Image Processing
lab 3
Group 1

Alessandro Pianese
S4106431

Viljami Linna
S4169921

January 13, 2020

Individual contributions

Viljami Linna mainly planned, implemented, coded and wrote the report related to Exercise 3. The code and report related to Exercise 1 and exercise 2 were mainly planned and implemented by Alessando Pianese. Both members of the group are familiar with all exercise solutions. Structure and language of the report has been checked by both members of the group.

REMARK: Alessandro Pianese S4106431 submitted his part of the report on time on Monday 13/01/2020.
Viljami Linna will submit as soon as possible his part of the report.

Exercise 1 – Basic morphology

Solution of problem 9.9 is displayed in figure 1, 2 and 3.

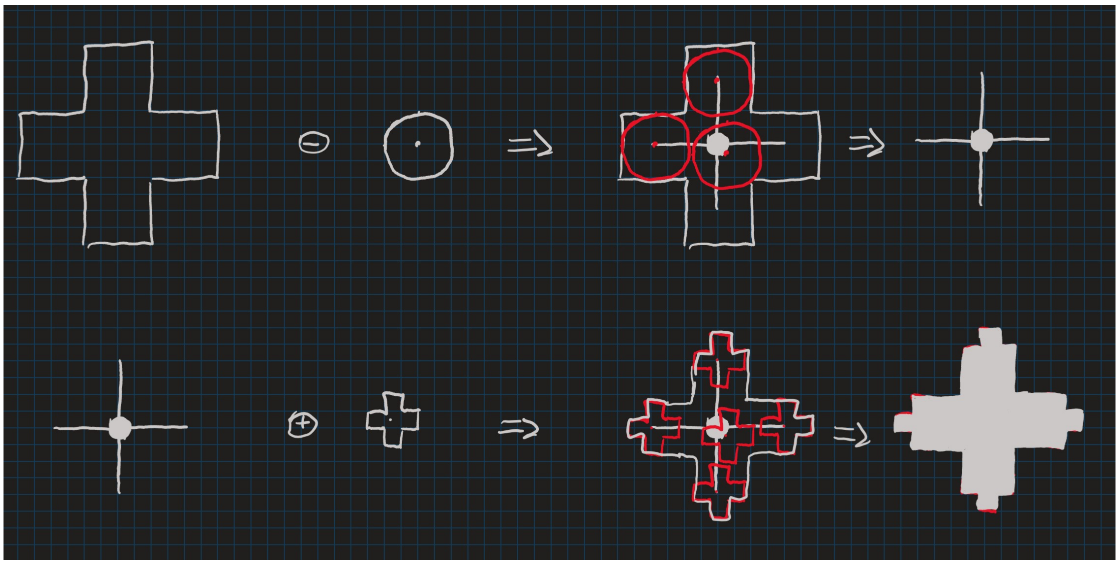


Figure 1: Solution of point a

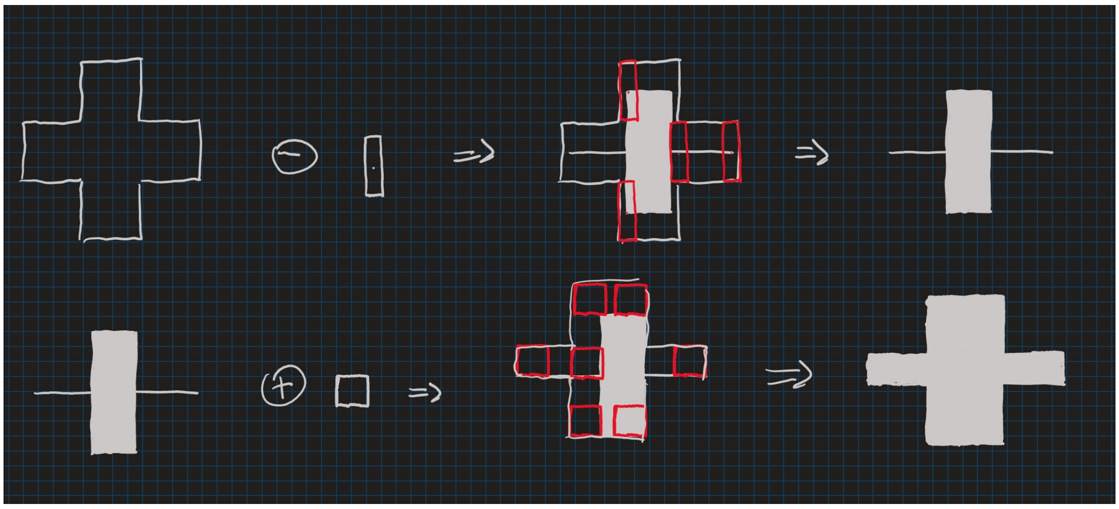


Figure 2: Solution of point b

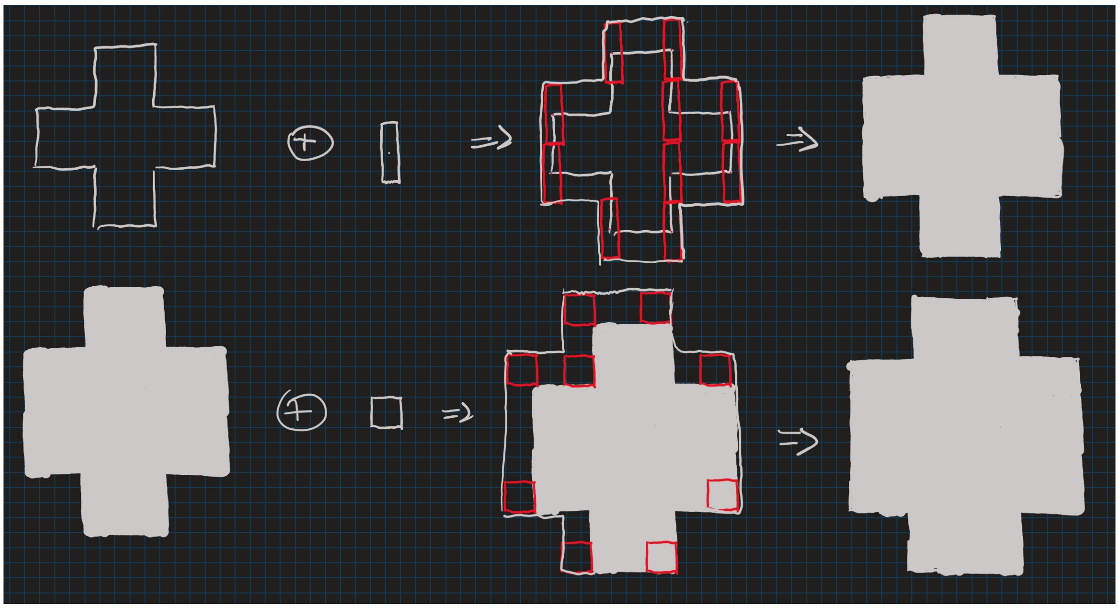


Figure 3: Solution of point c

Exercise 2 – Binary morphological operations

a. The problem required us to write a function `IPdilate` that would perform a dilation of a foreground object in an image with a 3 by 3 structuring element which has its origin in the center of the 3 by 3 matrix. The dilation is an operation in which a foreground object A in a certain image is dilated via a structuring element B such that

$$A \oplus B = \bigcup_{b \in B} A_b = \bigcup_{a \in A} B_a$$

where A_b is the shifting of A over each element $b \in B$. Our solution is presented in listing 6.
We can divide our solution in 3 main parts:

- In the first part, we prepared the mask by defining its "size". We then used a variable `interval` which is the size of the mask divided by two and rounded down. We will use this variable to apply the filter itself. Also here, we added the padding to the image to avoid overstepping the image boundaries.

```
%define mask size
mask_size = 3;

%find interval
interval = (mask_size-1)/2;

%add padding to image
img = ones(size(inImg)+2*interval);
img(interval+1:size(inImg,1)+interval, interval+1:size(inImg,2)+
    interval) = inImg;
```
Listing 1: Prepare the mask.

- In the second part we further prepare a couple of variables for the coming operation. Here we save all the row and column indexes of the white pixels in the image by using the find function. Then, since we applied some padding to the image, we have to "transorm" the pixel coordinate to the new space by adding the previous mentioned variable `interval`.

```
13  %find indexes of elements greater than zero
14  [row,col] = find(inImg>0);
15
16  %convert coordinates to new image space
17  row=row+interval;
18  col=col+interval;
```

Listing 2: Supporting variables.

