_kp, mkp1))
  keypoints2 = list(map(to_cv2_kp, mkp2))
  DMatch = list(map(to_cv2_dmatch, np.arange(0, len(keypoints1))))

  imm1_pts=np.empty((len(DMatch),2))
  imm2_pts=np.empty((len(DMatch),2))
  for i in range(0,len(DMatch)):
    m = DMatch[i]
    (a_x, a_y) = keypoints1[m.queryIdx].pt
    (b_x, b_y) = keypoints2[m.trainIdx].pt
    imm1_pts[i]=(a_x, a_y)
    imm2_pts[i]=(b_x, b_y)
  H=compute_Homography_fast(imm1_pts,imm2_pts)


  if disp==True:
    dispimg1 = cv2.drawMatches(imgs[0], keypoints1, imgs[1], keypoints2, DMatch, None)
    displayplot(dispimg1,'Robust Matching between Reference Image and Right Image ')


  return H/H[2,2]
```

In [29]:

```
import pickle
Fdb = open('all_feat_mser_left.dat', 'rb')
kpts_all = pickle.load(Fdb)
Fdb.close()

keypoints_all_left_mser = []
descriptors_all_left_mser = []

for j,kpt_each in enumerate(kpts_all):
  keypoints_each = []
  descrip_each = []
  for k,kpt_img in enumerate(kpt_each):
    temp_feature = cv2.KeyPoint(x=kpt_img[0][0],y=kpt_img[0][1],_size=kpt_img[1], _angle=kpt_img[2],
                                _response=kpt_img[3], _octave=kpt_img[4], _class_id=kpt_img[5])
    temp_descriptor = kpt_img[6]
    keypoints_each.append(temp_feature)
    descrip_each.append(temp_descriptor)
  points_all_left_mser.append(np.asarray([[p.pt[0], p.pt[1]] for p in keypoints_each]))
  keypoints_all_left_mser.append(keypoints_each)
  descriptors_all_left_mser.append(descrip_each)
```

In [30]:

```
import pickle
Fdb = open('all_feat_mser_right.dat', 'rb')
kpts_all = pickle.load(Fdb)
Fdb.close()
```

```
keypoints_all_right_mser = []
descriptors_all_right_mser = []

for j,kpt_each in enumerate(kpts_all):
  keypoints_each = []
  descrip_each = []
  for k,kpt_img in enumerate(kpt_each):
    temp_feature = cv2.KeyPoint(x=kpt_img[0][0],y=kpt_img[0][1],_size=kpt_img[1], _angle=kpt_img[2],
                                _response=kpt_img[3], _octave=kpt_img[4], _class_id=kpt_img[5])
    temp_descriptor = kpt_img[6]
    keypoints_each.append(temp_feature)
    descrip_each.append(temp_descriptor)
  points_all_right_mser.append(np.asarray([[p.pt[0], p.pt[1]] for p in keypoints_each]))
  keypoints_all_right_mser.append(keypoints_each)
  descriptors_all_right_mser.append(descrip_each)
```

In [31]:
```
H_left_mser = []
H_right_mser = []
images_left_bgr = []
images_right_bgr = []
num_matches_mser = []
num_good_matches_mser = []

for j in tqdm(range(len(left_files_path))):
  if j==len(left_files_path)-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],0.8,6)
  H_left_mser.append(H_a)
  num_matches_mser.append(matches)
  num_good_matches_mser.append(gd_matches)

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

  H_a,matches,gd_matches = get_Hmatrix(images_right_bgr[j:j+2][::-1],keypoints_all_right_mser[j:j+2][::-1],points_all_right_mser[j:j+2][::-1],descriptors_all_right_mser[j:j+2][::-1],0.8,6)
  H_right_mser.append(H_a)
  num_matches_mser.append(matches)
  num_good_matches_mser.append(gd_matches)
```

```
  2%|▌          | 1/51 [00:00<00:16,  2.98it/s]
Number of matches 2531
Number of matches After Lowe's Ratio 423
Number of Robust matches 235


  4%|▉          | 2/51 [00:00<00:16,  2.99it/s]
Number of matches 2316
Number of matches After Lowe's Ratio 68
Number of Robust matches 48


  6%|▌         | 3/51 [00:01<00:16,  2.99it/s]
Number of matches 2677
Number of matches After Lowe's Ratio 209
Number of Robust matches 137


