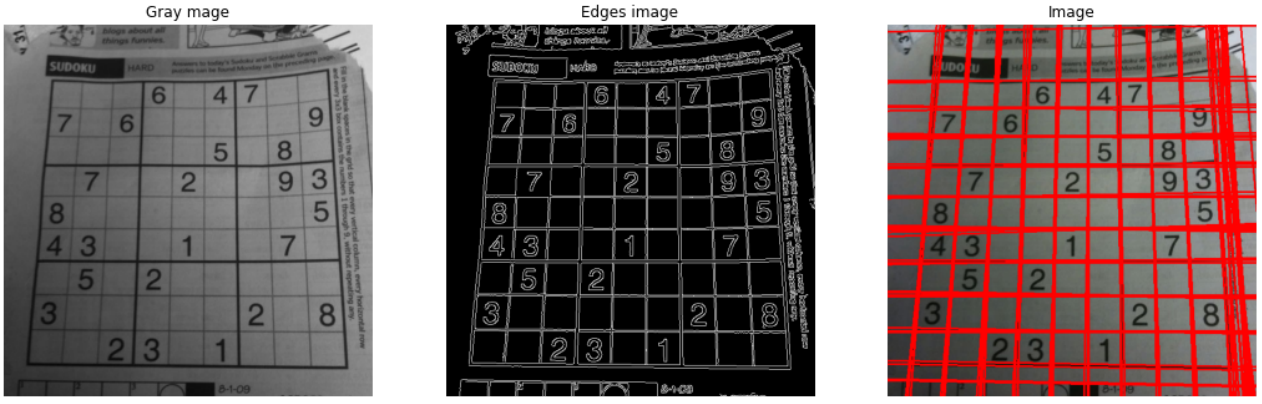
NAME : M.N.F. NIFLA

INDEX NUMBER : 190413D

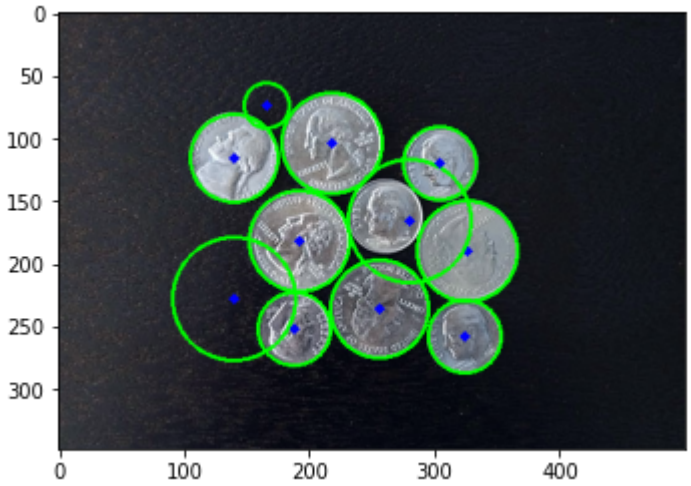
In [ ]:
```
import cv2 as cv
import numpy as np

im = cv.imread('images/sudoku.png', cv.IMREAD_COLOR)
assert im is not None

gray = cv.cvtColor(im, cv.COLOR_BGR2GRAY)
edges = cv.Canny(gray, 20, 120, apertureSize = 3)
lines = cv.HoughLines(edges, 1, np.pi/180, 150)

for line in lines:
    rho, theta = line[0]
    a = np.cos(theta)
    b = np.sin(theta)
    x0, y0 = a*rho, b*rho
    x1, y1 = int(x0 + 1000*(-b)), int(y0 + 1000*(a))
    x2, y2 = int(x0 - 1000*(-b)), int(y0 - 1000*(a))
    cv.line(im, (x1, y1), (x2, y2), (0, 0, 255), 2)

cv.namedWindow('Image', cv.WINDOW_NORMAL)
cv.imshow("Image", gray)
cv.waitKey()
cv.imshow("image", edges)
cv.waitKey()
cv.imshow("Image", im)
cv.waitKey()
cv.destroyAllWindows()

f,ax=plt.subplots(1,3,figsize=(18,6))
ax[0].imshow(cv.cvtColor(gray, cv.COLOR_BGR2RGB))
ax[0].set_title("Gray mage")
ax[0].axis('off')
ax[1].imshow(cv.cvtColor(edges, cv.COLOR_BGR2RGB))
ax[1].set_title("Edges image")
ax[1].axis('off')
ax[2].imshow(cv.cvtColor(im, cv.COLOR_BGR2RGB))
ax[2].set_title("Image")
ax[2].axis('off')
plt.show()
```



