Lab 04, Morphological Operations

João Santos, *MRSI, 76912*

***Index Terms*—OpenCV, Computer Vision, MRSI, UA, DETI, LaTeX.**

I. Introduction

THIS report is intended to be used alongside the Python3 code developed for this Lab.

The Lab #04 is a introduction class to morphological operations, OpenCV and Python 3.

This report was written using LaTeX .

II. Exercises

Lets analyse the resolution of the proposed exercises.

A. Ex. 4.1: Dilation

For this exercise, we were required to apply a dilation to a given image (Fig. 1) and, then, analyse the differences between using different kernels and amount of iterations.

Figs. 2 and 3 show the effect when only applying the dilation once. The first obvious effect is that the outer bounds of the region get expanded, i.e. the region grows. Contrarily, the inner hole has shrunk because the bounds of the hole get "expanded" towards the black portion.

We can see that when using a circular kernel, the outer corners of the region get rounder compared to the base image and when using a rectangular kernel. Also, the inner hole gets a little less round with the rectangular kernel.

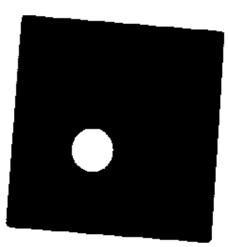

Fig. 1. Base image for this exercise.

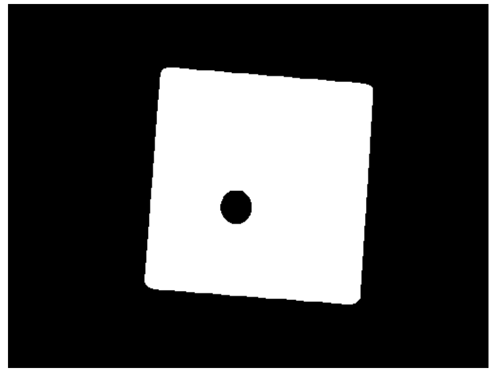

Fig. 2. Dilation using a circle with radius 11.

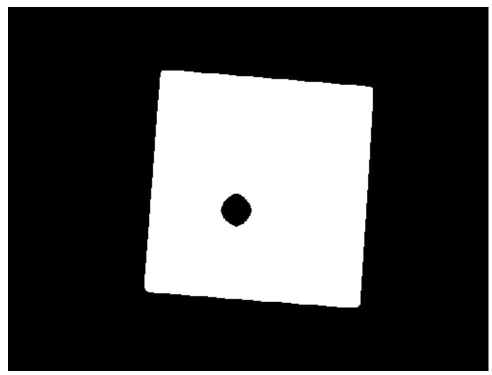

Fig. 3. Dilation using a square with side 11.

If we apply those same kernels, to the same image, but ten times, the changes are much more dramatic. The outputs of this procedure are Figs. 4 and 5 we see the same tendency for the outer bounds but with increased effect. Also, notice on both images the hole as been closed on the region itself due to the successive dilations.

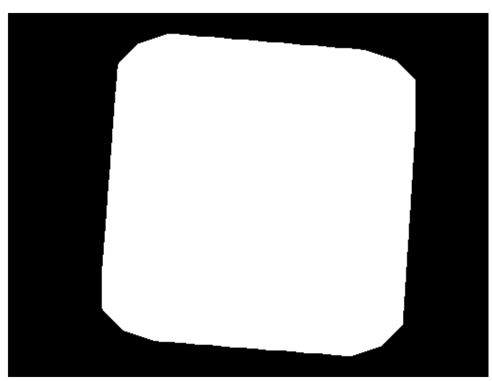

Fig. 4. Dilation using a circle with radius 11 (using 10 iterations).

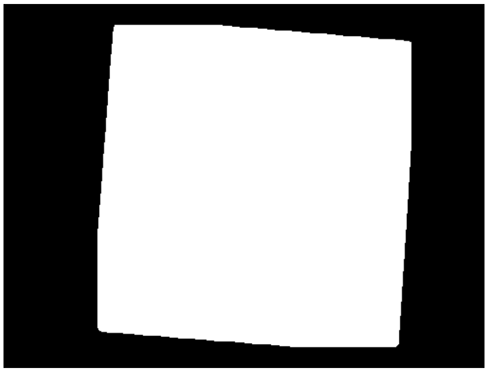

Fig. 5. Dilation using a square with side 11 (using 10 iterations).

