HW9
r10944013 網媒所碩一 馮啟倫

HW9-a:

演算法: Robert's Operator, 依照PPT上面寫kernel然後做convolution（有先做padding），
Threshold為30

Code segment and result picture

```python
def robert_operator(img):
    threshold = 30
    img_pad = np.pad(img,((0,1),(0,1)), 'edge').astype('int')
    img_new = np.zeros((512,512))
    for row in range(img.shape[0]):
        for col in range(img.shape[1]):
            x0 = col
            y0 = row
            x1 = x0+1
            y1 = y0+1
            r1 = int(img_pad[y1][x1]) - int(img_pad[y0][x0])
            r2 = img_pad[y1][x0] - img_pad[y0][x1]
            gradient = (r1**2 + r2**2)**(1/2)
            if gradient >= threshold:
                img_new[row][col] = 0
            else:
                img_new[row][col] = 255
    return img_new
```

HW9-b:

演算法: Prewitt's Edge Detector 依照PPT上面寫kernel然後做convolution（有先做padding），
Threshold為24

Code segment and result picture

```python
def prewitt_operator(img):
    threshold = 24
    img_pad = np.pad(img,((1,1),(1,1)), 'edge').astype('int')
    img_new = np.zeros((512,512))
    for row in range(img.shape[0]):
        for col in range(img.shape[1]):
            x_center = col+1
            x_back = x_center-1
            x_front = x_center+1
            y_center = row+1
            y_back = y_center-1
            y_front = y_center+1
            p1 = (img_pad[y_front][x_back] + img_pad[y_front][x_center] + img_pad[y_front][x_front]) - (img_pad[y_back][x_back] + img_pad[y_back][x_center] + img_pad[y_back][x_front])
            p2 = (img_pad[y_back][x_front] + img_pad[y_center][x_front] + img_pad[y_front][x_front]) - (img_pad[y_back][x_back] + img_pad[y_center][x_back] + img_pad[y_front][x_back])
            gradient = (p1**2 + p2**2)**(1/2)
            if gradient >= threshold:
                img_new[row][col] = 0
            else:
                img_new[row][col] = 255
    return img_new
```

HW9-c:

演算法: Sobel's Edge Detector 依照PPT上面寫kernel然後做convolution（有先做padding），**Threshold為38**

Code segment and result picture

```
def sobel_operator(img):
    threshold = 38
    img_pad = np.pad(img,((1,1),(1,1)), 'edge').astype('int')
    img_new = np.zeros((512,512))
    for row in range(img.shape[0]):
        for col in range(img.shape[1]):
            x_center = col+1
            x_back = x_center-1
            x_front = x_center+1
            y_center = row+1
            y_back = y_center-1
            y_front = y_center+1
            s1 = (img_pad[y_front][x_back] + 2*img_pad[y_front][x_center] + img_pad[y_front][x_front]) - (img_pad[y_back][x_back] + 2*img_pad[y_back][x_center] + img
            s2 = (img_pad[y_back][x_front] + 2*img_pad[y_center][x_front] + img_pad[y_front][x_front]) - (img_pad[y_back][x_back] + 2*img_pad[y_center][x_back] + img
            gradient = (s1**2 + s2**2)**(1/2)
            if gradient >= threshold:
                img_new[row][col] = 0
            else:
                img_new[row][col] = 255
    return img_new
```

HW9-d:

演算法:Frei and Chen's Gradient Operator 依照PPT上面寫kernel然後做convolution（有先做padding），**Threshold為30**

Code segment and result picture

```
def FreiChen_operator(img):
    threshold = 30
    img_pad = np.pad(img,((1,1),(1,1)), 'edge').astype('int')
    img_new = np.zeros((512,512))
    for row in range(img.shape[0]):
        for col in range(img.shape[1]):
            x_center = col+1
            x_back = x_center-1
            x_front = x_center+1
            y_center = row+1
            y_back = y_center-1
            y_front = y_center+1
            s1 = (img_pad[y_front][x_back] + (2**(1/2))*img_pad[y_front][x_center] + img_pad[y_front][x_front]) - (img_pad[y_back][x_back] + (2**(1/2))*img_pad[y_back][x_center
            s2 = (img_pad[y_back][x_front] + (2**(1/2))*img_pad[y_center][x_front] + img_pad[y_front][x_front]) - (img_pad[y_back][x_back] + (2**(1/2))*img_pad[y_center][x_back
            gradient = (s1**2 + s2**2)**(1/2)
            if gradient >= threshold:
                img_new[row][col] = 0
            else:
                img_new[row][col] = 255
    return img_new
```

