```
import cv2
import matplotlib.pyplot as plt
import numpy as np
from skimage.io import imread, imshow
%matplotlib inline


# Load the image
image1 = cv2.imread('EP-00-00012_0119_0003.JPG')
image2 = cv2.imread('EP-00-00012_0119_0004.JPG')

# Convert the training image to gray scale
gray1 = cv2.cvtColor(image1, cv2.COLOR_BGR2GRAY)
gray2 = cv2.cvtColor(image2, cv2.COLOR_BGR2GRAY)

# Display traning image and testing image
fx, plots = plt.subplots(1, 2, figsize=(20,10))
plots[0].set_title("Training Image")
plots[0].imshow(gray1)
plots[1].set_title("Testing Image")
plots[1].imshow(gray2)
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call last)
<ipython-input-11-5e5b819add10> in <module>()
      4
      5 # Convert the training image to gray scale
----> 6 gray1 = cv2.cvtColor(image1, cv2.COLOR_BGR2GRAY)
      7 gray2 = cv2.cvtColor(image2, cv2.COLOR_BGR2GRAY)
      8

Saved successfully!

/modules/imgproc/src/color.cpp:182: error:
(-215:Assertion failed) !_src.empty() in function 'cvtColor'

SEARCH STACK OVERFLOW
```

```
!pip install opencv-python==3.4.2.17
!pip3 install opencv-contrib-python==3.4.2.17
```

```
Requirement already satisfied: opencv-python==3.4.2.17 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: opencv-contrib-python==3.4.2.17 in /usr/local/lib/python3
Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (
```

```
# BRISK detector
BRISK = cv2.BRISK_create()

# find the keypoints and descriptors with BRISK
kp1, des1 = brisk.detectAndCompute(image1,None)
kp2, des2 = brisk.detectAndCompute(image2,None)
```

```
# Brute Force FMatcher with default params
bf = cv2.BFMatcher()
matches = bf.knnMatch(des1,des2, k=2) #KnnMatch clusters the matches in order to separate eac

# Applying ratio test
  #Ratio test is a outlier-removal technique that returns Two nearest descriptors for each ma
  #The match is returned only if the distance ratio between the first and second matches is b
feature = [[m] for m, n in matches if m.distance < 0.7*n.distance]
img3 = cv2.drawMatchesKnn(training_gray, kp1, test_gray, kp2, feature, None, flags=2)
plt.imshow(img3),plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-10-5da15d2856ae> in <module>()
      3
      4 # find the keypoints and descriptors with SIFT
----> 5 kp1, des1 = brisk.detectAndCompute(image1,None)
      6 kp2, des2 = brisk.detectAndCompute(image2,None)
      7

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

SEARCH STACK OVERFLOW

```
# Select good matched keypoints
ref_matched_kpts = np.float32([kp1[m[0].queryIdx].pt for m in feature])
sensed_matched_kpts = np.float32([kp2[m[0].trainIdx].pt for m in feature])

# Compute homography
                                                  d_matched_kpts, ref_matched_kpts, cv2.RANSAC,5.0)
```

Saved successfully!

```
warped_image = cv2.warpPerspective(img3, H, (img3.shape[1], img3.shape[0]))
cv2.imwrite('warped.jpg', warped_image)
cv2.imshow(warped_image)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-8-c47e1663ed64> in <module>()
      1 # Select good matched keypoints
----> 2 ref_matched_kpts = np.float32([kp1[m[0].queryIdx].pt for m in feature])
      3 sensed_matched_kpts = np.float32([kp2[m[0].trainIdx].pt for m in
feature])
      4
      5 # Compute homography

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

SEARCH STACK OVERFLOW

0s completed at 22:20

Saved successfully!