In [ ]:
```
import cv2 as cv
```

```
import numpy as np

im = cv.imread(r'./images/coins.jpg', cv.IMREAD_COLOR)
assert im is not None
gray = cv.cvtColor(im, cv.COLOR_BGR2GRAY)

circles = cv.HoughCircles(gray, cv.HOUGH_GRADIENT, 1, 50, param1=150, param2=20, minRad
circles = np.uint16(np.around(circles))

for i in circles[0, :]:
    #draw the outer circle
    cv.circle(im, (i[0],i[1]),i[2],(0,255,0),2)
    #draw the center of the circle
    cv.circle(im, (i[0],i[1]),2,(0,0,255),3)
cv.imshow('Detected Circles', im)
cv.waitKey(0)
cv.destroyAllWindows()

plt.imshow(im)
```

Out[ ]: <matplotlib.image.AxesImage at 0x19fdddce380>

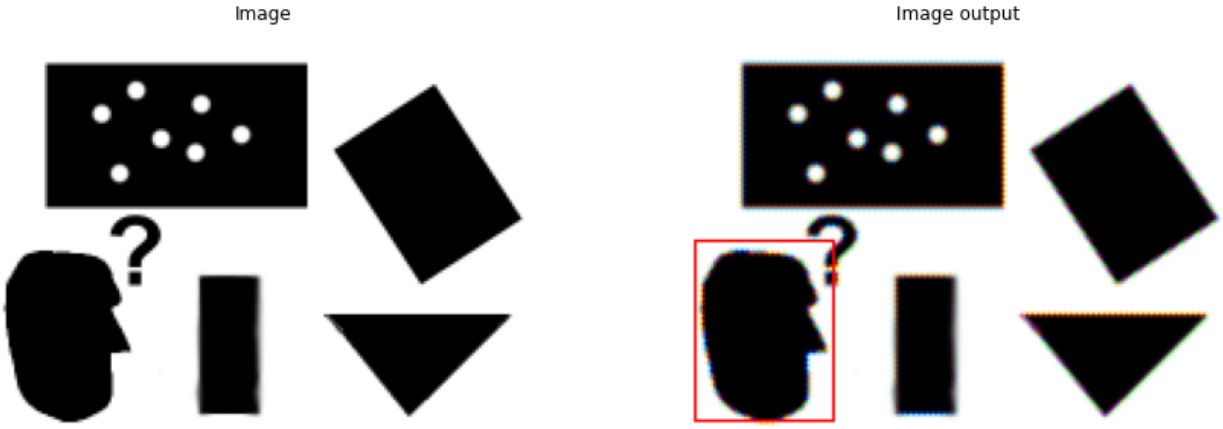


In [ ]:

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

img=cv.imread('images/pic1.png',cv.IMREAD_REDUCED_GRAYSCALE_2)
assert img is not None
temp=cv.imread('images/templ.png',cv.IMREAD_REDUCED_GRAYSCALE_2)
assert temp is not None

#Canny edge detection
im_edges= cv.Canny(img, 50, 250)
templ_edges = cv.Canny(temp, 50, 250)

alg = cv.createGeneralizedHoughGuil()
alg.setTemplate(templ_edges)
#Vote thresholds
alg.setAngleThresh(100000)
alg.setScaleThresh(40000)
alg.setPosThresh(1000)

alg.setAngleStep(1)
```

```
alg.setScaleStep(0.1)
alg.setMinScale(0.9)
alg.setMaxScale(1.1)

positions, votes = alg.detect(im_edges)

out = cv.cvtColor(img, cv.COLOR_BAYER_BG2BGR)
for x, y, scale, orientation in positions[0]:
    halfHeight = temp.shape[0] / 2. *scale
    halfWidth = temp.shape[1] / 2. *scale
    p1 = (int(x- halfWidth), int(y - halfHeight))
    p2 = (int(x + halfWidth), int(y + halfHeight))
    print("x = {}, y = {}, scale = {}, orientation= {}, p1= {}, p2 = {}".format(x, y,sc
    cv.rectangle(out, p1, p2, (0, 0, 255))

cv.imshow('Image', img)
cv.waitKey(0)
cv.imshow('Image', out)
cv.waitKey(0)
cv.destroyAllWindows()

