Language: Python3.8.3

Modules: OpenCV(cv2), numPy, MyMathematicalMorphologyModules
(self-defined module)

(a) Dilation

Algorithm:
Let the original binary image called **lena1**, and a copy of **lena1** called **lena2**. For all the pixels in **lena1**, if the pixel is white, we whiten all the corresponding pixels in **lena2**, which are generated from current pixel dilating by the kernel (3-5-5-5-3) and notice to check if the dilation is out of the image range. **lena2** is the result that we want.

(b) Erosion

Algorithm:
Let the original binary image called **lena1**, and a copy of **lena1** called **lena2**. For all the pixels in **lena1**, we use a flag (Initialized by 1) to determine if the pixel can match the kernel. If some part of kernel is out of the image range or not corresponded, the flag will off (0). This flag value will determine whether the same pixel in **lena2** should be white or black. **lena2** is the result that we want.

(c) Opening

Algorithm:

Opening is just doing erosion and then dilation on the image with the same kernel.

(d) Closing

Algorithm:

Closing is just doing dilation and then erosion on the image with the same kernel.

(e) Hit-and-miss transform

Algorithm:
We have two kinds of kernel: J (containing the origin) and K (not containing the origin). Both of kernels have three pixels on the right-up corner with L shape. This algorithm has three parts. First, erode **lena1** by kernel J. Next, erode **the reverse image of lena1** by kernel K. Finally, find the intersection of these two eroded images, we will get the result that we want.