  8%|▊         | 4/51 [00:01<00:16,  2.93it/s]
Number of matches 2613
Number of matches After Lowe's Ratio 398
Number of Robust matches 218


 10%|▉         | 5/51 [00:01<00:15,  2.90it/s]
Number of matches 2520
Number of matches After Lowe's Ratio 445
Number of Robust matches 296


 12%|▏         | 6/51 [00:02<00:15,  2.94it/s]
Number of matches 2483
Number of matches After Lowe's Ratio 354
Number of Robust matches 196


 14%|▍         | 7/51 [00:02<00:15,  2.90it/s]
Number of matches 2631
Number of matches After Lowe's Ratio 489
Number of Robust matches 277


 16%|▌        | 8/51 [00:02<00:15,  2.86it/s]
Number of matches 2539
Number of matches After Lowe's Ratio 690
Number of Robust matches 415


 18%|▋        | 9/51 [00:03<00:14,  2.89it/s]
Number of matches 2604
Number of matches After Lowe's Ratio 753
Number of Robust matches 511


 20%|▊        | 10/51 [00:03<00:14,  2.90it/s]
Number of matches 2567
Number of matches After Lowe's Ratio 493
Number of Robust matches 322


 22%|▉        | 11/51 [00:03<00:13,  2.87it/s]
Number of matches 2578
Number of matches After Lowe's Ratio 269
Number of Robust matches 165


 24%|▉        | 12/51 [00:04<00:13,  2.85it/s]
Number of matches 2594
Number of matches After Lowe's Ratio 500
Number of Robust matches 281


 25%|▉       | 13/51 [00:04<00:13,  2.91it/s]
Number of matches 2657
Number of matches After Lowe's Ratio 279
Number of Robust matches 160
```

```
 27%|██▋       | 14/51 [00:04<00:12,  2.92it/s]
Number of matches 2452
Number of matches After Lowe's Ratio 405
Number of Robust matches 175


 29%|██▉       | 15/51 [00:05<00:12,  2.91it/s]
Number of matches 2700
Number of matches After Lowe's Ratio 2
Number of Robust matches 0


Number of matches After Lowe's Ratio New 22
Number of Robust matches New 8


 31%|███       | 16/51 [00:05<00:12,  2.84it/s]
Number of matches 2446
Number of matches After Lowe's Ratio 46
Number of Robust matches 24


Number of matches After Lowe's Ratio New 117
Number of Robust matches New 40


 33%|███▎      | 17/51 [00:05<00:11,  2.88it/s]
Number of matches 2809
Number of matches After Lowe's Ratio 484
Number of Robust matches 199


 35%|███▌      | 18/51 [00:06<00:11,  2.78it/s]
Number of matches 3269
Number of matches After Lowe's Ratio 46
Number of Robust matches 23


Number of matches After Lowe's Ratio New 104
Number of Robust matches New 29


 37%|███▋      | 19/51 [00:06<00:12,  2.54it/s]
Number of matches 3600
Number of matches After Lowe's Ratio 53
Number of Robust matches 16


Number of matches After Lowe's Ratio New 123
Number of Robust matches New 29


 39%|███▉      | 20/51 [00:07<00:13,  2.34it/s]
Number of matches 3808
Number of matches After Lowe's Ratio 472
Number of Robust matches 244


 41%|████      | 21/51 [00:07<00:13,  2.19it/s]
Number of matches 3675
Number of matches After Lowe's Ratio 255
Number of Robust matches 106


 43%|████▎     | 22/51 [00:08<00:13,  2.08it/s]
Number of matches 3327
Number of matches After Lowe's Ratio 21
Number of Robust matches 11


Number of matches After Lowe's Ratio New 55
Number of Robust matches New 15


 45%|████▌     | 23/51 [00:08<00:13,  2.14it/s]
Number of matches 3318
Number of matches After Lowe's Ratio 434
Number of Robust matches 170


 47%|████▋     | 24/51 [00:09<00:12,  2.15it/s]
Number of matches 3130
Number of matches After Lowe's Ratio 578
Number of Robust matches 285


 49%|████▉     | 25/51 [00:09<00:11,  2.23it/s]
Number of matches 3375
Number of matches After Lowe's Ratio 503
Number of Robust matches 224