HW9-e:

演算法:Kirsch's Compass Operator 依照PPT上面寫kernel然後做convolution（有先做padding），**Threshold為135**

Code segment and result picture

```
def Kirsch_operator(img):
    threshold = 135
    img_pad = np.pad(img,((1,1),(1,1)), 'edge').astype('int')
    img_new = np.zeros((512,512))

    for row in range(img.shape[0]):
        for col in range(img.shape[1]):
            x_center = col+1
            x_back = x_center-1
            x_front = x_center+1
            y_center = row+1
            y_back = y_center-1
            y_front = y_center+1
            k0 = -3*img_pad[y_back][x_back] + -3*img_pad[y_back][x_center] + 5*img_pad[y_back][x_front] \
                 + -3*img_pad[y_center][x_back] + 5*img_pad[y_center][x_front] \
                 + -3*img_pad[y_front][x_back] + -3*img_pad[y_front][x_center] + 5*img_pad[y_front][x_front]
            k1 = -3*img_pad[y_back][x_back] + 5*img_pad[y_back][x_center] + 5*img_pad[y_back][x_front] \
                 + -3*img_pad[y_center][x_back] + 5*img_pad[y_center][x_front] \
                 + -3*img_pad[y_front][x_back] + -3*img_pad[y_front][x_center] + -3*img_pad[y_front][x_front]
            k2 = 5*img_pad[y_back][x_back] + 5*img_pad[y_back][x_center] + 5*img_pad[y_back][x_front] \
                 + -3*img_pad[y_center][x_back] + -3*img_pad[y_center][x_front] \
                 + -3*img_pad[y_front][x_back] + -3*img_pad[y_front][x_center] + -3*img_pad[y_front][x_front]
            k3 = 5*img_pad[y_back][x_back] + 5*img_pad[y_back][x_center] + -3*img_pad[y_back][x_front] \
                 + 5*img_pad[y_center][x_back] + -3*img_pad[y_center][x_front] \
                 + -3*img_pad[y_front][x_back] + -3*img_pad[y_front][x_center] + -3*img_pad[y_front][x_front]
            k4 = 5*img_pad[y_back][x_back] + -3*img_pad[y_back][x_center] + -3*img_pad[y_back][x_front] \
                 + 5*img_pad[y_center][x_back] + -3*img_pad[y_center][x_front] \
                 + 5*img_pad[y_front][x_back] + -3*img_pad[y_front][x_center] + -3*img_pad[y_front][x_front]
            k5 = -3*img_pad[y_back][x_back] + -3*img_pad[y_back][x_center] + -3*img_pad[y_back][x_front] \
                 + 5*img_pad[y_center][x_back] + -3*img_pad[y_center][x_front] \
                 + 5*img_pad[y_front][x_back] + 5*img_pad[y_front][x_center] + -3*img_pad[y_front][x_front]
            k6 = -3*img_pad[y_back][x_back] + -3*img_pad[y_back][x_center] + -3*img_pad[y_back][x_front] \
                 + -3*img_pad[y_center][x_back] + -3*img_pad[y_center][x_front] \
                 + 5*img_pad[y_front][x_back] + 5*img_pad[y_front][x_center] + 5*img_pad[y_front][x_front]
            k7 = -3*img_pad[y_back][x_back] + -3*img_pad[y_back][x_center] + -3*img_pad[y_back][x_front] \
                 + -3*img_pad[y_center][x_back] + 5*img_pad[y_center][x_front] \
                 + -3*img_pad[y_front][x_back] + 5*img_pad[y_front][x_center] + 5*img_pad[y_front][x_front]
```

HW9-f:

演算法:Robinson's Compass Operator 依照PPT上面寫kernel然後做convolution（有先做padding），**Threshold為43**

