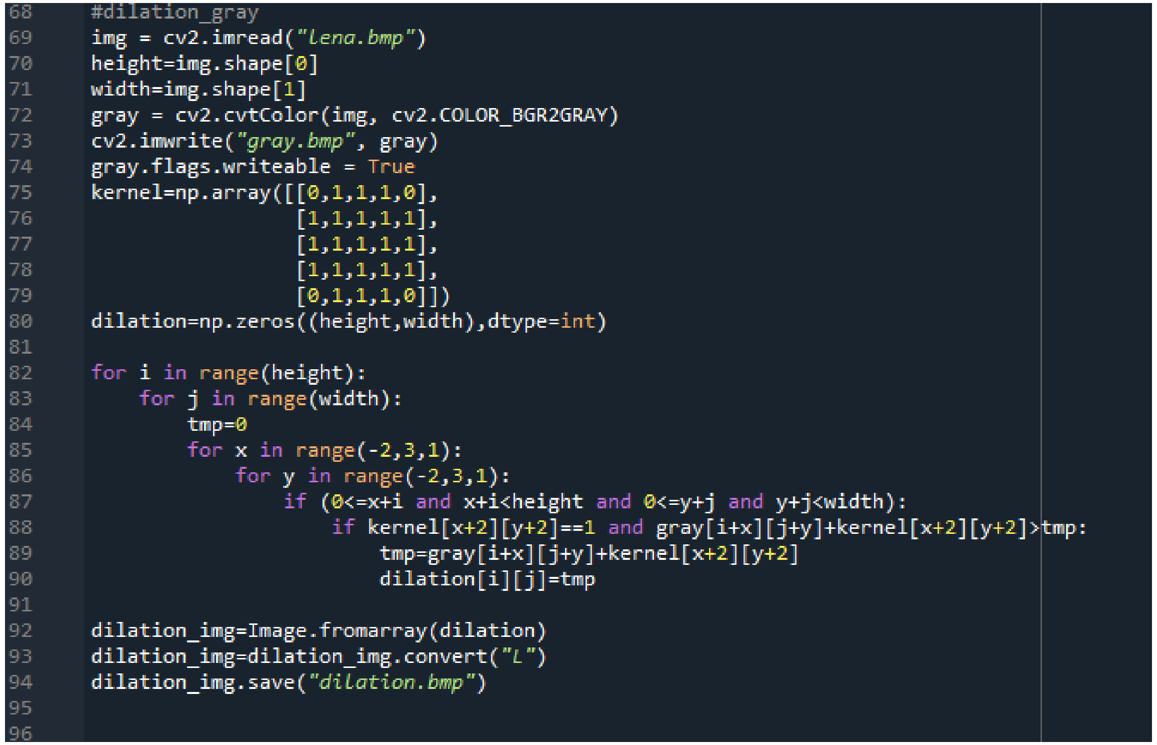
電腦視覺hw5

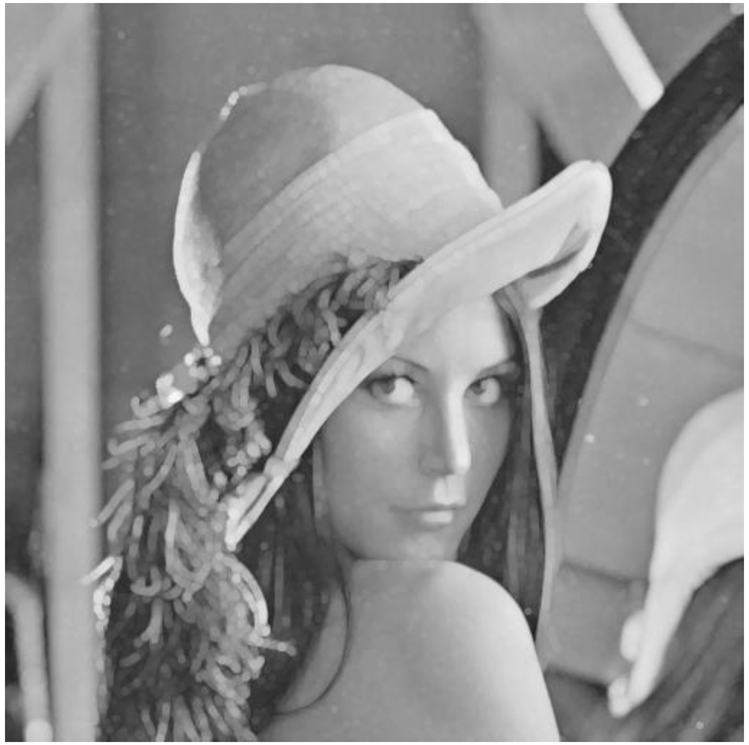
R11922131 周柏諺



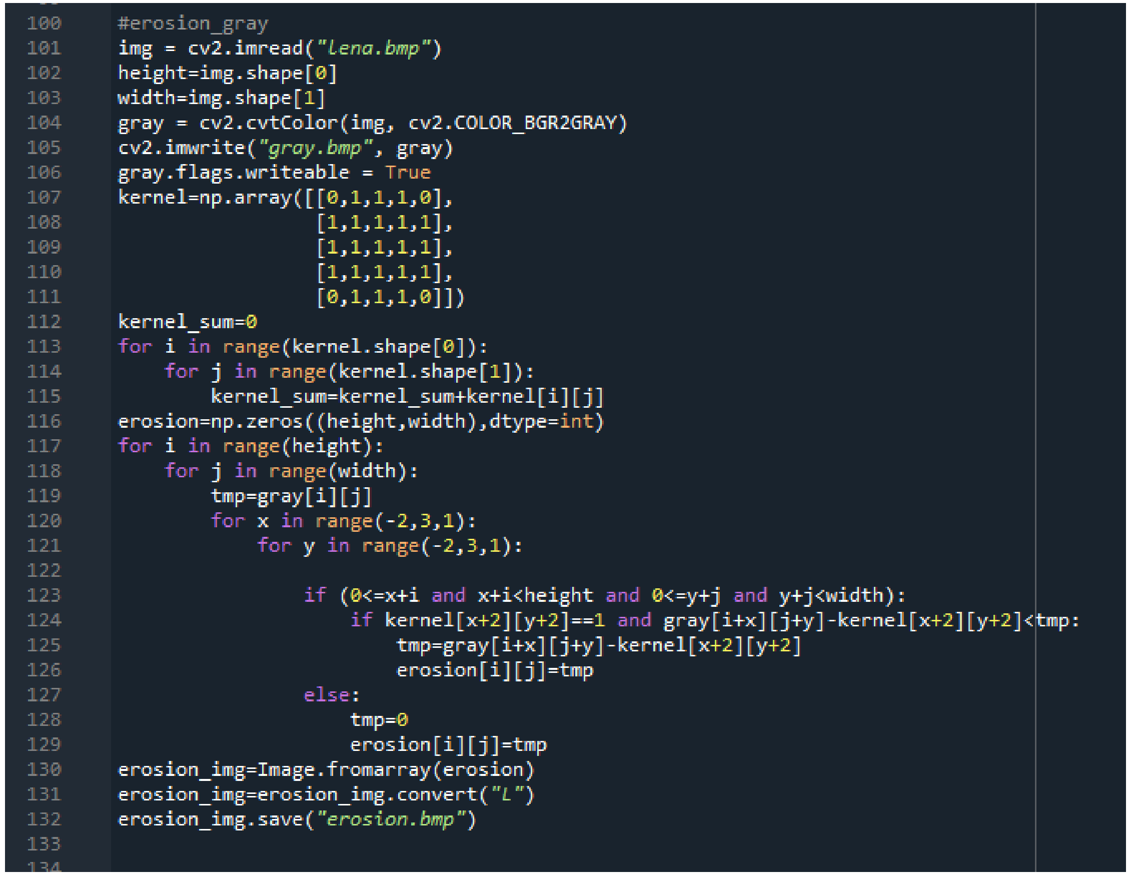
```
#dilation_gray
img = cv2.imread("lena.bmp")
height=img.shape[0]
width=img.shape[1]
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imwrite("gray.bmp", gray)
gray.flags.writeable = True
kernel=np.array([[0,1,1,1,0],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [0,1,1,1,0]])
dilation=np.zeros((height,width),dtype=int)

for i in range(height):
    for j in range(width):
        tmp=0
        for x in range(-2,3,1):
            for y in range(-2,3,1):
                if (0<=x+i and x+i<height and 0<=y+j and y+j<width):
                    if kernel[x+2][y+2]==1 and gray[i+x][j+y]+kernel[x+2][y+2]>tmp:
                        tmp=gray[i+x][j+y]+kernel[x+2][y+2]
                        dilation[i][j]=tmp

dilation_img=Image.fromarray(dilation)
dilation_img=dilation_img.convert("L")
dilation_img.save("dilation.bmp")
```