B. Ex. 4.2: Edge detection with morphological operations

Using again Fig. 1, the author carried out an edge detection methodology using morphological operations. More precisely, a subtraction to the dilated thresholded image was performed, using a 3x3 square kernel. Please Fig. 6 for a demonstration.

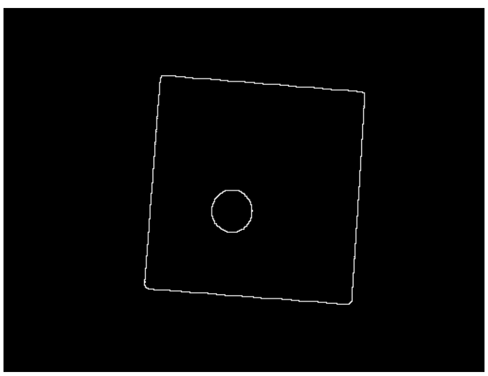

Fig. 6. Edges found using a 3x3 square kernel.

As expected, by using a larger kernel, the dilation image is larger and, therefore, the subtraction will ”produce” thicker edges, as shown in Fig. 7.

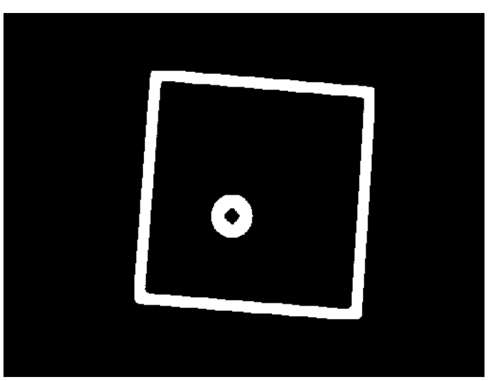

Fig. 7. Edges found using a 23x23 square kernel.

C. Ex. 4.3: Erosion

Contrarily to the explored on II-A, erosion essentially ”eats” the borders (inner and outer) of a region. Again using Fig. 1 as reference, we eroded the, using a circular kernel with eleven pixels of diameter, one and eleven fold. The outcomes are, respectively, on Figs. 8 and 9. In this Figs. it is clear that the use of excessive iterations can lead to the almost (or even complete) removal of a region.

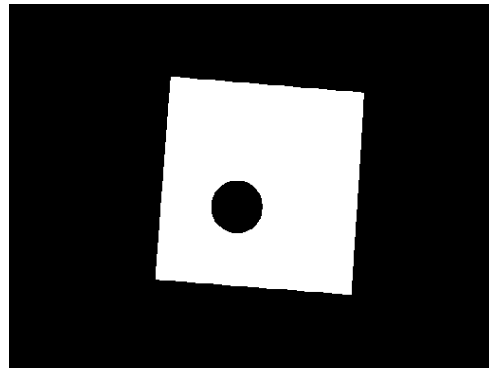

Fig. 8. Erosion using a circular kernel (11x11) with a single iteration.

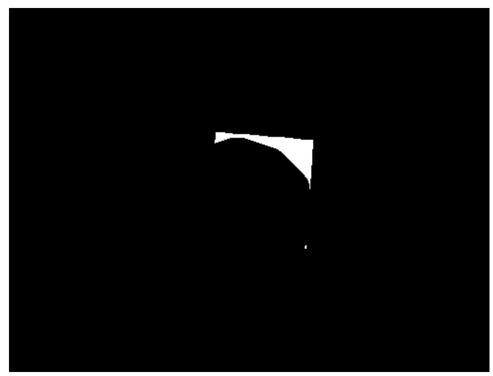

Fig. 9. Erosion using a circular kernel (11x11) with eleven iterations.

If, by other hand, we do the same procedure but with a rectangular kernel, it is observable that the inner hole gets ”squared” as a side effect (Fig. 10). This effect is also noticeable when using eleven iterations as the corner of the kernel get further enough to split the region into two (Fig. 11).

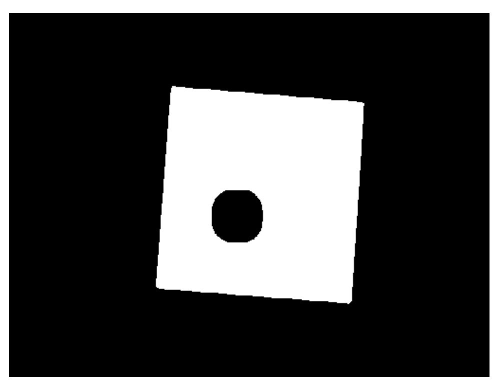

Fig. 10. Erosion using a rectangular kernel (11x11) with a single iteration.