f,ax=plt.subplots(1,2,figsize=(15,5))
ax[0].imshow(cv.cvtColor(img, cv.COLOR_BGR2RGB))
ax[0].set_title("Image")
ax[0].axis('off')
ax[1].imshow(cv.cvtColor(out, cv.COLOR_BGR2RGB))
ax[1].set_title("Image output")
ax[1].axis('off')
plt.show()
```

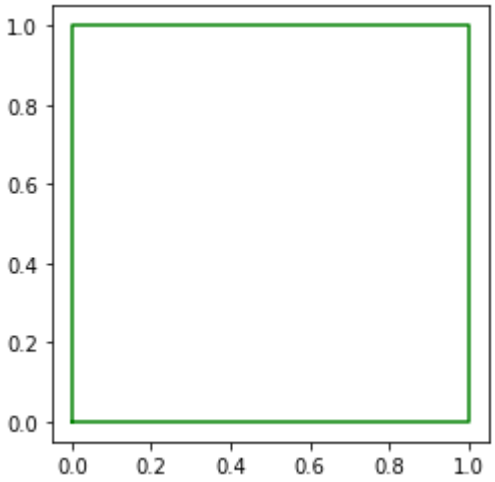
```
x = 29.0, y = 109.0, scale = 1.0, orientation= 0.0, p1= (4, 76), p2 = (54, 141)
```



In [ ]:

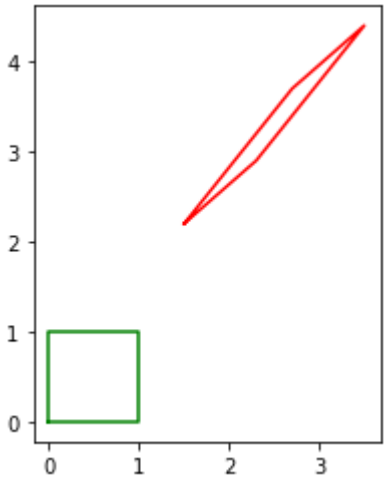
```
import matplotlib.pyplot as plt
import numpy as np

a, b, c, d= [0,0,1], [0,1,1], [1,1,1] , [1,0,1]
X =np.array([a,b,c,d]).T

theta = np.pi*30/180
s = 2
tx ,ty = 2, 3
#H = np.array([[s*np.cos(theta), s*np.sin(theta), tx],[s*np.sin(theta), s*np.cos(theta)
#Y = H@X
a11, a12, a21 , a22 = 0.8, 1.2, 0.7, 1.5
A= np.array([[a11, a12, tx], [a21, a22, ty],[]])
x = np.append(X[0,:],X[0,0])
```

```
y = np.append(X[1, :], X[1,0])
fig ,ax = plt.subplots(1,1)
ax.plot(x, y, color='g')
ax.set_aspect('equal')
plt.show()
```



In [ ]:

```
a, b, c, d = [0, 0, 1], [0, 1, 1], [1,1,1], [1,0,1]
X = np.array([a,b,c,d]).T

theta = np.pi*30/180
s = 1
tx, ty = 1.5, 2.2
# H = np.array([[s*np.cos(theta), -s*np.sin(theta), tx], [s*np.sin(theta), s*np.cos(the
# Y = H @ X

a11, a12, a21, a22 = 0.8, 1.2, 0.7, 1.5 #Should be a non-singular matrix here
A = np.array([[a11,a12,tx], [a21, a22, ty], [0,0,1]])
Y = A @ X

x = np.append(X[0, :], X[0, 0])
y = np.append(X[1, :], X[1, 0])
fig, ax = plt.subplots(1,1)
ax.plot(x, y, color='g')
ax.set_aspect('equal')

x = np.append(Y[0, :], Y[0, 0])
y = np.append(Y[1, :], Y[1, 0])
ax.plot(x, y, color='r')
ax.set_aspect('equal')
plt.show()
```



In [ ]:

```
# Warping usign teh given homography
import matplotlib.pyplot as plt
import numpy as np
import cv2 as cv

im1 = cv.imread('images/graf/img1.ppm', cv.IMREAD_ANYCOLOR)
im4 = cv.imread('images/graf/img4.ppm', cv.IMREAD_ANYCOLOR)

#H = np.array([[ 6.6378505e-01, 6.8003334e-01, -3.1230335e+01]])
H=[]

with open('images/graf/H1to4p') as f:
    H = np.array([[float(h) for h in line.split()] for line in f])

im1to4 = cv.warpPerspective(im4, np.linalg.inv(H), (2000, 2000))

cv.namedWindow('Image 1', cv.WINDOW_AUTOSIZE)
cv.imshow('Image 1', im1)
cv.waitKey(0)
cv.namedWindow('Image 2', cv.WINDOW_AUTOSIZE)
cv.imshow('Image 2', im4)
cv.waitKey(0)
cv.namedWindow('Image 1 warped', cv.WINDOW_AUTOSIZE)
cv.imshow('Image 1 warped', im1to4)
cv.waitKey(0)
cv.destroyAllWindows()

f,ax=plt.subplots(1,3,figsize=(18,6))
ax[0].imshow(cv.cvtColor(im1, cv.COLOR_BGR2RGB))
ax[0].set_title("Image 1")
ax[0].axis('off')
ax[1].imshow(cv.cvtColor(im4, cv.COLOR_BGR2RGB))
ax[1].set_title("Image 4")
ax[1].axis('off')
ax[2].imshow(cv.cvtColor(im1to4, cv.COLOR_BGR2RGB))
ax[2].set_title("Image 1 to 4")
ax[2].axis('off')
plt.show()
```

Image 1

Image 4

Image 1 to 4