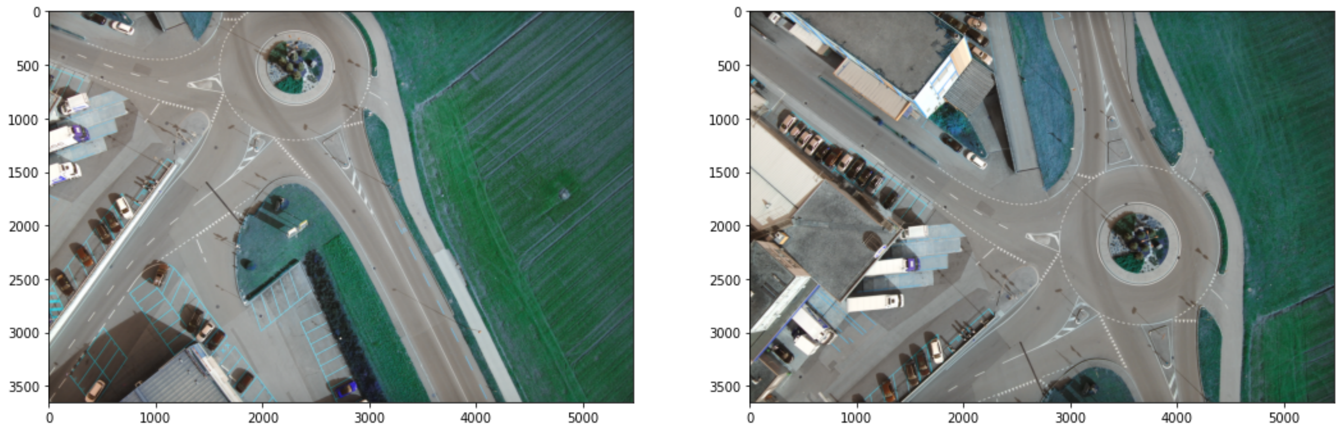
```
# import the necessary packages
import cv2
import numpy as np
import matplotlib.pyplot as plt
from skimage.io import imread, imshow
from random import randrange
```

```
#Reading the Images
img_ = cv2.imread('EP-00-00012_0119_0003.JPG')
img = cv2.imread('EP-00-00012_0119_0004.JPG')
```

```
figure, ax = plt.subplots(1, 2, figsize=(18, 8))
ax[0].imshow(img_)
ax[1].imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7fe5f992e410>
```

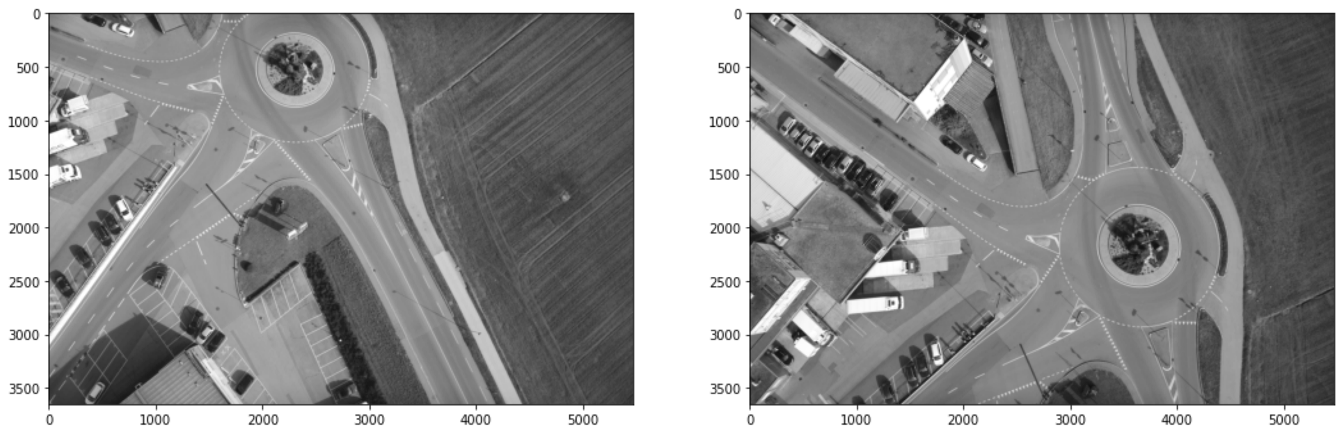


```
#Resizing of Images
img_ = cv2.resize(img_, (0,0), fx=1, fy=1)
img = cv2.resize(img, (0,0), fx=1, fy=1)
```

```
#Coverting RGB Images to Grey Scale
img1 = cv2.imread('EP-00-00012_0119_0003.JPG', cv2.IMREAD_GRAYSCALE)
img2 = cv2.imread('EP-00-00012_0119_0004.JPG', cv2.IMREAD_GRAYSCALE)
```

```
figure, ax = plt.subplots(1, 2, figsize=(18, 8))
ax[0].imshow(img1, cmap='gray')
ax[1].imshow(img2, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7fe5f8b9c390>
```



```
!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 (
```

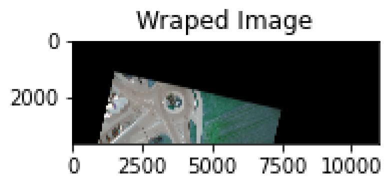
```
#Using Sift for defining Key points and descriptors for each image
sift = cv2.xfeatures2d.SIFT_create()

# find the keypotnts and descriptors with SIFT
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)
```

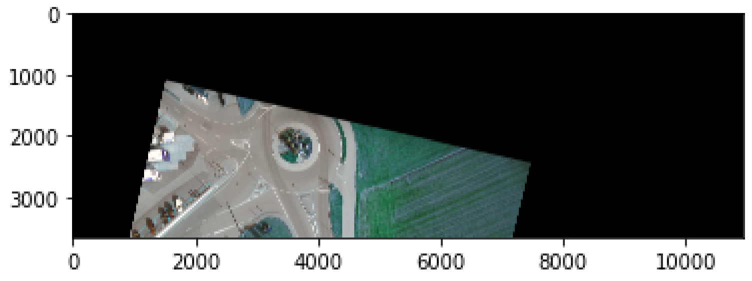
```
print('No.of key points in image1: ',len(kp1), '\n No.of key points in image2: ',len(kp2))
```

```
No.of key points in image1:  122951
 No.of key points in image2:  70171
```

```
#Feature Matching

#Using Brute Force Matches, KNN (Keeping value as 2 since we are applying for 2 images)
bf = cv2.BFMatcher()
```

```
matches = bf.knnMatch(des1,des2, k=2)


# Apply ratio test
good = []
for m in matches:
  if m[0].distance < 0.5*m[1].distance:
    good.append(m)
matches = np.asarray(good)



if len(matches[:,0]) >= 4:
  src = np.float32([ kp1[m.queryIdx].pt for m in matches[:,0] ]).reshape(-1,1,2)
  dst = np.float32([ kp2[m.trainIdx].pt for m in matches[:,0] ]).reshape(-1,1,2)
  H, masked = cv2.findHomography(src, dst, cv2.RANSAC, 5.0)
#prtnt 8
else:
  raise AssertionError("Can’t find enough keypoints.")



#Wrapping Images (Stitching)
dst = cv2.warpPerspective(img_,H,(img.shape[1]+img_.shape[1], img.shape[0]))
plt.subplot(122),plt.imshow(dst),plt.title('Wraped Image')
plt.show()
plt.figure()

cv2.imwrite('Output.jpg' ,dst)
plt.imshow(dst)
plt.show()
```





✓ 6s completed at 04:13