ac-and-projective-transformations

November 28, 2024

0.1 Lab Exercise 6: RANSAC and Projective Transformations

- **Objective:** Implement RANSAC for homography estimation and projective transformations.
- **Task**: Use RANSAC to estimate the homography between two images, and then apply projective transformation to create a panorama or mosaic.

[1]:
```
!pip install opencv-contrib-python
```

```
Requirement already satisfied: opencv-contrib-python in
/usr/local/lib/python3.10/dist-packages (4.10.0.84)
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.10/dist-
packages (from opencv-contrib-python) (1.26.4)
```

[2]:
```
import cv2
print(cv2.__version__)
```

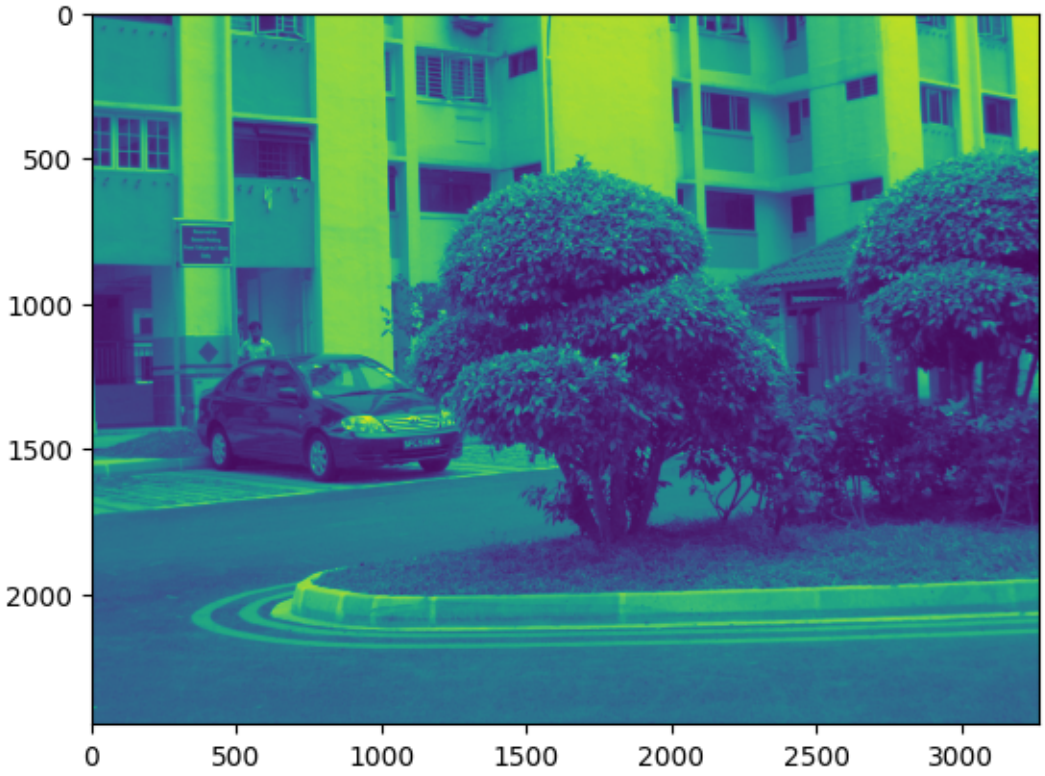
```
4.10.0
```

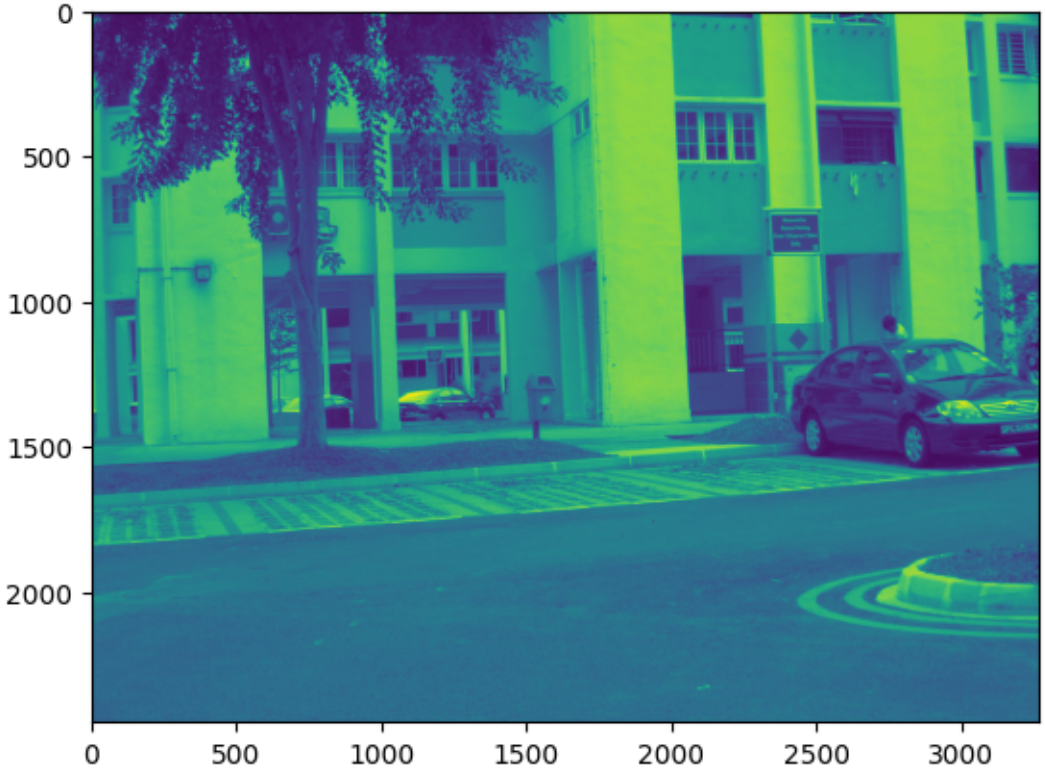
[3]:
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

[4]:
```
# Load the images using OpenCV
img_1 = cv2.imread('img2.JPG')
img_2 = cv2.imread('img1.JPG')

# Convert the images from BGR to RGB
img_1_rgb = cv2.cvtColor(img_1, cv2.COLOR_BGR2RGB)
img_2_rgb = cv2.cvtColor(img_2, cv2.COLOR_BGR2RGB)

# Display the first image
plt.imshow(img_1_rgb)
plt.axis('off')  # Optional: remove axes
plt.title('Image 1')  # Optional: add a title
