HW1

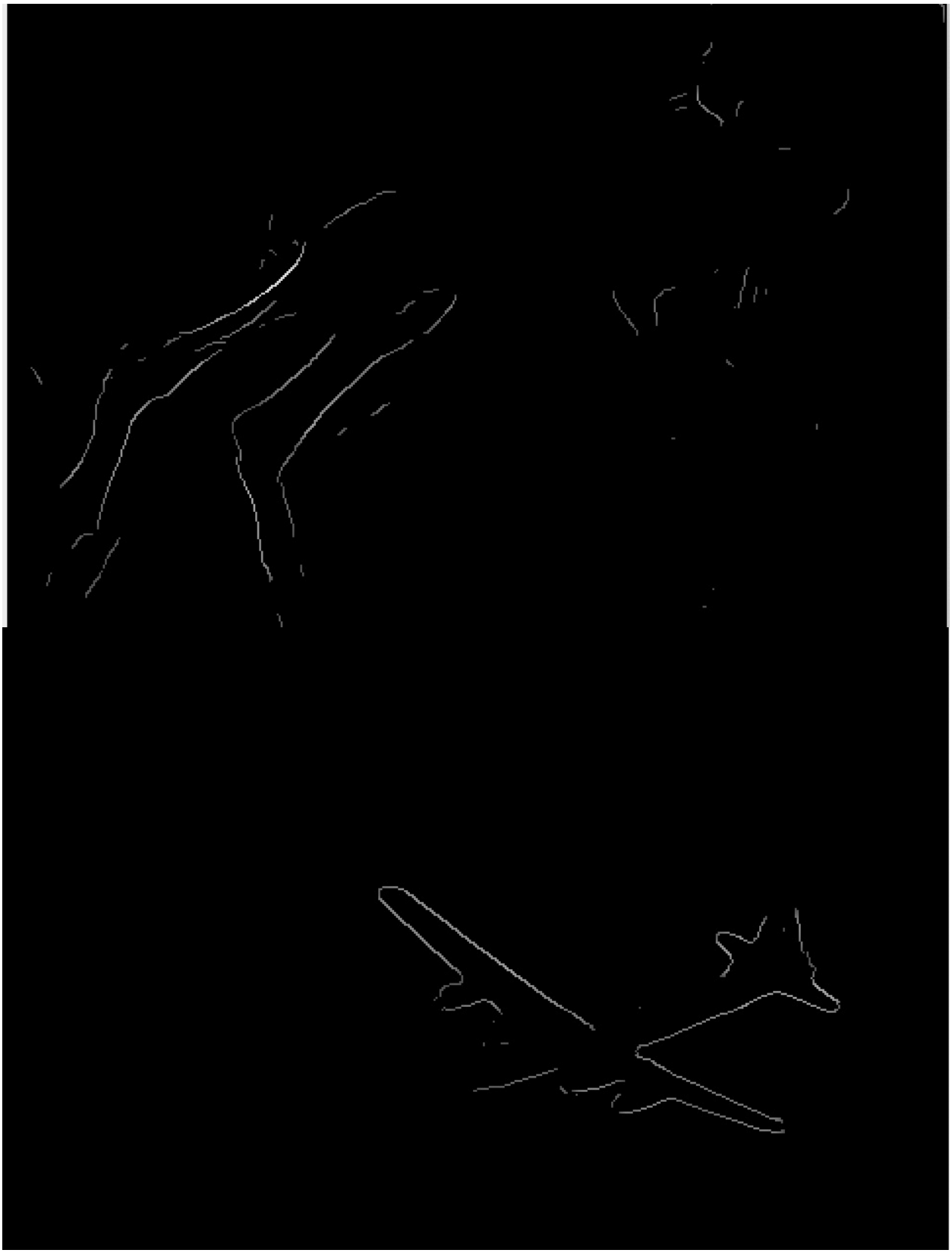
Code

```
import PIL.Image as Image
import numpy
import math

def process_iamge(image, size):
    size = (size - 1) / 2
    size = int(size)
    row, col = image.shape
    row_op = row + 2 * size
    col_op = col + 2 * size
    image_op = numpy.zeros((row_op, col_op))
    for i, j in [(i, j) for i in range(row) for j in range(col)]:
        image_op[i + size][j + size] = image[i][j]
    for i, j in [(i, j) for i in range(size) for j in range(size)]:
        image_op[i][j] = image_op[size][size]
    for i, j in [(i, j) for i in range(size) for j in range(col_op - size, col_op)]:
        image_op[i][j] = image_op[size][col_op - size - 1]
    for i, j in [(i, j) for i in range(row_op - size, row_op) for j in range(size)]:
        image_op[i][j] = image_op[row_op - size - 1][size]
    for i, j in [(i, j) for i in range(row_op - size, row_op) for j in range(col_op - size,
        col_op)]:
        image_op[i][j] = image_op[row_op - size - 1][col_op - size - 1]
    for i, j in [(i, j) for i in range(size) for j in range(size, col_op - size)]:
        image_op[i][j] = image_op[size][j]
    for i, j in [(i, j) for i in range(row_op - size, row_op) for j in range(size, col_op - size)]:
        image_op[i][j] = image_op[row_op - size - 1][j]
    for i, j in [(i, j) for i in range(size, row_op - size) for j in range(size)]:
        image_op[i][j] = image_op[i][size]
    for i, j in [(i, j) for i in range(size, row_op - size) for j in range(col_op - size, col_op)]:
            image_op[i][j] = image_op[i][col_op - size - 1]
    return image_op


def convolute(ip, filter):
    row, col = ip.shape
    size = int(math.sqrt(filter.size))
    row = row - 2 * int((size - 1) / 2)
    col = col - 2 * int((size - 1) / 2)
    image = numpy.zeros((row, col))