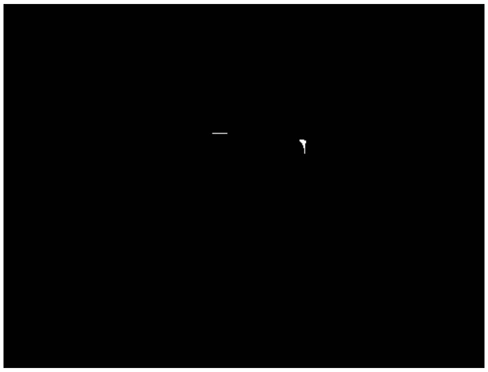

Fig. 11. Erosion using a rectangular kernel (11x11) with eleven iterations.

Using an asymmetric kernel, the effect on Fig. 12 is similar to the described for Fig. 10 but only in the larger dimension of the kernel.

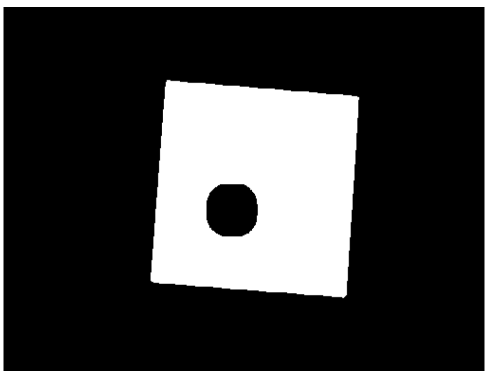

Fig. 12. Erosion using a rectangular kernel (11x1) with a single iteration.

Another way to build am asymmetric kernel is to change its anchor point. In Fig. 13 we used a 3x3 rectangular kernel but with the anchor on the upper left pixel. The effect is not that noticeable but what happens is that the region is shrunk most os the lower right corner due to the said asymmetry of the kernel.

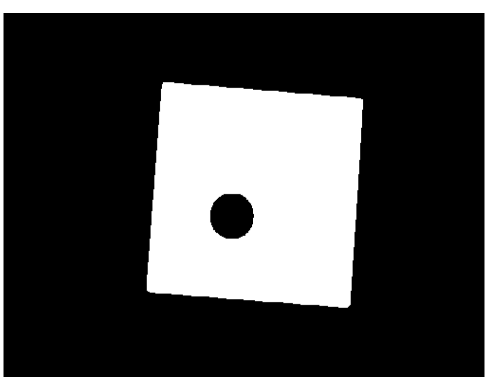

Fig. 13. Erosion using a rectangular kernel (3x3) with eleven iterations and the upper left corner as the anchor.

D. Ex. 4.4: Segmentation with morphological operations

Now using Fig. 14, the goal is to segment the circular portions of the image using will an erosion, applied twice.

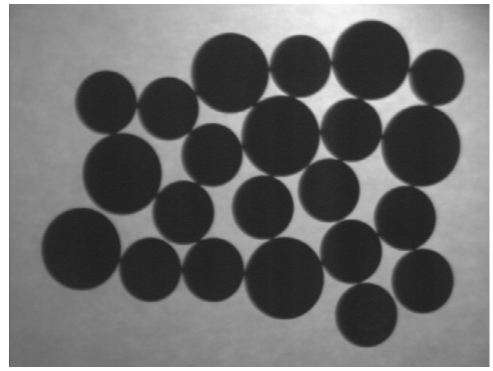

Fig. 14. Base figure for this exercise.

The outputs differ when we use a circular (eleven pixels of diameter, see Fig. 15) or a rectangular (square with a size og 9 pixels, see Fig. 16) kernel. Since we are trying to segment circular regions, a circular kernels proves to be the best choice as it gives smother borders to the regions. The usage of the square kernel also segments the same regions, but produces borders with noticeable ”spikes”, which are not problematic in this specific case, but in some other image can lead to regions being connected by the corners created.