plt.show()

# Display the second image
plt.imshow(img_2_rgb)
```

```
plt.axis('off')  # Optional: remove axes
plt.title('Image 2')  # Optional: add a title
plt.show()
```

[5]:
```
img1 = cv2.cvtColor(img_1_rgb,cv2.COLOR_BGR2GRAY)
plt.imshow(img1)
plt.show()
img2 = cv2.cvtColor(img_2_rgb,cv2.COLOR_BGR2GRAY)
plt.imshow(img2)
plt.show()
```


0
500
1000
1500
2000
0
500
1000
1500
2000
2500
3000


0
500
1000
1500
2000
0
500
1000
1500
2000
2500
3000

[6]:
```
sift = cv2.SIFT_create()
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)
```

[7]:
```
bf = cv2.BFMatcher()
```

[8]:
```
matches = bf.knnMatch(des1,des2, k=2)
```

[9]:
```
good = []
for m in matches:
    if (m[0].distance < 0.5*m[1].distance):
        good.append(m)
matches = np.asarray(good)
```

[10]:
```
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)
else:
    raise AssertionError('Can't find enough keypoints.')
```

[11]:
```
dst = cv2.warpPerspective(img_1_rgb,H,((img_1_rgb.shape[1] + img_2_rgb.
 ↪shape[1]), img_2_rgb.shape[0])) #wraped image
dst[0:img_2_rgb.shape[0], 0:img_2_rgb.shape[1]] = img_2_rgb #stitched image
cv2.imwrite('output.jpg',dst)
plt.imshow(dst)
plt.show()
```

[11]: