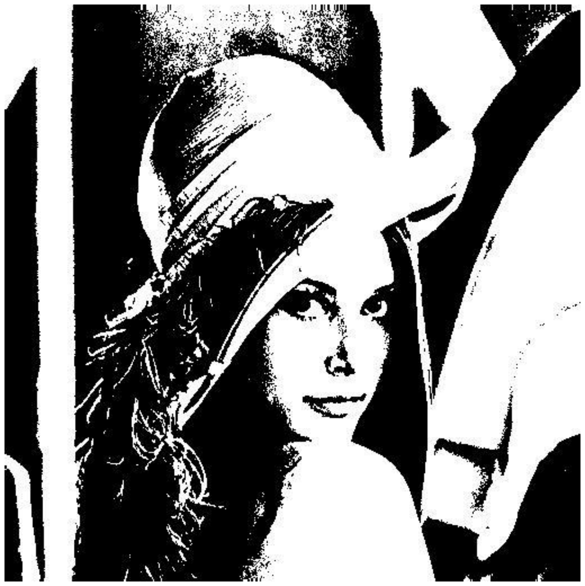
Computer Vision HW4 Report
蕭恩慈 / B07902095

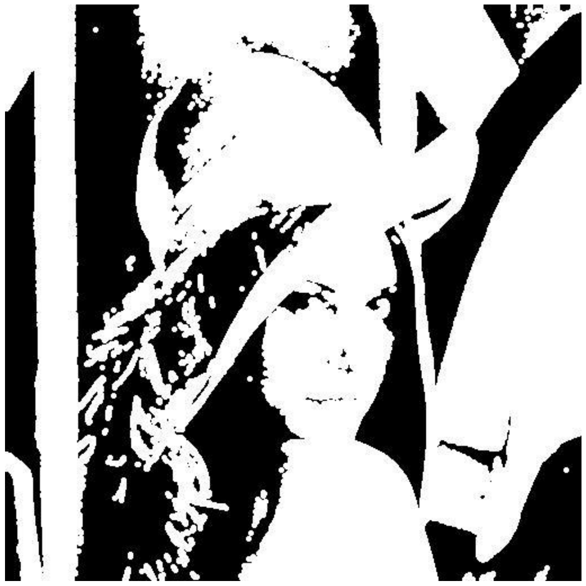
The task is to write a program which does binary morphology on a binary image of dilation, erosion, opening, closing, and hit-and-miss.
To complete the task, I use Python as my program language using numpy and PIL library. For start, make the original image into a binary image, to be used later as the "original image" in the main program.

(original image: lena.bmp)


(binary image: binary-lena.bmp)

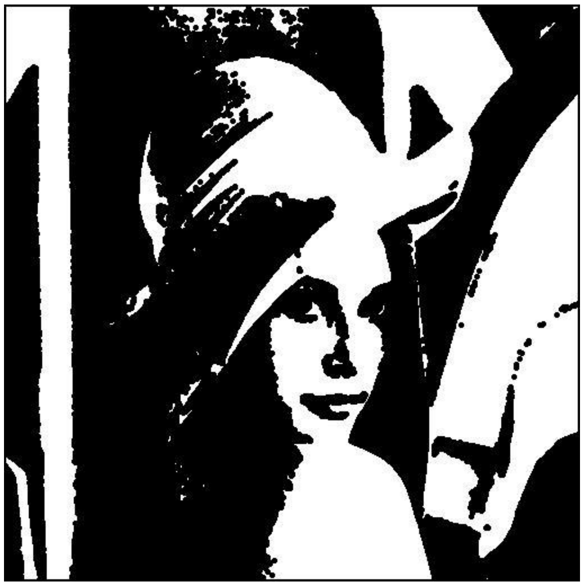
For the morphology method, we want to get the binary image then turn it into kernel by cutting the binary image into pixels, then turned each pixel into kernel, then calculate the center position.
For dilation, erosion, opening and closing process, we set the kernel the octagonal 3-5-5-5-3 kernel
For dilation process, if the kernel value is "1", calculate the destination of x and y position. To avoid getting out of range image, we want to set the program to only put the value "1" for position that is within the image size.

Result:


(dilated-lena.bmp)

For erosion process, set a flag matching indicator, then like the dilation part before, configure the range that is within the image size. For the image within the range, check if there is a match with the kernel. Put the value "1" to the original image if and only if there exists a match. (To check if there is a match, check in within-the-range x y position and filter out whether the pixel of the binary image is 0)

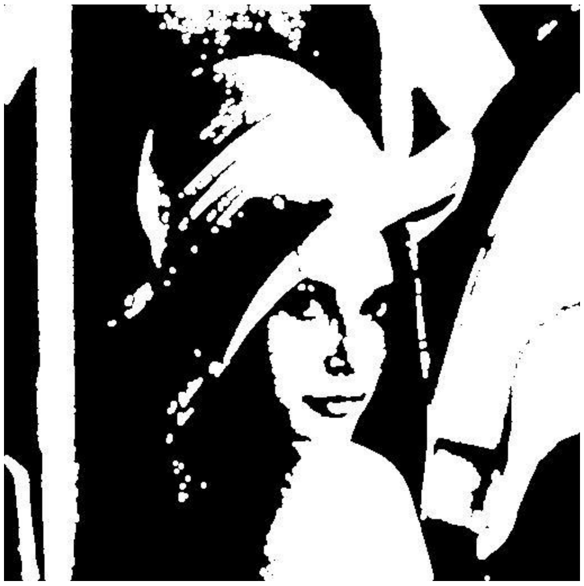
Result:



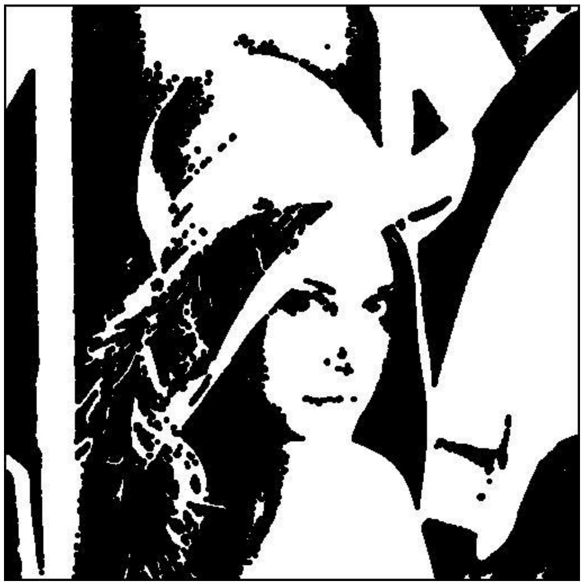
(erosion-lena.bmp)

Opening and closing process basically involves the dilation and erosion process. The difference is for opening process, it checks for the union of all translations of y that fit entirely within x, while closing is the complement of opening process. So to create opening process, run the erosion function within the dilation function, then for closing, run the dilation function within the erosion function.

Result:



(opening-lena.bmp)



(closing-lena.bmp)

For hit and miss process, we first need to figure out the complement of binary image, then create a intersection function. The intersection function is used for comparing the original binary image, and the complement, if the pixels in each binary and complement image value is 0, then put the value “0” in the resulting intersection image. If both have value in it, then put value “1” in the resulting image. Next, is to create the erosion function but with the assigned kernel center, so we do not need to calculate the center kernel. Lastly, for the main hit and miss function, run the erosion function of the binary image, and the complement image of it within the intersection function. The goal is to find the intersection of the eroded original image, and the eroded complement image.
Upon running the function, set the kernel K and J as [1, 1] and [0, 1]. While the center of kernel J is (1, 0) and kernel K is (0, 1).

Result:

(hit-and-miss-lena.bmp)