Code segment and result picture

```
def Robinson_operator(img):
    threshold = 43
    img_pad = np.pad(img,((1,1),(1,1)), 'edge').astype('int')
    img_new = np.zeros((512,512))

    for row in range(img.shape[0]):
        for col in range(img.shape[1]):
            x_center = col+1
            x_back = x_center-1
            x_front = x_center+1
            y_center = row+1
            y_back = y_center-1
            y_front = y_center+1
            k0 = -1*img_pad[y_back][x_back]  + 1*img_pad[y_back][x_front] \
                 + -2*img_pad[y_center][x_back] + 2*img_pad[y_center][x_front] \
                 + -1*img_pad[y_front][x_back] + 1*img_pad[y_front][x_front]

            k1 = 1*img_pad[y_back][x_center] + 2*img_pad[y_back][x_front] \
                 + -1*img_pad[y_center][x_back] + 1*img_pad[y_center][x_front] \
                 + -2*img_pad[y_front][x_back] + -1*img_pad[y_front][x_center]

            k2 = 1*img_pad[y_back][x_back] + 2*img_pad[y_back][x_center] + 1*img_pad[y_back][x_front] \
                 + -1*img_pad[y_front][x_back] + -2*img_pad[y_front][x_center] + -1*img_pad[y_front][x_front]

            k3 = 2*img_pad[y_back][x_back] + 1*img_pad[y_back][x_center] \
                 + 1*img_pad[y_center][x_back] + -1*img_pad[y_center][x_front] \
                 + -1*img_pad[y_front][x_center] + -2*img_pad[y_front][x_front]

            k4 = 1*img_pad[y_back][x_back] + -1*img_pad[y_back][x_front] \
                 + 2*img_pad[y_center][x_back] + -2*img_pad[y_center][x_front] \
                 + 1*img_pad[y_front][x_back] + -1*img_pad[y_front][x_front]

            k5 = -1*img_pad[y_back][x_center] + -2*img_pad[y_back][x_front] \
                 + 1*img_pad[y_center][x_back] + -1*img_pad[y_center][x_front] \
                 + 2*img_pad[y_front][x_back] + 1*img_pad[y_front][x_center]

            k6 = -1*img_pad[y_back][x_back] + -2*img_pad[y_back][x_center] + -1*img_pad[y_back][x_front] \
                 + 1*img_pad[y_front][x_back] + 2*img_pad[y_front][x_center] + 1*img_pad[y_front][x_front]
            k7 = -2*img_pad[y_back][x_back] + -1*img_pad[y_back][x_center]  \
                 + -1*img_pad[y_center][x_back] + 1*img_pad[y_center][x_front] \
                 + 1*img_pad[y_front][x_center] + 2*img_pad[y_front][x_front]
            gradient = np.max([k0,k1,k2,k3,k4,k5,k6,k7])
```

HW9-g:

演算法:Nevatia-Babu 5x5 Operator依照PPT上面寫kernel然後做convolution（有先做padding），
Threshold為12500