 51%|█████     | 26/51 [00:10<00:11,  2.16it/s]
Number of matches 3932
Number of matches After Lowe's Ratio 614
Number of Robust matches 276


 53%|█████▎    | 27/51 [00:10<00:11,  2.02it/s]
Number of matches 4190
Number of matches After Lowe's Ratio 557
Number of Robust matches 236


 55%|█████▍    | 28/51 [00:11<00:12,  1.89it/s]
Number of matches 4405
Number of matches After Lowe's Ratio 799
Number of Robust matches 352


 57%|█████▋    | 29/51 [00:11<00:12,  1.77it/s]
Number of matches 3922
Number of matches After Lowe's Ratio 695
Number of Robust matches 328


 59%|█████▉    | 30/51 [00:12<00:11,  1.77it/s]
Number of matches 3607
Number of matches After Lowe's Ratio 729
Number of Robust matches 335
```

 61%|██████ | 31/51 [00:13<00:10, 1.82it/s]
Number of matches 3134
Number of matches After Lowe's Ratio 710
Number of Robust matches 301

 63%|██████ | 32/51 [00:13<00:09, 1.96it/s]
Number of matches 2825
Number of matches After Lowe's Ratio 603
Number of Robust matches 328

 65%|██████ | 33/51 [00:13<00:08, 2.12it/s]
Number of matches 2884
Number of matches After Lowe's Ratio 666
Number of Robust matches 355

 67%|██████ | 34/51 [00:14<00:07, 2.22it/s]
Number of matches 2899
Number of matches After Lowe's Ratio 589
Number of Robust matches 269

 69%|██████ | 35/51 [00:14<00:06, 2.32it/s]
Number of matches 2757
Number of matches After Lowe's Ratio 658
Number of Robust matches 327

 71%|███████ | 36/51 [00:14<00:06, 2.45it/s]
Number of matches 2636
Number of matches After Lowe's Ratio 696
Number of Robust matches 412

 73%|███████ | 37/51 [00:15<00:05, 2.58it/s]
Number of matches 2487
Number of matches After Lowe's Ratio 561
Number of Robust matches 343

 75%|███████ | 38/51 [00:15<00:04, 2.70it/s]
Number of matches 2417
Number of matches After Lowe's Ratio 335
Number of Robust matches 191

 76%|███████ | 39/51 [00:15<00:04, 2.83it/s]
Number of matches 2497
Number of matches After Lowe's Ratio 728
Number of Robust matches 385

 78%|███████ | 40/51 [00:16<00:03, 2.88it/s]
Number of matches 2634
Number of matches After Lowe's Ratio 618
Number of Robust matches 359

 80%|████████ | 41/51 [00:16<00:03, 2.86it/s]
Number of matches 2371
Number of matches After Lowe's Ratio 440
Number of Robust matches 249

 82%|████████ | 42/51 [00:16<00:03, 2.89it/s]
Number of matches 2594
Number of matches After Lowe's Ratio 503
Number of Robust matches 278

 84%|████████ | 43/51 [00:17<00:02, 2.85it/s]
Number of matches 2869
Number of matches After Lowe's Ratio 758
Number of Robust matches 410

 86%|████████ | 44/51 [00:17<00:02, 2.78it/s]
Number of matches 2717
Number of matches After Lowe's Ratio 341
Number of Robust matches 175

 88%|████████ | 45/51 [00:18<00:02, 2.76it/s]
Number of matches 2954
Number of matches After Lowe's Ratio 330
Number of Robust matches 156

 90%|████████ | 46/51 [00:18<00:01, 2.66it/s]
Number of matches 3115
Number of matches After Lowe's Ratio 423
Number of Robust matches 173

 92%|████████ | 47/51 [00:18<00:01, 2.55it/s]
Number of matches 3244
Number of matches After Lowe's Ratio 588
Number of Robust matches 266

 94%|████████ | 48/51 [00:19<00:01, 2.43it/s]
Number of matches 3304
Number of matches After Lowe's Ratio 355
Number of Robust matches 126

 96%|████████ | 49/51 [00:19<00:00, 2.37it/s]
Number of matches 3293
Number of matches After Lowe's Ratio 746
Number of Robust matches 306

  0%| | 0/50 [00:00<?, ?it/s]
Number of matches 3152
Number of matches After Lowe's Ratio 103