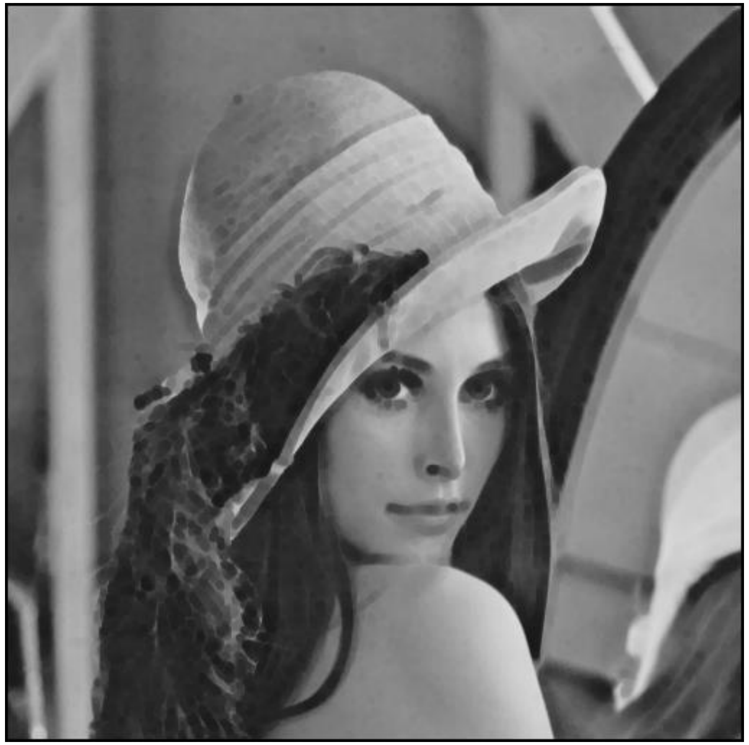


dilation

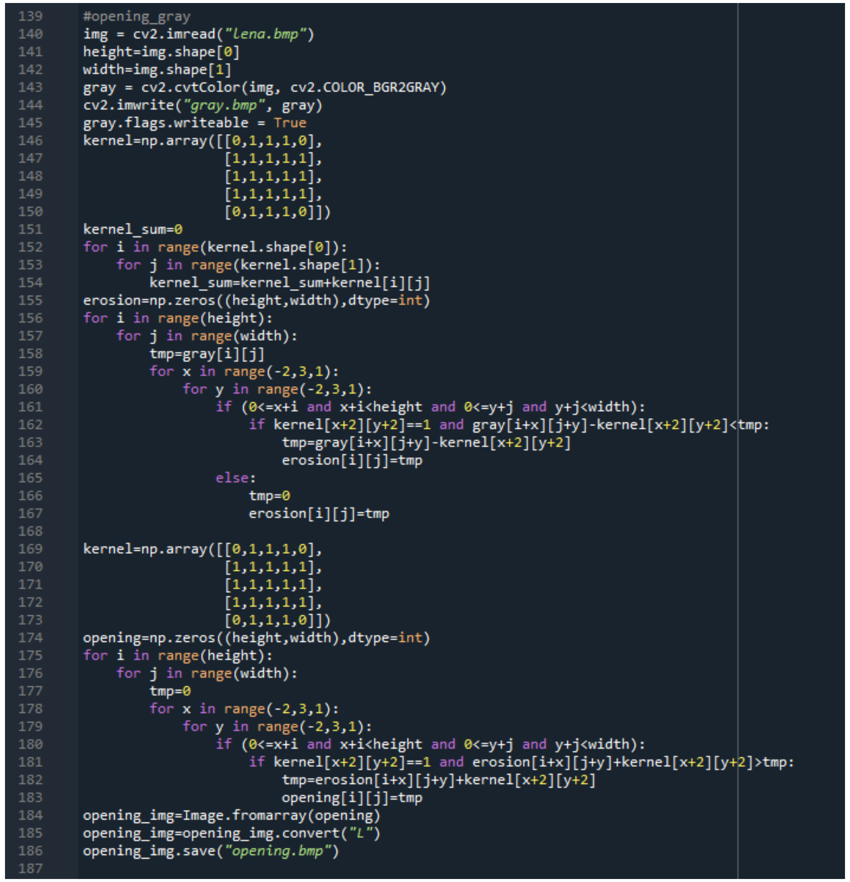


```
#erosion_gray
img = cv2.imread("lena.bmp")
height=img.shape[0]
width=img.shape[1]
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imwrite("gray.bmp", gray)
gray.flags.writeable = True
kernel=np.array([[0,1,1,1,0],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [0,1,1,1,0]])
kernel_sum=0
for i in range(kernel.shape[0]):
    for j in range(kernel.shape[1]):
        kernel_sum=kernel_sum+kernel[i][j]
erosion=np.zeros((height,width),dtype=int)
for i in range(height):
    for j in range(width):
        tmp=gray[i][j]
        for x in range(-2,3,1):
            for y in range(-2,3,1):

                if (0<=x+i and x+i<height and 0<=y+j and y+j<width):
                    if kernel[x+2][y+2]==1 and gray[i+x][j+y]-kernel[x+2][y+2]<tmp:
                        tmp=gray[i+x][j+y]-kernel[x+2][y+2]
                        erosion[i][j]=tmp
                else:
                    tmp=0
                    erosion[i][j]=tmp
erosion_img=Image.fromarray(erosion)
erosion_img=erosion_img.convert("L")
erosion_img.save("erosion.bmp")
```

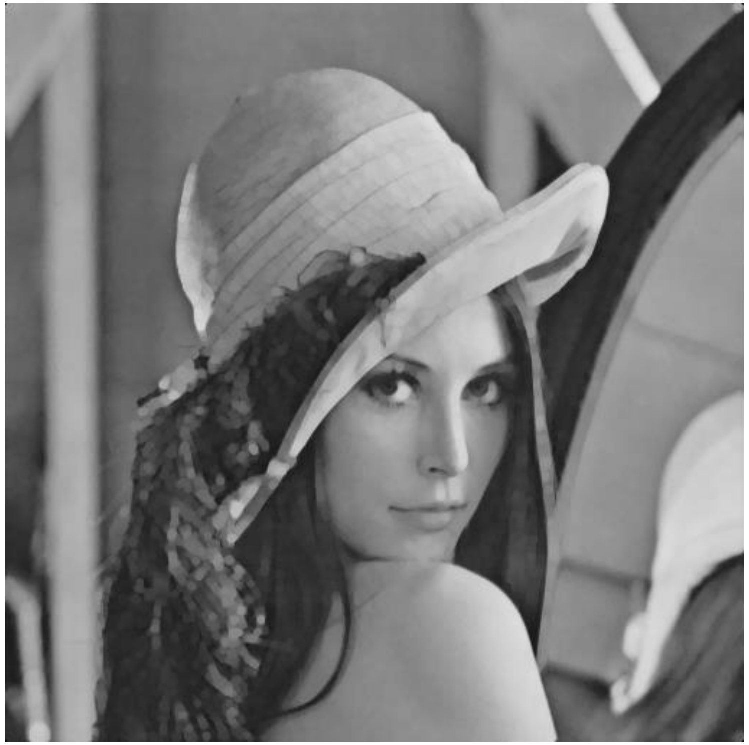


erosion

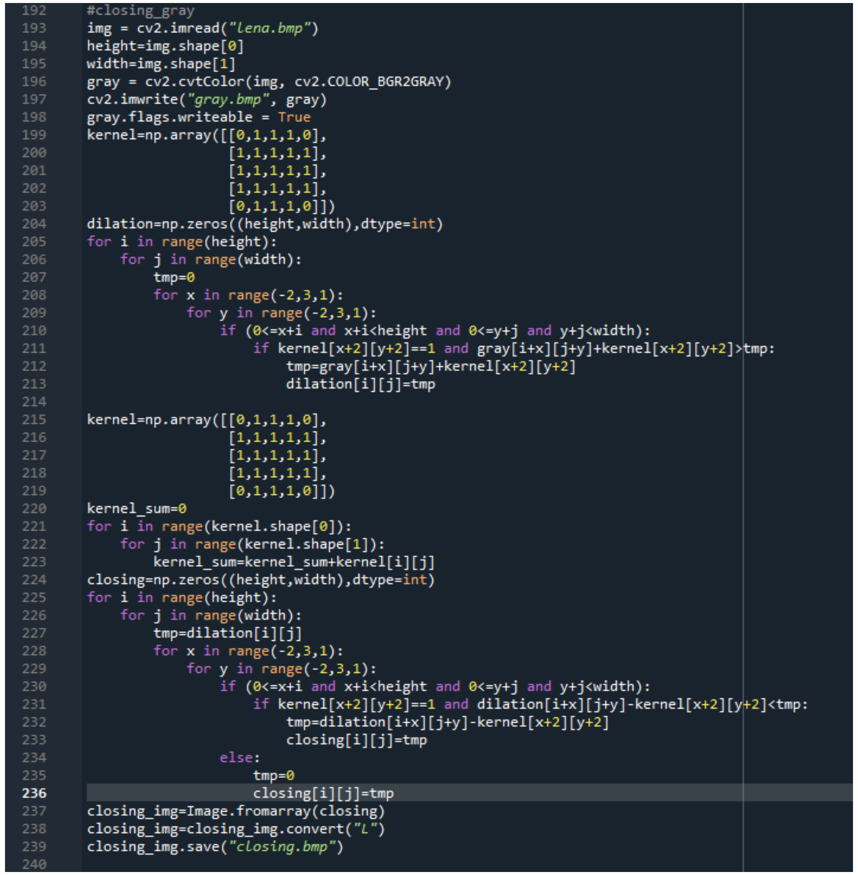


```python
#opening_gray
img = cv2.imread("Lena.bmp")
height=img.shape[0]
width=img.shape[1]
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imwrite("gray.bmp", gray)
gray.flags.writeable = True
kernel=np.array([[0,1,1,1,0],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [0,1,1,1,0]])
kernel_sum=0
for i in range(kernel.shape[0]):
    for j in range(kernel.shape[1]):
        kernel_sum=kernel_sum+kernel[i][j]
erosion=np.zeros((height,width),dtype=int)
for i in range(height):
    for j in range(width):
        tmp=gray[i][j]
        for x in range(-2,3,1):
            for y in range(-2,3,1):
                if (0<=x+i and x+i<height and 0<=y+j and y+j<width):
                    if kernel[x+2][y+2]==1 and gray[i+x][j+y]-kernel[x+2][y+2]<tmp:
                        tmp=gray[i+x][j+y]-kernel[x+2][y+2]
                        erosion[i][j]=tmp
                else:
                    tmp=0
                    erosion[i][j]=tmp

kernel=np.array([[0,1,1,1,0],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [0,1,1,1,0]])
opening=np.zeros((height,width),dtype=int)
for i in range(height):
    for j in range(width):
        tmp=0
        for x in range(-2,3,1):
            for y in range(-2,3,1):
                if (0<=x+i and x+i<height and 0<=y+j and y+j<width):
                    if kernel[x+2][y+2]==1 and erosion[i+x][j+y]+kernel[x+2][y+2]>tmp:
                        tmp=erosion[i+x][j+y]+kernel[x+2][y+2]
                        opening[i][j]=tmp
opening_img=Image.fromarray(opening)
opening_img=opening_img.convert("L")
opening_img.save("opening.bmp")
```