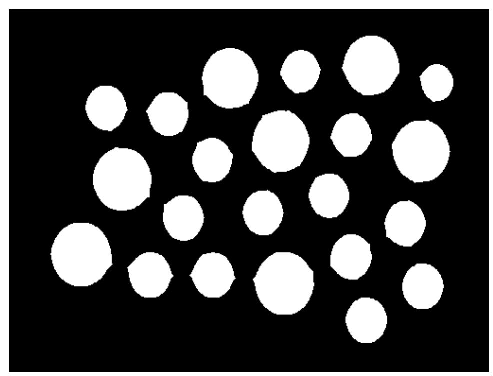

Fig. 15. Erosion using a circular (11x11) kernel, with two iteration.

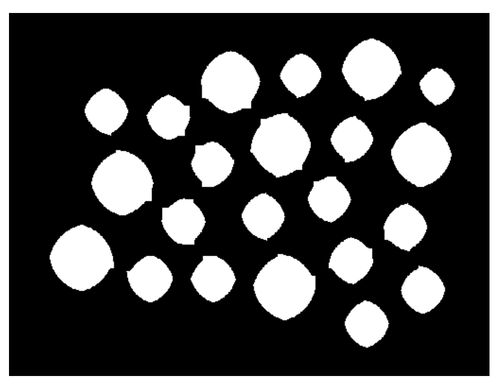

Fig. 16. Erosion using a rectangular (11x11) kernel, with two iteration.

E. Ex. 4.5: Opening

The opening morphological operation can be useful in a variate of situations. Namely, as proposed in this exercise, we want to segment the circular regions on Fig. 17 that are overlapped by several lines.

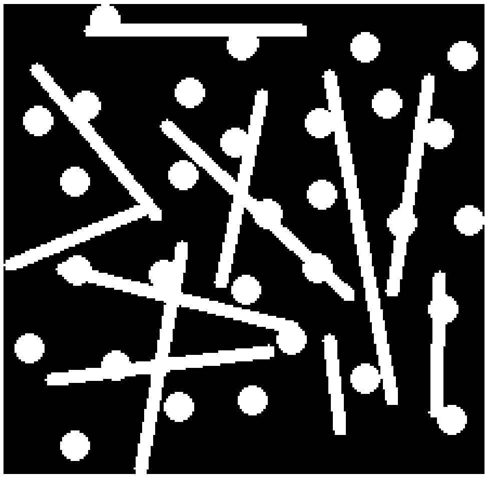

Fig. 17. Base image for ex. 4.5.

By doing a morphological opening with a carefully selected size (in this case, a circular kernel with eleven pixels of diameter), we can very easily and with only one iteration segment the desired regions (see Fig. 18)

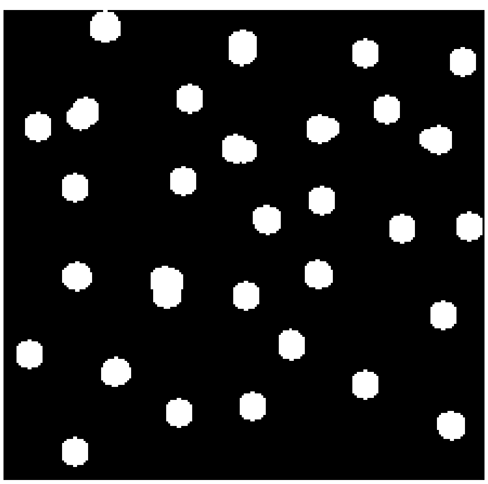

Fig. 18. Segmented image after the opening operation.

We can also use the same method, but with asymmetrical kernels, to segment the vertical and horizontal lines on Fig. 19.

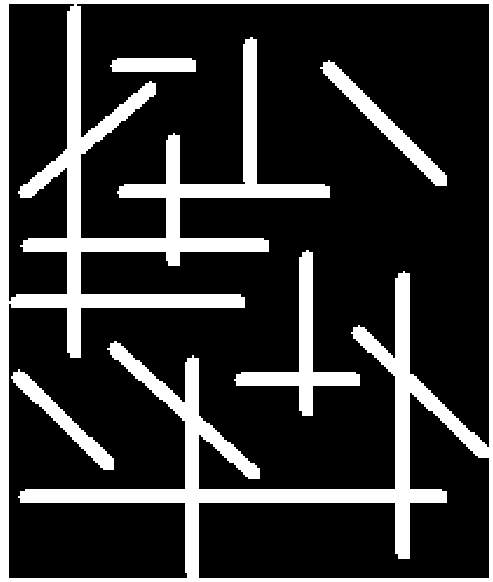

Fig. 19. Image where vertical and horizontal lines will be segmented using the opening operation.