```
Number of Robust matches 47


  2%|▏          | 1/50 [00:00<00:16,  2.96it/s]
Number of matches 2912
Number of matches After Lowe's Ratio 166
Number of Robust matches 92


  4%|▎          | 2/50 [00:00<00:16,  2.84it/s]
Number of matches 2498
Number of matches After Lowe's Ratio 261
Number of Robust matches 131


  6%|▍          | 3/50 [00:01<00:16,  2.77it/s]
Number of matches 3089
Number of matches After Lowe's Ratio 304
Number of Robust matches 139


  8%|▌          | 4/50 [00:01<00:17,  2.59it/s]
Number of matches 2926
Number of matches After Lowe's Ratio 202
Number of Robust matches 95


 10%|▋          | 5/50 [00:01<00:17,  2.53it/s]
Number of matches 3235
Number of matches After Lowe's Ratio 292
Number of Robust matches 130


 12%|▊          | 6/50 [00:02<00:18,  2.36it/s]
Number of matches 3444
Number of matches After Lowe's Ratio 95
Number of Robust matches 35


 14%|▉          | 7/50 [00:02<00:18,  2.28it/s]
Number of matches 3478
Number of matches After Lowe's Ratio 674
Number of Robust matches 313


 16%|█          | 8/50 [00:03<00:20,  2.08it/s]
Number of matches 3999
Number of matches After Lowe's Ratio 15
Number of Robust matches 8


Number of matches After Lowe's Ratio New 44
Number of Robust matches New 9


 18%|█▏         | 9/50 [00:04<00:20,  2.00it/s]
Number of matches 3203
Number of matches After Lowe's Ratio 178
Number of Robust matches 59


 20%|█▎         | 10/50 [00:04<00:19,  2.04it/s]
Number of matches 2949
Number of matches After Lowe's Ratio 231
Number of Robust matches 85


 22%|█▍         | 11/50 [00:04<00:18,  2.16it/s]
Number of matches 2594
Number of matches After Lowe's Ratio 182
Number of Robust matches 76


 24%|█▌         | 12/50 [00:05<00:16,  2.28it/s]
Number of matches 2756
Number of matches After Lowe's Ratio 364
Number of Robust matches 177


 26%|█▋         | 13/50 [00:05<00:15,  2.40it/s]
Number of matches 2538
Number of matches After Lowe's Ratio 492
Number of Robust matches 284


 28%|█▊         | 14/50 [00:06<00:14,  2.52it/s]
Number of matches 2048
Number of matches After Lowe's Ratio 235
Number of Robust matches 130


 30%|█▉         | 15/50 [00:06<00:12,  2.72it/s]
Number of matches 2889
Number of matches After Lowe's Ratio 154
Number of Robust matches 91


 32%|██         | 16/50 [00:06<00:12,  2.70it/s]
Number of matches 2518
Number of matches After Lowe's Ratio 597
Number of Robust matches 419


 34%|██▏        | 17/50 [00:07<00:11,  2.78it/s]
Number of matches 2561
Number of matches After Lowe's Ratio 315
Number of Robust matches 209


 36%|██▎        | 18/50 [00:07<00:11,  2.78it/s]
Number of matches 2690
Number of matches After Lowe's Ratio 700
Number of Robust matches 462


