Project 5

ID：0811562 name：何祁恩

(a) Source codes:

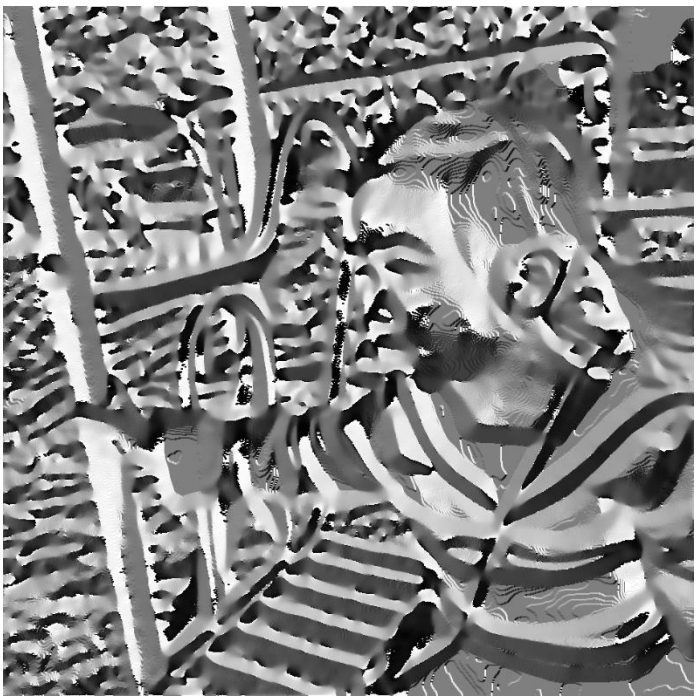
```python
import cv2 as cv
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
from google.colab.patches import cv2_imshow
from scipy import ndimage

def save_picture(filename, source):
  image = Image.fromarray(np.uint8(source))
  image.save('./'+filename, dpi=(200, 200))

#readfile and perform gaussian blur
img = cv.imread('Kid at playground.tif', cv.IMREAD_GRAYSCALE)
blur = cv.GaussianBlur(img, (29, 29), min(img.shape[0], img.shape[1] * 0.005))
cv2_imshow(img)
cv2_imshow(blur)
print(img.shape[1] * 0.005)

#Sobel gradient
gx = cv.Sobel(np.float32(blur), cv.CV_32F, 1, 0, 3)
gy = cv.Sobel(np.float32(blur), cv.CV_32F, 0, 1, 3)

G = cv.addWeighted(np.abs(gx), 0.5, np.abs(gy),0.5,0)
G = G / G.max() * 255

theta = np.rad2deg(np.arctan2(gy, gx))

cv2_imshow(G)
cv2_imshow((theta + 180) * 255 / 360)

save_picture('Gradient_Magnitude.png', G)
save_picture('Gradient_Angle.png', (theta + 180) * 255 / 360)
```

```python
#perform nonmaxima suppression

size = G.shape
gN = np.zeros(size)
theta[theta < 0] += 180

for i in range(1, size[0] - 1):
    for j in range(1, size[1] - 1):

        # d1 : horizontal edge
        if (0 <= theta[i, j] < 22.5) or (157.5 <= theta[i, j] <= 180):
            temp = max(G[i, j - 1], G[i, j + 1])

        # d2 : -45 deg edge
        elif (22.5 <= theta[i, j] < 67.5):
            temp = max(G[i - 1, j - 1], G[i + 1, j + 1])

        # d3 :  vertical edge
        elif (67.5 <= theta[i, j] < 112.5):
            temp = max(G[i - 1, j], G[i + 1, j])

        # d4 : +45 deg edge
        else:
            temp = max(G[i + 1, j - 1], G[i - 1, j + 1])

        if G[i, j] >= temp:
            gN[i, j] = G[i, j]

gN = np.multiply(gN, 255.0 / gN.max())
cv2_imshow(gN)
save_picture('Non-MaximaSuppressedImage.png', gN)
```

```
65
66  #get weak and strong
67  gN = gN / 255
68  size = gN.shape
69  gNL = np.zeros(size)
70  gNH = np.zeros(size)
71
72  TL = 0.04
73  TH = 0.1
74
75  strong_x, strong_y = np.where(gN >= TH)
76  weak_index = np.zeros(size)
77
78  for i in range(size[0]):
79      for j in range(size[1]):
80          if gN[i,j] >= TH:
81              gNH[i,j] = gN[i,j]
82
83          if (TL <= gN[i,j]) and (gN[i,j] < TH):
84              gNL[i,j] = gN[i,j]
85              weak_index[i, j] = 1
86
87  gNL = np.multiply(gNL, 255.0 / gNL.max())
88  gNH = np.multiply(gNH, 255.0 / gNH.max())
89
90  cv2_imshow(gNL)
91  cv2_imshow(gNH)
92  save_picture('gNL.png', gNL)
93  save_picture('gNH.png', gNH)
```

```
94
95  #perform hysterisis thresholding
96  size = gN.shape
97  e = np.zeros(size)
98
99  while(len(strong_x) and len(strong_y) > 0):
100     a = strong_x[0]
101     b = strong_y[0]
102     strong_x = np.delete(strong_x, 0)
103     strong_y = np.delete(strong_y, 0)
104     e[a, b] = 1
105
106     for i in range(-1,2):
107         for j in range(-1, 2):
108             if i == 0 and j == 0:
109                 continue
110             else:
111                 new_a = a + i
112                 new_b = b + j
113                 if((new_a >= 0 & new_a < size[0] & new_b >= 0 & new_b < size[1]) and (weak_index[new_a, new_b] == 1)):
114                     e[new_a, new_b] = 1
115
116 e = np.multiply(e, 255.0 / e.max())
117 cv2_imshow(e)
118 save_picture('e.png', e)
```

(b) Plot images of the gradient magnitude and gradient angle:

magnitude	angle
	

© Plot □onmaximal suppressed image g_N (x,y) as well as images of g_{NL}(x,y) and g_{NH}(x,y):

g_N (x,y)	g_{NL}(x,y)
	

$g_{NH}(x,y)$

(d) Plot final edge map e(x,y):