By using a taller kernel (3x9) the vertical portions will be transformed into line-like shapes during the erosion phase, will the horizontal will vanish. Then, when the dilation happens, the line-like structures get close to the original thickness, performing this way the segmentation. A parallel thought can be developed in the other case, where using a widder kernel (9x3) produces the segmentation of the horizontal structures.

The outputs are, respectively, Figs. 20 and 21.

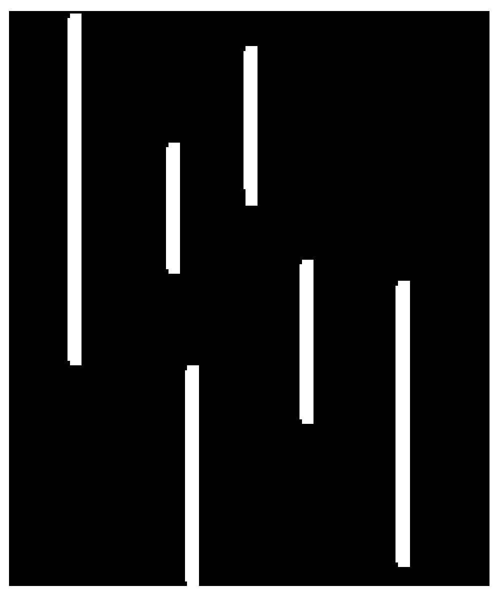

Fig. 20. Vertical regions segmented.

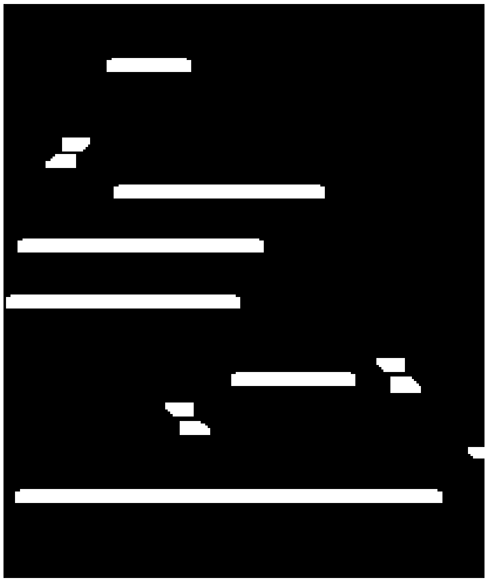

Fig. 21. Horizontal regions segmented.

F. Ex. 4.6: Closing

Inversely to the opening operation, the closing operation firstly dilates the image to, only then, erode it.

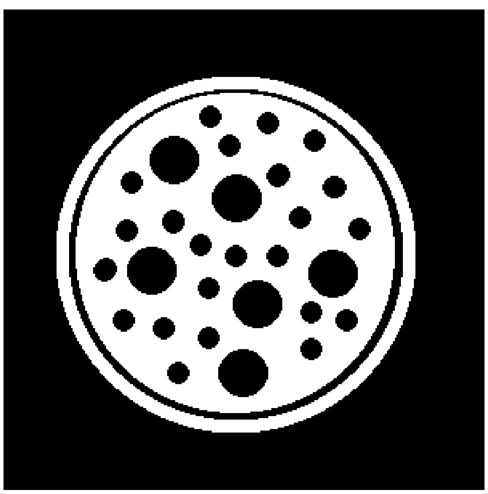

Fig. 22. Base image for closing.

In this exercise, we used circular kernels with three different diameters: twenty three, five and fifty one pixels.

When using the diameter of twenty three pixels, the size is just right to fill the black ring as well as the smaller holes (see Fig. 23).