 38%|██▍        | 19/50 [00:07<00:10,  2.85it/s]
Number of matches 2305
Number of matches After Lowe's Ratio 592
Number of Robust matches 391
```

 40%|███ | 20/50 [00:08<00:10, 2.92it/s]
Number of matches 2632
Number of matches After Lowe's Ratio 600
Number of Robust matches 349

 42%|███ | 21/50 [00:08<00:10, 2.90it/s]
Number of matches 2616
Number of matches After Lowe's Ratio 679
Number of Robust matches 477

 44%|████ | 22/50 [00:08<00:09, 2.86it/s]
Number of matches 2787
Number of matches After Lowe's Ratio 366
Number of Robust matches 236

 46%|████ | 23/50 [00:09<00:09, 2.83it/s]
Number of matches 2817
Number of matches After Lowe's Ratio 539
Number of Robust matches 329

 48%|████▊ | 24/50 [00:09<00:09, 2.72it/s]
Number of matches 2988
Number of matches After Lowe's Ratio 496
Number of Robust matches 242

 50%|█████ | 25/50 [00:09<00:09, 2.63it/s]
Number of matches 3233
Number of matches After Lowe's Ratio 609
Number of Robust matches 326

 52%|█████▏ | 26/50 [00:10<00:09, 2.48it/s]
Number of matches 3308
Number of matches After Lowe's Ratio 462
Number of Robust matches 252

 54%|█████▍ | 27/50 [00:10<00:09, 2.40it/s]
Number of matches 3326
Number of matches After Lowe's Ratio 757
Number of Robust matches 406

 56%|█████▌ | 28/50 [00:11<00:09, 2.33it/s]
Number of matches 3221
Number of matches After Lowe's Ratio 571
Number of Robust matches 254

 58%|█████▊ | 29/50 [00:11<00:08, 2.35it/s]
Number of matches 3120
Number of matches After Lowe's Ratio 857
Number of Robust matches 411

 60%|██████ | 30/50 [00:12<00:08, 2.38it/s]
Number of matches 3373
Number of matches After Lowe's Ratio 712
Number of Robust matches 307

 62%|██████▏ | 31/50 [00:12<00:08, 2.29it/s]
Number of matches 3546
Number of matches After Lowe's Ratio 432
Number of Robust matches 230

 64%|██████▍ | 32/50 [00:13<00:08, 2.21it/s]
Number of matches 3310
Number of matches After Lowe's Ratio 342
Number of Robust matches 152

 66%|██████▌ | 33/50 [00:13<00:07, 2.15it/s]
Number of matches 3974
Number of matches After Lowe's Ratio 129
Number of Robust matches 48

 68%|██████▊ | 34/50 [00:14<00:08, 1.87it/s]
Number of matches 3784
Number of matches After Lowe's Ratio 542
Number of Robust matches 265

 70%|███████ | 35/50 [00:14<00:08, 1.78it/s]
Number of matches 4078
Number of matches After Lowe's Ratio 10
Number of Robust matches 6

Number of matches After Lowe's Ratio New 46
Number of Robust matches New 9

 72%|███████▏ | 36/50 [00:15<00:07, 1.79it/s]
Number of matches 3642
Number of matches After Lowe's Ratio 427
Number of Robust matches 190

 74%|███████▍ | 37/50 [00:15<00:06, 1.87it/s]
Number of matches 3479
Number of matches After Lowe's Ratio 424
Number of Robust matches 191

 76%|███████▌ | 38/50 [00:16<00:06, 1.95it/s]
Number of matches 3150
Number of matches After Lowe's Ratio 579
Number of Robust matches 236

```
 78%|███████▊  | 39/50 [00:16<00:05,  2.06it/s]
Number of matches 3570
Number of matches After Lowe's Ratio 357
Number of Robust matches 140


 80%|████████  | 40/50 [00:17<00:04,  2.08it/s]
Number of matches 3527
Number of matches After Lowe's Ratio 443
Number of Robust matches 162


 82%|████████▏ | 41/50 [00:17<00:04,  2.09it/s]
Number of matches 3480
Number of matches After Lowe's Ratio 577
Number of Robust matches 219


 84%|████████▍ | 42/50 [00:18<00:03,  2.12it/s]
Number of matches 3385
Number of matches After Lowe's Ratio 895
Number of Robust matches 405


 86%|████████▌ | 43/50 [00:18<00:03,  2.15it/s]
Number of matches 3303
Number of matches After Lowe's Ratio 584
Number of Robust matches 310


 88%|████████▊ | 44/50 [00:19<00:02,  2.19it/s]
Number of matches 2947
Number of matches After Lowe's Ratio 55
Number of Robust matches 23


Number of matches After Lowe's Ratio New 154
Number of Robust matches New 51


 90%|█████████ | 45/50 [00:19<00:02,  2.28it/s]
Number of matches 2865
Number of matches After Lowe's Ratio 435
Number of Robust matches 252


 92%|█████████▏| 46/50 [00:19<00:01,  2.36it/s]
Number of matches 3103
Number of matches After Lowe's Ratio 338
Number of Robust matches 199


