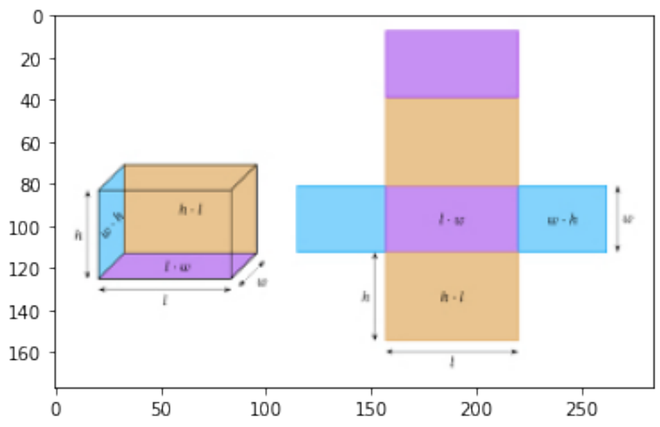
In [1]:
```
import numpy as np
import cv2
import matplotlib.pyplot as plt
import math
```

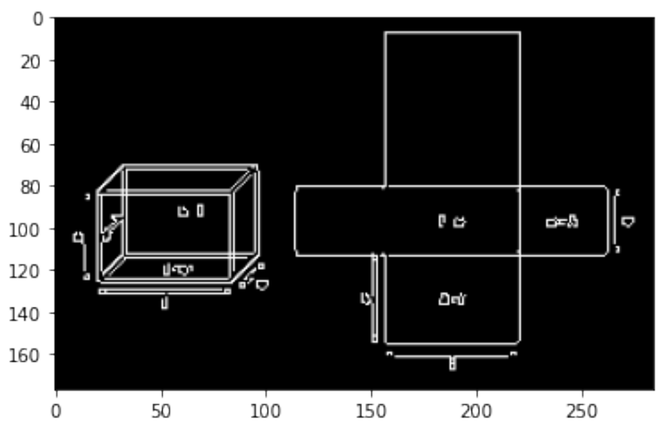
In [2]:
```
img = cv2.imread("../images/rectangle2.png")
plt.imshow(img, cmap= "gray")
```

Out[2]:
```
<matplotlib.image.AxesImage at 0x7ff680b0f250>
```



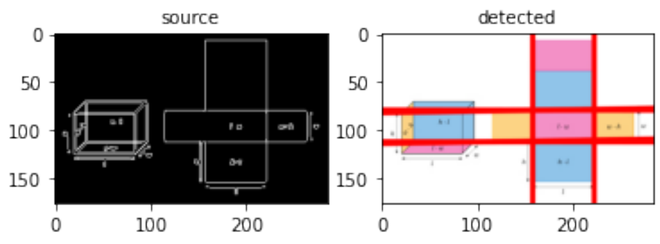
In [3]:
```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
dst = cv2.Canny(gray, 50, 200, None, 3)
plt.imshow(dst, cmap= "gray")
```

Out[3]:
```
<matplotlib.image.AxesImage at 0x7ff680ccb370>
```



In [6]:
```
dstp = np.copy(img)
lines = cv2.HoughLines(dst, 1, np.pi/ 180, 90, None, 0, 0)

if lines is not None:
    for i in range(0, len(lines)):
        rho = lines[i][0][0]
        theta = lines[i][0][1]
        a = math.cos(theta)
        b = math.sin(theta)
        x0 = a*rho
        y0 = b* rho
        point1 = (int(x0+500*(-b)), int(y0+500*(a)))
        point2 = (int(x0-500*(-b)), int(y0-500*(a)))
        cv2.line(dstp, point1, point2, (0,0,255), 3, cv2.LINE_AA) # antialiasing

fig, axs = plt.subplots(1,2)
axs[0].imshow(cv2.cvtColor(dst, cv2.COLOR_BGR2RGB))
axs[0].set_title('source', fontsize=10)

axs[1].imshow(cv2.cvtColor(dstp, cv2.COLOR_BGR2RGB))
axs[1].set_title('detected', fontsize=10)
```

