

## ✓ Anirbaan Ghatak C026 B1 EXP 4

### Aim:

Task1 : Download the given image

Task2: Determine corner response using Harris Corner detector

Task3: Apply threshold to detect corners

Task4: resize, rotate, change brightness and contrast of the image and repeat task 2 and task3

Task5: Use corner response to identify edges

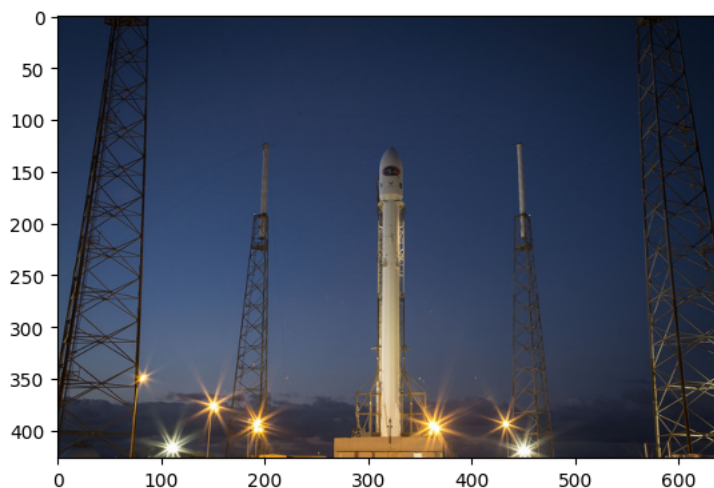
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from skimage import data
```

```
image = cv2.imread('image')
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
new_image = data.rocket()
```

```
image_g = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
new_image_g = cv2.cvtColor(new_image, cv2.COLOR_RGB2GRAY)
```

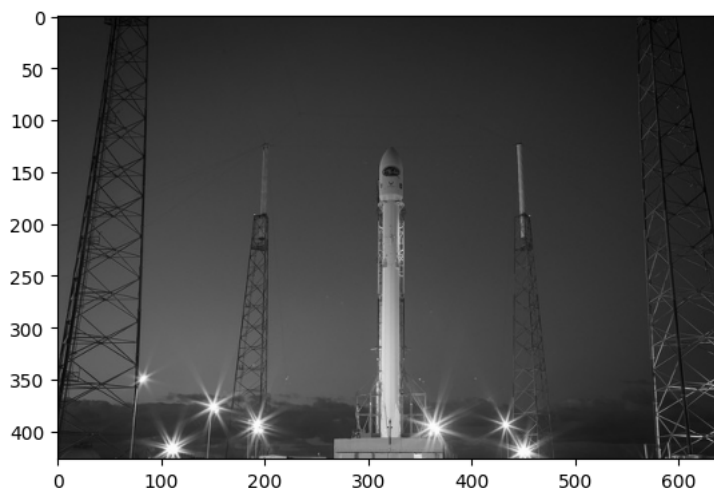
```
plt.imshow(new_image)
```

 <matplotlib.image.AxesImage at 0x7a9b53d43a50>



```
plt.imshow(new_image_g, cmap='gray')
```

 <matplotlib.image.AxesImage at 0x7a9b53b30350>



```
img_harris=cv2.cornerHarris(image_g,5,5,0.04)
new_img_harris=cv2.cornerHarris(new_image_g,5,5,0.04)
```


```
rows,cols = img_harris.shape
n_rows,n_cols = new_img_harris.shape
```

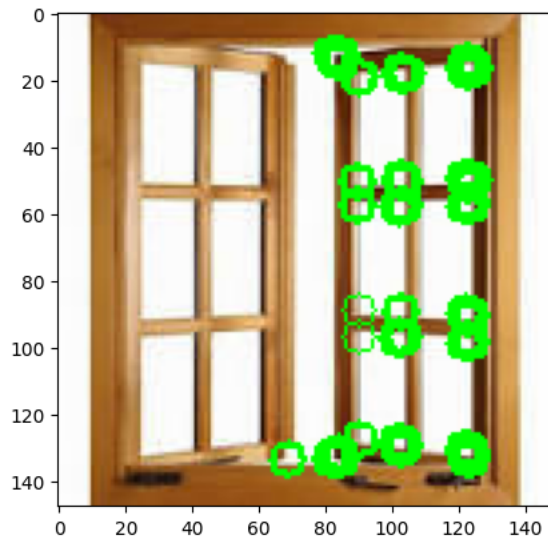
```
th = 0.2*img_harris.max()
n_th = 0.2*new_img_harris.max()
```