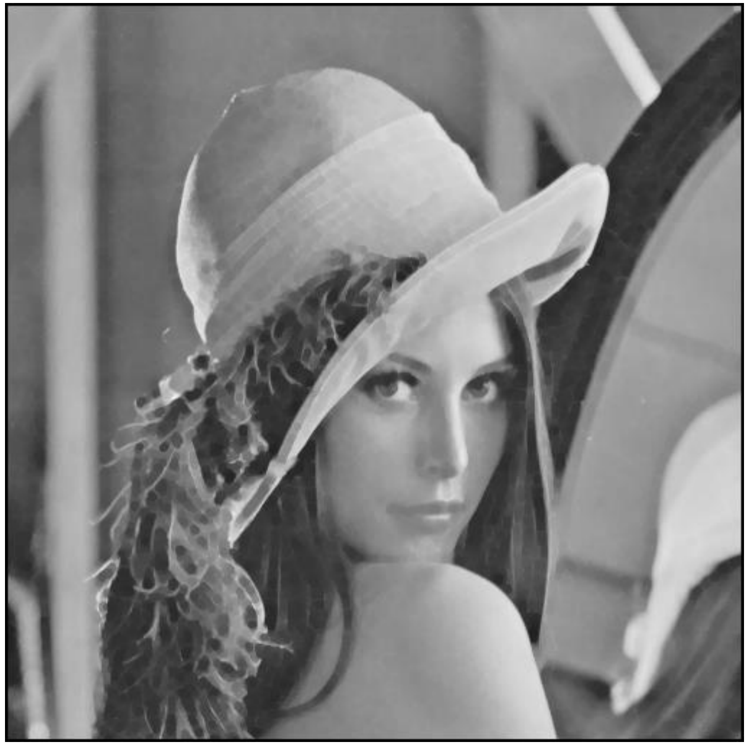


opening



```python
#closing_gray
img = cv2.imread("lena.bmp")
height=img.shape[0]
width=img.shape[1]
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imwrite("gray.bmp", gray)
gray.flags.writeable = True
kernel=np.array([[0,1,1,1,0],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [0,1,1,1,0]])
dilation=np.zeros((height,width),dtype=int)
for i in range(height):
    for j in range(width):
        tmp=0
        for x in range(-2,3,1):
            for y in range(-2,3,1):
                if (0<=x+i and x+i<height and 0<=y+j and y+j<width):
                    if kernel[x+2][y+2]==1 and gray[i+x][j+y]+kernel[x+2][y+2]>tmp:
                        tmp=gray[i+x][j+y]+kernel[x+2][y+2]
                        dilation[i][j]=tmp

kernel=np.array([[0,1,1,1,0],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [1,1,1,1,1],
                 [0,1,1,1,0]])
kernel_sum=0
for i in range(kernel.shape[0]):
    for j in range(kernel.shape[1]):
        kernel_sum=kernel_sum+kernel[i][j]
closing=np.zeros((height,width),dtype=int)
for i in range(height):
    for j in range(width):
        tmp=dilation[i][j]
        for x in range(-2,3,1):
            for y in range(-2,3,1):
                if (0<=x+i and x+i<height and 0<=y+j and y+j<width):
                    if kernel[x+2][y+2]==1 and dilation[i+x][j+y]-kernel[x+2][y+2]<tmp:
                        tmp=dilation[i+x][j+y]-kernel[x+2][y+2]
                        closing[i][j]=tmp
                else:
                    tmp=0
                    closing[i][j]=tmp
closing_img=Image.fromarray(closing)
closing_img=closing_img.convert("L")
closing_img.save("closing.bmp")
```



closing