- In the last part we apply the actual dilation. Here we iterate over all white pixels and we "paint white" all the pixels that the structuring elements overlap with. The overlapping is implemented using the `interval` variable. In fact if we have a 3 by 3 structuring element over a pixel $f(x, y)$, our interval variable would be equal to one and we would be selecting an image area $f(x-1 : x+1, y-1 : y+1)$ which is a 3 by 3 area.

```
20  %apply dilation
21  for i=1:size(row,1)
22      r=row(i,1); c=col(i,1);
23      %for each pixel in the image that is white, we turn to 1 all
            the values
24      %in said interval
25      img( r-interval:r+interval, c-interval:c+interval)=1;
26  end
```

Listing 3: Supporting variables.

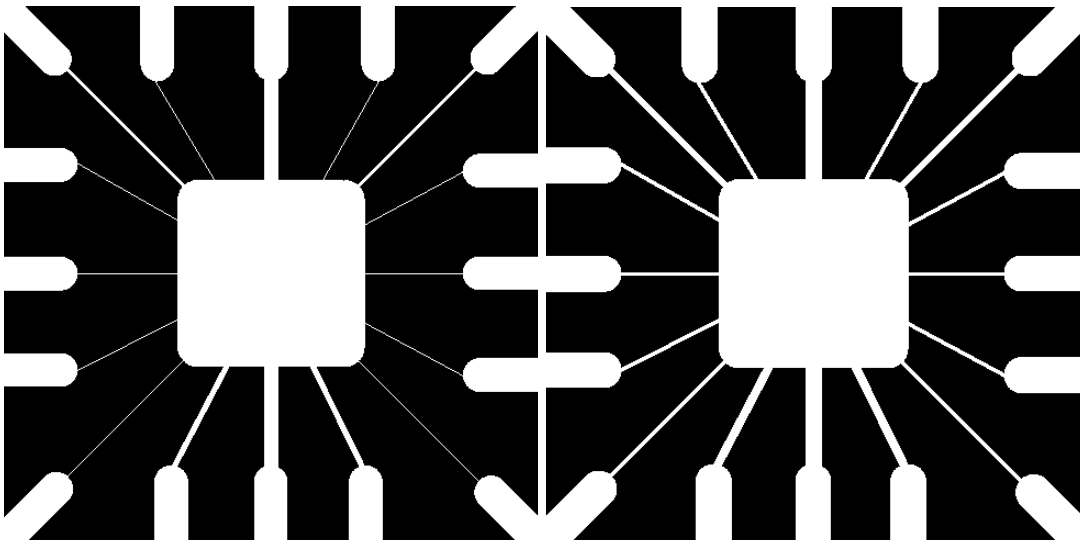
As we can see in figure 4, the dilated image 4b clearly presents stronger lines. In fact those lines were "increases" by two pixels on each side. This is due to the mask being 3 by 3 in size. Bigger masks would imply bigger increments.

b. The problem required us to write a function `IPerode` that would perform an erosion of a foreground object in an image with a 3 by 3 structuring element which has its origin in the center of the 3 by 3 matrix. The erosion is and operation in which a foreground object A in a certain image is eroded by a structuring element B such that

$$A \ominus B = \bigcap_{a \in A} A_{-b}$$

where A_{-b} is the translation of A by $-b$. This operation will retain only the elements of A such that when overlapped with the structuring element B, B would fit into A. Our solution is presented in listing 7. This solution can also be divided into three main parts but since the first two parts (from line 3 to line 18) are identical to the `IPdilate` function discussed before, we will only talk about the difference.

- This time around we are going to specify an output image that is going to store only the pixel we retain



(a) The image: '`wirebond.tif`'   (b) Dilation of '`plant.tif`'

Figure 4: Comparison between original image and dilated image

```
20 %define out image
21 out=zeros(size(inImg));
```

Listing 4: Output image.