Code segment and result picture

```
x_center = col+2
x_back = x_center-1
x_back_two = x_center-2
x_front = x_center+1
x_front_two = x_center + 2

y_center = row+2
y_back = y_center-1
y_back_two = y_center-2
y_front = y_center+1
y_front_two = y_center+2

k0 = 100*img_pad[y_back_two][x_back_two]+100*img_pad[y_back_two][x_back]+100*img_pad[y_back_two][x_center]+100*img_pad[y_back_two][x_front]+100*img_pad[y_back_two][x_fro
     +100*img_pad[y_back][x_back_two]+100*img_pad[y_back][x_back]+100*img_pad[y_back][x_center]+100*img_pad[y_back][x_front]+100*img_pad[y_back][x_front_two] \
     + -100*img_pad[y_front][x_back_two]+ -100*img_pad[y_front][x_back]+ -100*img_pad[y_front][x_center]+ -100*img_pad[y_front][x_front]+ -100*img_pad[y_front][x_front_t
     + -100*img_pad[y_front_two][x_back_two]+ -100*img_pad[y_front_two][x_back]+ -100*img_pad[y_front_two][x_center]+ -100*img_pad[y_front_two][x_front]+ -100*img_pad[y_


k1 = 100*img_pad[y_back_two][x_back_two]+100*img_pad[y_back_two][x_back]+100*img_pad[y_back_two][x_center]+100*img_pad[y_back_two][x_front]+100*img_pad[y_back_two][x_fro
     +100*img_pad[y_back][x_back_two]+100*img_pad[y_back][x_back]+100*img_pad[y_back][x_center]+78*img_pad[y_back][x_front]+ -32*img_pad[y_back][x_front_two] \
     +100*img_pad[y_center][x_back_two]+92*img_pad[y_center][x_back]+ -92*img_pad[y_center][x_front]+ -100*img_pad[y_center][x_front_two] \
     + 32*img_pad[y_front][x_back_two]+ -78*img_pad[y_front][x_back]+ -100*img_pad[y_front][x_center]+ -100*img_pad[y_front][x_front]+ -100*img_pad[y_front][x_front_two]
     + -100*img_pad[y_front_two][x_back_two]+ -100*img_pad[y_front_two][x_back]+ -100*img_pad[y_front_two][x_center]+ -100*img_pad[y_front_two][x_front]+ -100*img_pad[y_

k2 = 100*img_pad[y_back_two][x_back_two]+100*img_pad[y_back_two][x_back]+100*img_pad[y_back_two][x_center]+32*img_pad[y_back_two][x_front]+ -100*img_pad[y_back_two][x_fr
     +100*img_pad[y_back][x_back_two]+100*img_pad[y_back][x_back]+92*img_pad[y_back][x_center]+ -78*img_pad[y_back][x_front]+ -100*img_pad[y_back][x_front_two] \
     +100*img_pad[y_center][x_back_two]+100*img_pad[y_center][x_back]+ -100*img_pad[y_center][x_front]+ -100*img_pad[y_center][x_front_two] \
     + 100*img_pad[y_front][x_back_two]+ 78*img_pad[y_front][x_back]+ -92*img_pad[y_front][x_center]+ -100*img_pad[y_front][x_front]+ -100*img_pad[y_front][x_front_two]
     + 100*img_pad[y_front_two][x_back_two]+ -32*img_pad[y_front_two][x_back]+ -100*img_pad[y_front_two][x_center]+ -100*img_pad[y_front_two][x_front]+ -100*img_pad[y_fr

k3 = -100*img_pad[y_back_two][x_back_two]+ -100*img_pad[y_back_two][x_back]+ 100*img_pad[y_back_two][x_front]+100*img_pad[y_back_two][x_front_two] \
     + -100*img_pad[y_back][x_back_two]+ -100*img_pad[y_back][x_back]+ 100*img_pad[y_back][x_front]+ 100*img_pad[y_back][x_front_two] \
     + -100*img_pad[y_center][x_back_two]+ -100*img_pad[y_center][x_back]+ 100*img_pad[y_center][x_front]+ 100*img_pad[y_center][x_front_two] \
     + -100*img_pad[y_front][x_back_two]+ -100*img_pad[y_front][x_back]+ 100*img_pad[y_front][x_front]+ 100*img_pad[y_front][x_front_two] \
     + -100*img_pad[y_front_two][x_back_two]+ -100*img_pad[y_front_two][x_back]+ 100*img_pad[y_front_two][x_front]+ 100*img_pad[y_front_two][x_front_two]

k4 = -100*img_pad[y_back_two][x_back_two]+32*img_pad[y_back_two][x_back]+100*img_pad[y_back_two][x_center]+100*img_pad[y_back_two][x_front]+100*img_pad[y_back_two][x_fro
     + -100*img_pad[y_back][x_back_two]+ -78*img_pad[y_back][x_back]+ 92*img_pad[y_back][x_center]+ 100*img_pad[y_back][x_front]+ 100*img_pad[y_back][x_front_two] \
     + -100*img_pad[y_center][x_back_two]+ -100*img_pad[y_center][x_back]+ 100*img_pad[y_center][x_front]+ 100*img_pad[y_center][x_front_two] \
     + -100*img_pad[y_front][x_back_two]+ -100*img_pad[y_front][x_back]+ -92*img_pad[y_front][x_center]+ 78*img_pad[y_front][x_front]+ 100*img_pad[y_front][x_front_two]
     + -100*img_pad[y_front_two][x_back_two]+ -100*img_pad[y_front_two][x_back]+ -100*img_pad[y_front_two][x_center]+ -32*img_pad[y_front_two][x_front]+ 100*img_pad[y_fr

k5 = 100*img_pad[y_back_two][x_back_two]+100*img_pad[y_back_two][x_back]+100*img_pad[y_back_two][x_center]+100*img_pad[y_back_two][x_front]+100*img_pad[y_back_two][x_fro
     + -32*img_pad[y_back][x_back_two]+ 78*img_pad[y_back][x_back]+100*img_pad[y_back][x_center]+ 100*img_pad[y_back][x_front]+ 100*img_pad[y_back][x_front_two] \
     + -100*img_pad[y_center][x_back_two]+ -92*img_pad[y_center][x_back]+ 92*img_pad[y_center][x_front]+ 100*img_pad[y_center][x_front_two] \
     + -100*img_pad[y_front][x_back_two]+ -100*img_pad[y_front][x_back]+ -100*img_pad[y_front][x_center]+ -78*img_pad[y_front][x_front]+ 32*img_pad[y_front][x_front_two]
     + -100*img_pad[y_front_two][x_back_two]+ -100*img_pad[y_front_two][x_back]+ -100*img_pad[y_front_two][x_center]+ -100*img_pad[y_front_two][x_front]+ -100*img_pad[y_
```