```
for r in range(0,rows):
    for c in range(0, cols):
        if img_harris[r,c] > th:

            cv2.circle(image,(c,r),5,(0,255,0),1)
```

```
plt.imshow(image)
```

 <matplotlib.image.AxesImage at 0x7a9b53b88f50>



```
for n_r in range(0,n_rows):
    for n_c in range(0,n_cols):
        if new_img_harris[n_r,n_c] > n_th:

            cv2.circle(new_image,(n_c,n_r),5,(0,255,0),1)
```

```
plt.imshow(new_image)
```

 <matplotlib.image.AxesImage at 0x7a9b539c3a50>



```
img1 = image
gray1 = cv2.cvtColor(img1, cv2.COLOR_RGB2GRAY)
```

```
img2 = cv2.rotate(img1, cv2.ROTATE_90_CLOCKWISE)
gray2 = cv2.cvtColor(img2, cv2.COLOR_RGB2GRAY)
```

```
img3 = cv2.convertScaleAbs(img1, beta=-10)
gray3 = cv2.cvtColor(img3, cv2.COLOR_RGB2GRAY)
```

```
img4 = cv2.convertScaleAbs(img1, alpha=1.2)
gray4 = cv2.cvtColor(img4, cv2.COLOR_RGB2GRAY)
```

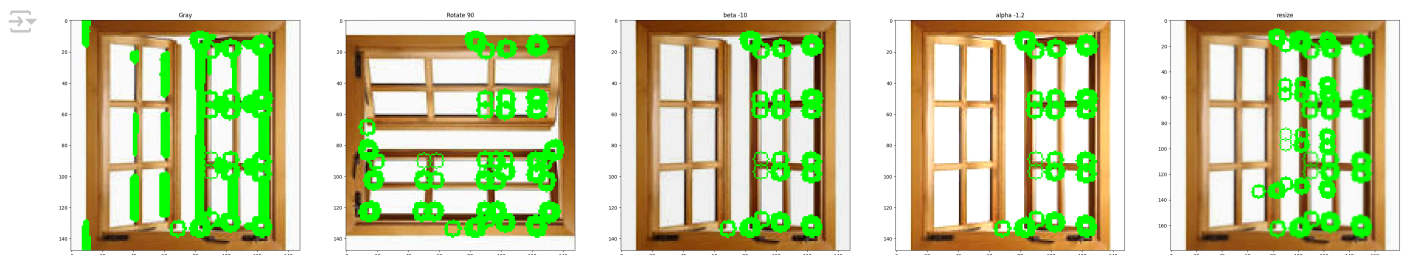
```
img5 = cv2.resize(img1, (180,180))
gray5 = cv2.cvtColor(img5, cv2.COLOR_RGB2GRAY)
```

```
for r in range(0,rows):
    for c in range(0, cols):
        if img_harris[r,c] > th:

            cv2.circle(img1,(c,r),5,(0,255,0),1)
            # cv2.circle(gray1,(c,r),5,(0,255,0),1)
            cv2.circle(img2,(c,r),5,(0,255,0),1)
            # cv2.circle(gray2,(c,r),5,(0,255,0),1)
            cv2.circle(img3,(c,r),5,(0,255,0),1)
            # cv2.circle(gray3,(c,r),5,(0,255,0),1)
            cv2.circle(img4,(c,r),5,(0,255,0),1)
            # cv2.circle(gray4,(c,r),5,(0,255,0),1)
            cv2.circle(img5,(c,r),5,(0,255,0),1)
            # cv2.circle(gray5,(c,r),5,(0,255,0),1)
```

