Corner Detection

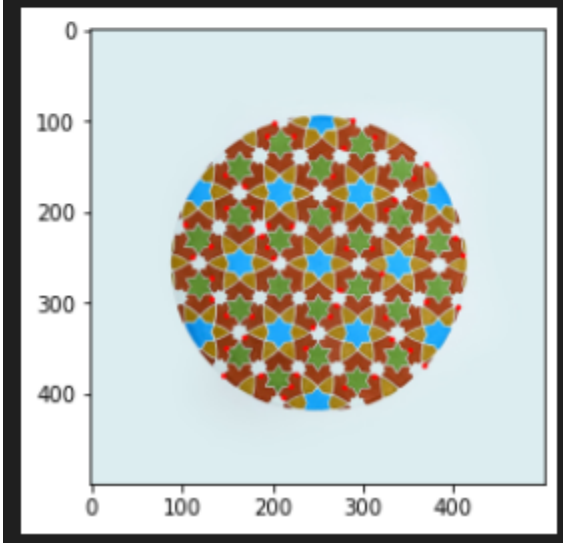
```python
import numpy as np
import cv2
import matplotlib.pyplot as plt


img = cv2.imread('shape.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)


corners = cv2.goodFeaturesToTrack(gray, 47, 0.01, 20)
corners = np.int0(corners)


for i in corners:
    x, y = i.ravel()
    cv2.circle(img,(x,y),3,255,-1)


plt.imshow(img)
plt.show()
```



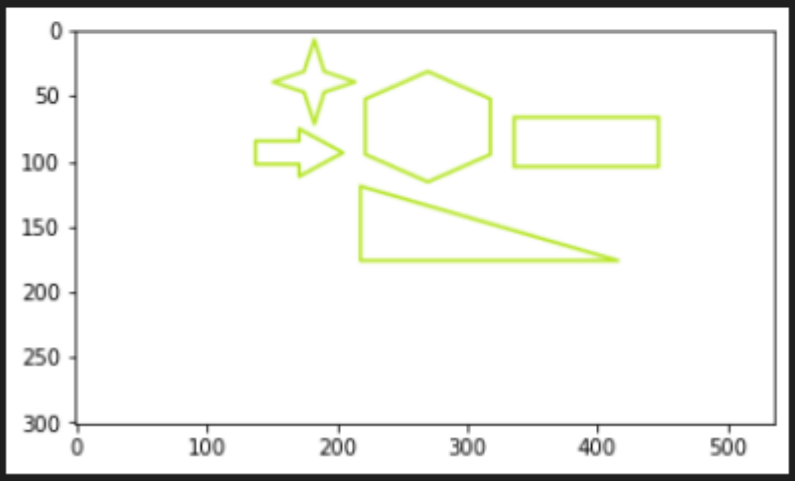
Shi-Tomasi Corner Detection
Harris Corner Detection

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
%matplotlib inline
```

```
image = cv2.imread('shapes.png')
image_copy = np.copy(image)
image_copy = cv2.cvtColor(image_copy, cv2.COLOR_BGR2RGB)
```

```
plt.imshow(image_copy)
```

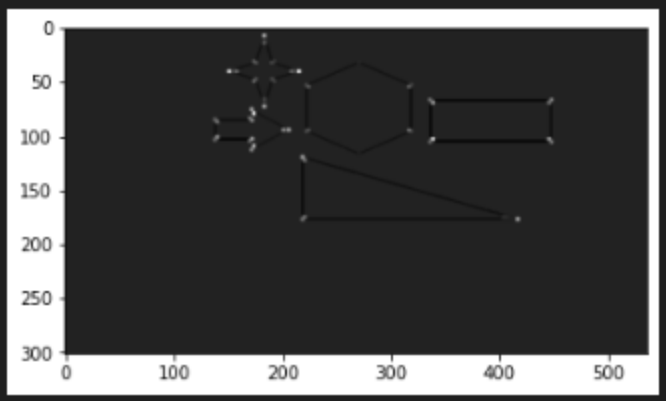
```
<matplotlib.image.AxesImage at 0x270fe4525b0>
```



```
gray = cv2.cvtColor(image_copy, cv2.COLOR_RGB2GRAY)
gray = np.float32(gray)
dst = cv2.cornerHarris(gray, 2, 3, 0.04)
dst = cv2.dilate(dst, None)
```

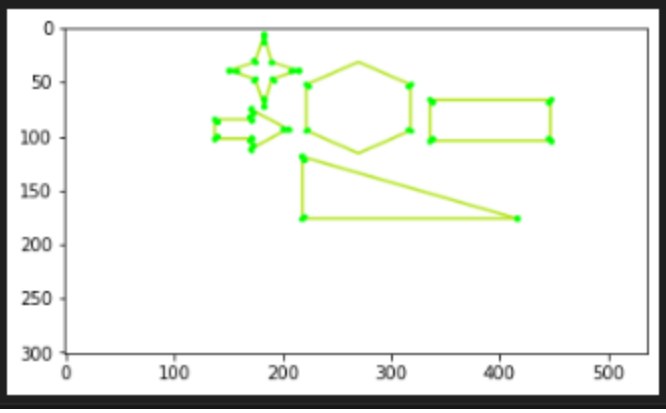
```
plt.imshow(dst, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x270fe4ab250>
```




```
thresh = 0.1*dst.max()
corner_image = np.copy(image_copy)
for j in range(0, dst.shape[0]):
    for i in range(0, dst.shape[1]):
        if(dst[j,i] > thresh):
            cv2.circle(corner_image,(i,j),1,(0,255,0),
plt.imshow(corner_image)
```

```
<matplotlib.image.AxesImage at 0x270fe500a30>
```



Smile Detection

```python
import cv2


image = cv2.imread("smile.jfif")
smile_cascade = cv2.CascadeClassifier('haarcascade_smile.xml')
smiles  = smile_cascade.detectMultiScale(image, scaleFactor = 1.8, minNeighbors = 20)


for (sx, sy, sw, sh) in smiles:
    cv2.rectangle(image, (sx, sy), ((sx + sw), (sy + sh)), (0, 255,0), 5)


cv2.imshow("Smile Detected", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```