 94%|█████████▍| 47/50 [00:20<00:01,  2.43it/s]
Number of matches 2596
Number of matches After Lowe's Ratio 177
Number of Robust matches 94


 96%|█████████▌| 48/50 [00:20<00:00,  2.54it/s]
Number of matches 2509
Number of matches After Lowe's Ratio 350
Number of Robust matches 199


 98%|█████████▊| 49/50 [00:20<00:00,  2.69it/s]
Number of matches 2569
Number of matches After Lowe's Ratio 335
Number of Robust matches 187
```

In [32]:
```
import h5py as h5
f=h5.File('drive/MyDrive/H_left_mser_40.h5','w')
t0=time.time()
f.create_dataset('data',data=H_left_mser)
f.close()
print('HDF5  w/o comp.:',time.time()-t0,'[s] ... size',os.path.getsize('drive/MyDrive/H_left_mser_40.h5')/1.e6,'MB')
```

```
HDF5  w/o comp.: 0.004593610763549805 [s] ... size 0.005648 MB
```

In [33]:
```
import h5py as h5
f=h5.File('drive/MyDrive/H_right_mser_40.h5','w')
t0=time.time()
f.create_dataset('data',data=H_right_mser)
f.close()
print('HDF5  w/o comp.:',time.time()-t0,'[s] ... size',os.path.getsize('drive/MyDrive/H_right_mser_40.h5')/1.e6,'MB')
```

```
HDF5  w/o comp.: 0.003834247589111328 [s] ... size 0.005576 MB
```

In [34]:
```
del H_left_mser, H_right_mser,keypoints_all_left_mser, keypoints_all_right_mser, descriptors_all_left_mser, descriptors_all_right_mser, points_all_left_mser, points_all_right_mser
```

In [35]:
```
len_files = len(left_files_path) + len(right_files_path[1:])
num_detectors = 1
Dataset = 'University Campus'
```

In [36]:
```
d = {'Dataset': [f'{Dataset}']*(num_detectors*len_files), 'Number of Keypoints': num_kps_mser, 'Detector/Descriptor':['MSER+SIFT']*len_files }
df_numkey_16 = pd.DataFrame(data=d)
df_numkey_16['Number of Keypoints'] = df_numkey_16['Number of Keypoints']/(len_files)
```

In [37]:
```
df_numkey_16.to_csv('drive/MyDrive/Num_Kypoints_16.csv')
```

In [38]:
```
d = {'Dataset': [f'{Dataset}']*(num_detectors*(len_files-1)), 'Number of Total Matches': num_matches_mser, 'Detector/Descriptor':['MSER+SIFT']*(len_files-1) }
df_match_16 = pd.DataFrame(data=d)
df_match_16['Number of Total Matches'] = df_match_16['Number of Total Matches']/(len_files-1)
```

In [39]:
```
df_match_16 = pd.DataFrame(data=d)
df_match_16['Number of Total Matches'] = df_match_16['Number of Total Matches']/(len_files-1)
```

In [40]:
```
df_match_16['Number of Good Matches'] =  num_good_matches_mser
df_match_16['Number of Good Matches'] = df_match_16['Number of Good Matches']/(len_files-1)
```

In [41]:
```
df_match_16['Recall Rate of Matches'] = df_match_16['Number of Good Matches']/df_match_16['Number of Total Matches']
```

In [42]:
```
df_match_16['1 - Precision Rate of Matches'] = (df_match_16['Number of Total Matches'] - df_match_16['Number of Good Matches'])/df_match_16['Number of Total Matches']
```

In [43]:
```
df_match_16['F-Score'] = (2* (1 - df_match_16['1 - Precision Rate of Matches']) * df_match_16['Recall Rate of Matches'])/((1 - df_match_16['1 - Precision Rate of Matches']) + df_match_16['Rec
```

In [44]:
```
df_match_16.to_csv('drive/MyDrive/All_metrics_16.csv')
```

In [45]:
```
d = {'Dataset': [f'{Dataset}']*(num_detectors), 'Time': [time_all[0]], 'Detector/Descriptor': ['MSER+SIFT']*(1) }
df_time_16 = pd.DataFrame(data=d)
```

In [46]:
```
print(df_time_16)
```

```
            Dataset        Time Detector/Descriptor
0  University Campus  695.318446           MSER+SIFT
```