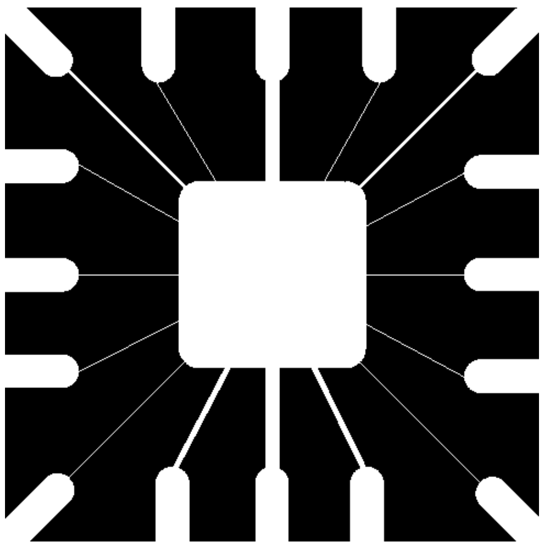
- Here, in the main for loop, we perform the actual erosion. For all the white pixels in the input image we select `f`, the 3 by 3 area around it. If all pixel inside it are white (greater than zero), we retain that pixel by painting it white in the output image on line 30. Also on line 30, since the original image doesn't have any padding, we bring the coordinates back to the original numbering by subtracting the interval.

```
23 %apply erosion
24 for i=1:size(row,1)
25     r=row(i,1); c=col(i,1);
26     f=img(r-interval:r+interval, c-interval:c+interval);
27     %if all element of this section are greater than zero (aka 1)
28     %the mask fit and we keep the pixel
29     if f>0
30         out(r-interval,c-interval)=1;
31     end
32 end
```

Listing 5: Output image.

As we can observe in figure 5, image 5b lost all the weaker lines by only retaining four of them. This imply that the disappeared lines were smaller than 3 pixel wide while the retained ones were bigger than 3 pixel which is the mask size.

c. The last part of this assignment asked us to test our functions. For the sake of clarity presented result with each function in figures 4 and 5. Since we already discussed what we obtained, there won't be any further discussion on the results.


(a) The image: '`wirebond.tif`'

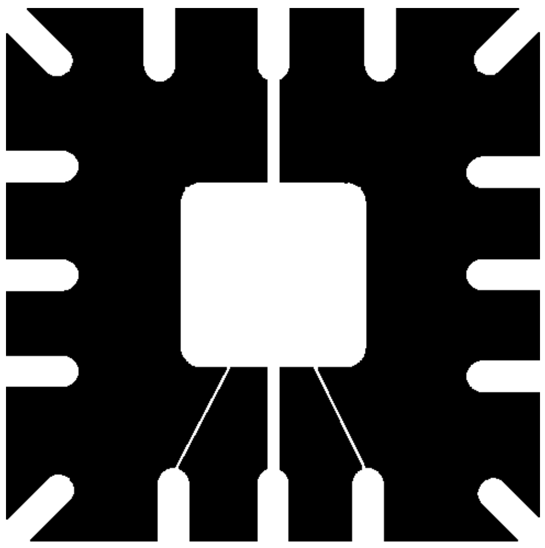

(b) Erosion of '`plant.tif`'

Figure 5: Comparison between original image and eroded image

Exercise 3 – Skeletons

```
function g = IPdilate(inImg)

%define mask size
mask_size = 3;

%find interval
interval = (mask_size-1)/2;

%add padding to image
img = ones(size(inImg)+2*interval);
img(interval+1:size(inImg,1)+interval, interval+1:size(inImg,2)+interval)
     = inImg;

%find indexes of elements greater than zero
[row,col] = find(inImg>0);

%convert coordinates to new image space
row=row+interval;
col=col+interval;

%apply dilation
for i=1:size(row,1)
    r=row(i,1); c=col(i,1);
    %for each pixel in the image that is white, we turn to 1 all the
        values
    %in said interval
    img( r-interval:r+interval, c-interval:c+interval)=1;
end

%return dilated image without padding;
g = img(interval+1:size(inImg,1)+interval, interval+1:size(inImg,2)+
    interval);

end
```

Listing 6: IPdilate

```
function g = IPerode(inImg)

%define mask
mask_size = 3;

%find interval
interval = (mask_size-1)/2;

%add padding to image
img = ones(size(inImg)+2*interval);
img(interval+1:size(inImg,1)+interval, interval+1:size(inImg,2)+interval)
    = inImg;

%find indexes of elements greater than zero
[row,col] = find(inImg>0);

%convert coordinates to new image
row=row+interval;
col=col+interval;

%define out image
out=zeros(size(inImg));

%apply erosion
for i=1:size(row,1)
    r=row(i,1); c=col(i,1);
    f=img(r-interval:r+interval, c-interval:c+interval);
    %if all element of this section are greater than zero (aka 1)
    %the mask fit and we keep the pixel
    if f>0
        out(r-interval,c-interval)=1;
    end
end

%return eroded image
g=out;
end
```

Listing 7: IPerode