Out[6]:
```
Text(0.5, 1.0, 'detected')
```

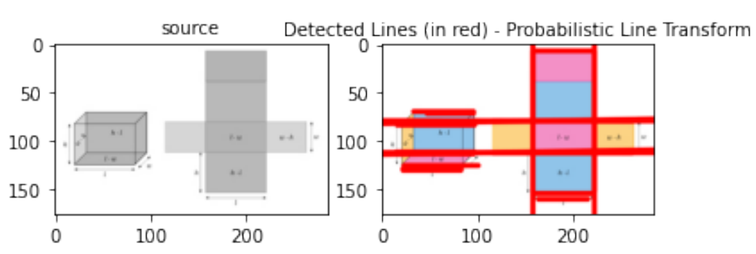


In [9]:
```
lines.shape
```

Out[9]:
```
(6, 1, 2)
```

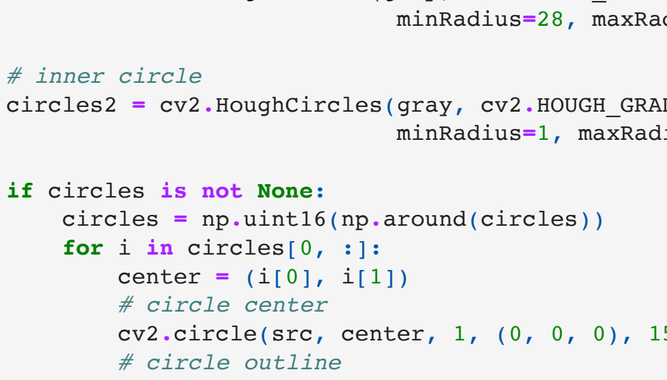
In [7]:
```
linesP = cv2.HoughLinesP(dst, 1, np.pi / 180, 50, None, 50, 10)
if linesP is not None:
    for i in range(0, len(linesP)):
        l = linesP[i][0]
        cv2.line(dstp, (l[0], l[1]), (l[2], l[3]), (0,0,255), 3, cv2.LINE_AA)

fig, axs = plt.subplots(1,2)
axs[0].imshow(cv2.cvtColor(gray, cv2.COLOR_BGR2RGB))
axs[0].set_title('source', fontsize=10)

axs[1].imshow(cv2.cvtColor(dstp, cv2.COLOR_BGR2RGB))
axs[1].set_title('Detected Lines (in red) - Probabilistic Line Transform', fontsize=10)
```

Out[7]:
```
Text(0.5, 1.0, 'Detected Lines (in red) - Probabilistic Line Transform')
```



In [8]:
```
src = cv2.imread('../images/Hough_circle.png')
src_original= np.copy(src)
gray = cv2.cvtColor(src,cv2.COLOR_BGR2GRAY)
rows = gray.shape[0]

# outer circle
circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1, rows / 8, param1=100, param2=30,
                           minRadius=28, maxRadius=100)

# inner circle
circles2 = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1, rows / 8, param1=100, param2=30,
                           minRadius=1, maxRadius=60)

if circles is not None:
    circles = np.uint16(np.around(circles))
    for i in circles[0, :]:
        center = (i[0], i[1])
        # circle center
        cv2.circle(src, center, 1, (0, 0, 0), 15)
        # circle outline
        radius = i[2]
        cv2.circle(src, center, radius, (0, 0, 0), 15)

if circles2 is not None:
    circles2 = np.uint16(np.around(circles2))
    for i in circles2[0, :]:
        center = (i[0], i[1])
        # circle center
        cv2.circle(src, center, 1, (0, 0, 0), 15)
        # circle outline
        radius = i[2]
        cv2.circle(src, center, radius, (0, 0, 0), 15)

fig, axs = plt.subplots(1,2)
axs[0].imshow(cv2.cvtColor(src_original, cv2.COLOR_BGR2RGB))
axs[0].set_title('original', fontsize=10)

axs[1].imshow(cv2.cvtColor(src, cv2.COLOR_BGR2RGB))
axs[1].set_title('detected circles', fontsize=10)
```

Out[8]:
```
Text(0.5, 1.0, 'detected circles')
```



In [ ]:
```
```