```
fig, axs = plt.subplots(1, 5, figsize=(50, 50))
```

```
axs[0].imshow(img1)
axs[0].title.set_text('Gray')
axs[1].imshow(img2)
axs[1].title.set_text('Rotate 90')
axs[2].imshow(img3)
axs[2].title.set_text('beta -10')
axs[3].imshow(img4)
axs[3].title.set_text('alpha -1.2')
axs[4].imshow(img5)
axs[4].title.set_text('resize')
```



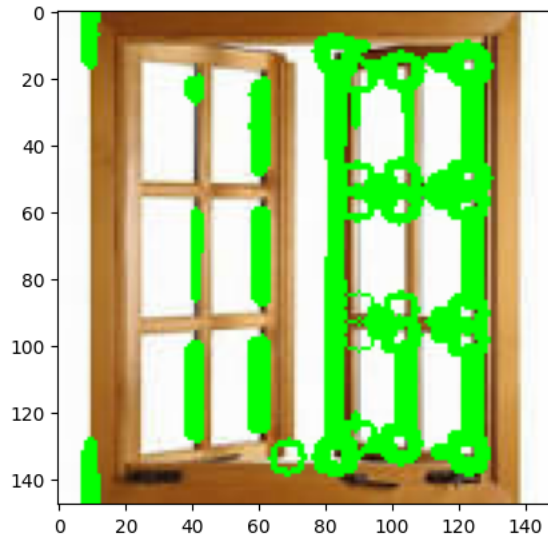
```
neg_th= 0.2*img_harris.min()

for r in range(0, rows):
    for c in range(0, cols):
        if img_harris[r,c] < neg_th:

            cv2.circle(image,(c,r),1,(0,255,0),1)

plt.imshow(image)
```

 <matplotlib.image.AxesImage at 0x7a9b5c1bc810>



```
n_img1 = new_image
n_gray1 = cv2.cvtColor(n_img1, cv2.COLOR_RGB2GRAY)

n_img2 = cv2.rotate(n_img1, cv2.ROTATE_90_CLOCKWISE)
n_gray2 = cv2.cvtColor(n_img2, cv2.COLOR_RGB2GRAY)

n_img3 = cv2.convertScaleAbs(n_img1, beta=-10)
n_gray3 = cv2.cvtColor(n_img3, cv2.COLOR_RGB2GRAY)

n_img4 = cv2.convertScaleAbs(n_img1, alpha=1.2)
n_gray4 = cv2.cvtColor(n_img4, cv2.COLOR_RGB2GRAY)

n_img5 = cv2.resize(n_img1, (180,180))
n_gray5 = cv2.cvtColor(n_img5, cv2.COLOR_RGB2GRAY)

for r in range(0,rows):
    for c in range(0, cols):
        if img_harris[r,c] > th:

            cv2.circle(n_img1,(c,r),5,(0,255,0),1)
            cv2.circle(n_img2,(c,r),5,(0,255,0),1)
            cv2.circle(n_img3,(c,r),5,(0,255,0),1)
            cv2.circle(n_img4,(c,r),5,(0,255,0),1)
            cv2.circle(n_img5,(c,r),5,(0,255,0),1)
```

## ✓ Conclusion

harris corner detector is used to identify the corners and edges of the window(window.jfif), threshold to identify the corner is taken as 10% of the maximum value of corner response, if this image is rotated, brightness is reduced by 10, contrast is reduced by 1.2 and is resized by 180x180 then the same corners are detected in all the modified versions of the image.

This shows that the harris corner detector is robust against rotation, resizing and change in illumination of the image

To detect edge pixels, corner response at the pixel should be large and negative from the given image threshold to detect the edge pixel is considered 20% of the minimum value (which is negative) the detected pixels show the edges of the window.

the above process is applied to the Rocket image and it shows the corners of the image

```
fig, axs = plt.subplots(1, 5, figsize=(30, 30))

axs[0].imshow(n_img1)
axs[0].title.set_text('Gray')
axs[1].imshow(n_img2)
axs[1].title.set_text('Rotate 90')
axs[2].imshow(n_img3)
axs[2].title.set_text('beta -10')
axs[3].imshow(n_img4)
axs[3].title.set_text('alpha -1.2')
axs[4].imshow(n_img5)
axs[4].title.set_text('resize')
```