    for i, j, k, l in [(i, j, k, l) for i in range(row) for j in range(col) for k in range(size)
        for l in range(size)]:
        image[i][j] = image[i][j] + (filter[k][l] * ip[i + k][j + l])
    return image
```

```
def main():
    image_name = input('img name: ')
    sigma = input('sigma: ')
    size = input('size: ')
    image = Image.open(image_name)
    image.show()
    im, sigma2 = image, sigma
    size_g = int(size)
    if (size_g % 2 == 0):
        size_g += 1
    sigma2 = float(sigma2)
    sum = 0
    image3 = numpy.array(im)
    filter = numpy.zeros((size_g,size_g))
    filter_n = int((size_g-1)/2)
    y, x = numpy.ogrid[float(-filter_n):float(filter_n+1),float(-filter_n):float(filter_n+1)]
    for i, j in [(i, j) for i in range(size_g) for j in range(size_g)]:
        filter[i][j] =
            (math.exp((-((x[0][j]**2)+(y[i][0]**2))/(2*(sigma2**2)))))*(1/(2*math.pi*(sigma2**2)))
        sum = sum + filter[i][j]
    for i, j in [(i, j) for i in range(size_g) for j in range(size_g)]:
            filter[i][j] = filter[i][j]/sum
    temp1 = Image.fromarray(convolute(process_iamge(image3, size_g), filter))
    original_image = numpy.array(temp1)
    t_im = process_iamge(original_image, 3)
    x_image = convolute(t_im, numpy.array([[-1, 0, +1], [-2, 0, +2], [-1, 0, +1]]))
    y_image = convolute(t_im, numpy.array([[+1, +2, +1], [0, 0, 0], [-1, -2, -1]]))
    row, col = original_image.shape
    orient = numpy.zeros((row, col))
    image_opr = numpy.zeros((row, col))
```

```
(base) kannimne:cs558_hw1 pinwei$ python hw1.py
img name: kangaroo.pgm
sigma: 3
size: 9
(base) kannimne:cs558_hw1 pinwei$ python hw1.py
img name: plane.pgm
sigma: 3
size: 9
(base) kannimne:cs558_hw1 pinwei$ python hw1.py
img name: red.pgm
sigma: 3
size: 9
(base) kannimne:cs558_hw1 pinwei$
```

Output




Search