Name : Vakeesan.K

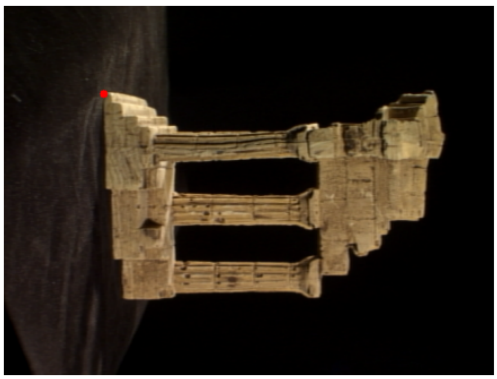
Index N.O:190643G

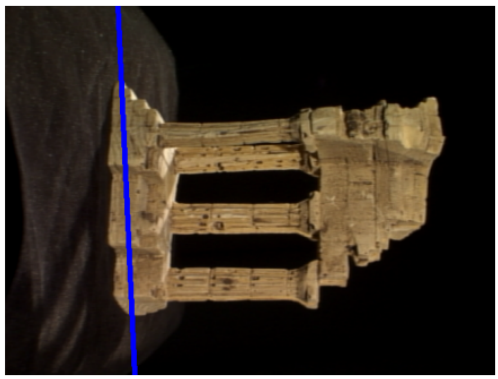
1) to 3)

In [ ]:
```
%matplotlib inline
import numpy as np
import cv2 as cv
from scipy.linalg import null_space
import matplotlib.pyplot as plt

f = open(r'C:\Python39\cv\exercices\lec 8\templeSparseRing\templeSR_par.txt','r')
assert f is not None

n = int(f.readline())

#reading the information on the first image
l=f.readline().split()
im1_fn = l[0]
K1 = np.array([float(i) for i in l[1:10]]).reshape(3,3)
R1 = np.array([float(i) for i in l[10:19]]).reshape(3,3)
t1 = np.array([float(i) for i in l[19:22]]).reshape(3,1)

#reaing the information on the second image
l=f.readline().split()
im2_fn = l[0]
K2 = np.array([float(i) for i in l[1:10]]).reshape(3,3)
R2 = np.array([float(i) for i in l[10:19]]).reshape(3,3)
t2 = np.array([float(i) for i in l[19:22]]).reshape(3,1)

#read the two images and show
im1 =cv.imread(r'C:\Python39\cv\exercices\lec 8\templeSparseRing/'+im1_fn, cv.IMRE
im2 =cv.imread(r'C:\Python39\cv\exercices\lec 8\templeSparseRing/'+im2_fn, cv.IMRE
assert im1 is not None
assert im2 is not None

#compute p1 and p2
P1 = K1 @ np.hstack((R1,t1))
P2 = K2 @ np.hstack((R2,t2)) #P =K*[R| t]

#compute the fundamental matrix
def skew(x):
    x=x.ravel()
    return np.array([[0,-x[2],x[1]],[x[2], 0,-x[0]],[-x[1],x[0],0]])
c = null_space(P1)
c = c *np.sign(c[0,0])
e2 =P2 @ c

e2x =skew(e2)

F = e2x @ P2 @ np.linalg.pinv(P1)
print('F =',F)

#compute epipolar line
x = np.array([130,115,1])
cv.circle(im1,(x[0],x[1]),5,(0,0,255),-1)
l2 =F @ x.T
p1 = np.array([0,(l2[0]*0+l2[2])/l2[1]]).astype(int)
p2 = np.array([500,(l2[0]*500+l2[2])/l2[1]]).astype(int)
cv.line(im2,(p1[0],p1[1]),(p2[0],p2[1]),(255,0,0),5)
fig,ax=plt.subplots(1,2,figsize=(20,6))
```

```
ax[0].imshow(cv.cvtColor(im1,cv.COLOR_BGR2RGB))
ax[0].axis("off")
ax[1].imshow(cv.cvtColor(im2,cv.COLOR_BGR2RGB))
ax[1].axis("off")
plt.show()
```

```
F = [[-2.87071497e-04 -3.96261289e-02  2.94221686e+02]
 [-3.55039713e-02  1.65329260e-04  1.78860854e+01]
 [-2.76702814e+02  2.12942175e+01 -9.06669374e+03]]
```





4)

In [ ]:

```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt

def drawlines(img1,img2,lines,pts1,pts2):
    ''' img1 - image on which we draw the epilines for the points in img2
        lines - corresponding epilines '''
    r,c = img1.shape
    img1 = cv.cvtColor(img1,cv.COLOR_GRAY2BGR)
    img2 = cv.cvtColor(img2,cv.COLOR_GRAY2BGR)
    for r,pt1,pt2 in zip(lines,pts1,pts2):
        color = tuple(np.random.randint(0,255,3).tolist())
        x0,y0 = map(int, [0, -r[2]/r[1] ])
        x1,y1 = map(int, [c, -(r[2]+r[0]*c)/r[1] ])
        img1 = cv.line(img1, (x0,y0), (x1,y1), color,1)
        img1 = cv.circle(img1,tuple(pt1),5,color,-1)
        img2 = cv.circle(img2,tuple(pt2),5,color,-1)
    return img1,img2
img1 =cv.imread(r'C:\Python39\cv\exercices\lec 8\templeSparseRing\templeSR0001.png
img2 =cv.imread(r'C:\Python39\cv\exercices\lec 8\templeSparseRing\templeSR0004.png
assert img1 is not None

sift = cv.SIFT_create()
# find the keypoints and descriptors with SIFT
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)
# FLANN parameters
FLANN_INDEX_KDTREE = 1
index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
search_params = dict(checks=50)
flann = cv.FlannBasedMatcher(index_params,search_params)
matches = flann.knnMatch(des1,des2,k=2)
pts1 = []
pts2 = []
# ratio test as per Lowe's paper
for i,(m,n) in enumerate(matches):
    if m.distance < 0.8*n.distance:
        pts2.append(kp2[m.trainIdx].pt)
        pts1.append(kp1[m.queryIdx].pt)
```

```
pts1 = np.int32(pts1)
pts2 = np.int32(pts2)
F, mask = cv.findFundamentalMat(pts1,pts2,cv.FM_LMEDS)
# We select only inlier points
pts1 = pts1[mask.ravel()==1]
pts2 = pts2[mask.ravel()==1]
# Find epilines corresponding to points in right image (second image) and
# drawing its lines on left image
lines1 = cv.computeCorrespondEpilines(pts2.reshape(-1,1,2), 2,F)
lines1 = lines1.reshape(-1,3)
img5,img6 = drawlines(img1,img2,lines1,pts1,pts2)
# Find epilines corresponding to points in left image (first image) and
# drawing its lines on right image
lines2 = cv.computeCorrespondEpilines(pts1.reshape(-1,1,2), 1,F)
lines2 = lines2.reshape(-1,3)
img3,img4 = drawlines(img2,img1,lines2,pts2,pts1)
fig,ax=plt.subplots(1,2,figsize=(20,6))
ax[0].imshow(cv.cvtColor(img5,cv.COLOR_BGR2RGB))
ax[0].axis("off")
ax[1].imshow(cv.cvtColor(img3,cv.COLOR_BGR2RGB))
ax[1].axis("off")
plt.show()
```