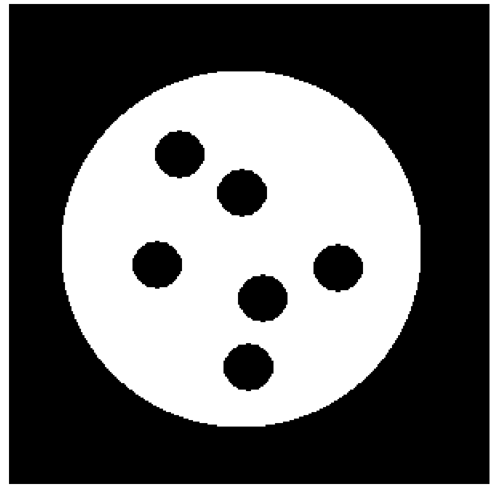

Fig. 23. Base image closed with a circular kernel of diameter twenty three pixels.

Using the smaller diameter of the set, the output is not as well ”polished” as the smaller holes are still on the image and the black ring is only partially filled (see Fig. 24).

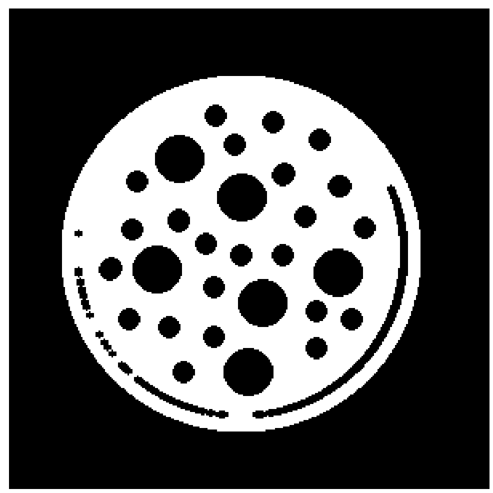

Fig. 24. Base image closed with a circular kernel of diameter five pixels.

On the other hand, the largest diameter, the dilation step is enough to fill all the region and then the erosion only sets the object close to its original size (see Fig. 25).

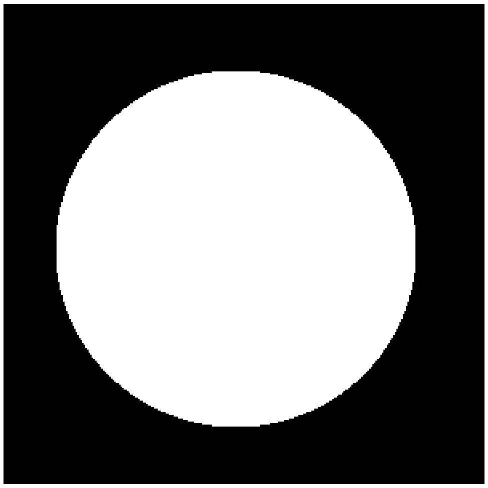

Fig. 25. Base image closed with a circular kernel of diameter fifty one pixels.

G. *Ex. 4.7: Region Segmentation using Flood-Fill*

The flood-fill operation allows us to segment a image based on the gray level of the neighbors. Is we thinks of a gray image as a 3D image, where the gray value would be the height of that X, Y coordinate, what the flood fill does is filling the region adjacent to the selected pixel that have a ”height” within a certain bounds of the selected one.

Using the famous Lena picture as example, we can use the flood-fill operation to segment the regions containing the pixel (430,30) while the gray value is within ± 5 of the chosen pixel, has seen on Fig. 26

Fig. 26. Flood-fill applied to the lena picture.

1) Interactive Selection: Using a mouse callback, we can make this process interactive. Figs. 27, 28 and 29 show the outputs of interactive flood-fill applied to different base images.

Fig. 27. Flood-fill applied to the lena picture, but a different region.

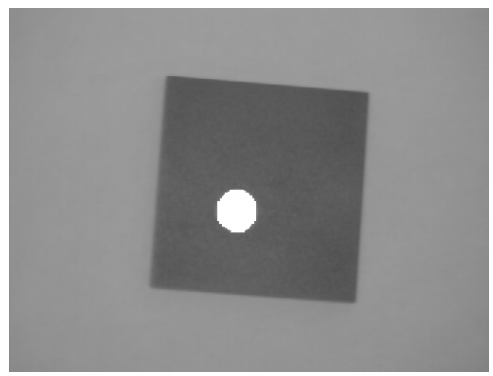

Fig. 28. Flood-fill applied to the wdg2 picture.

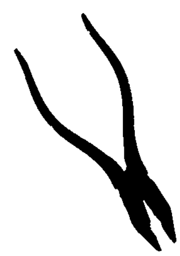

Fig. 29. Flood-fill applied to the tools2 picture.