25.11.2020

Robot Vision

RV ex5 Image binarization and fundamental operations of mathematical morphology

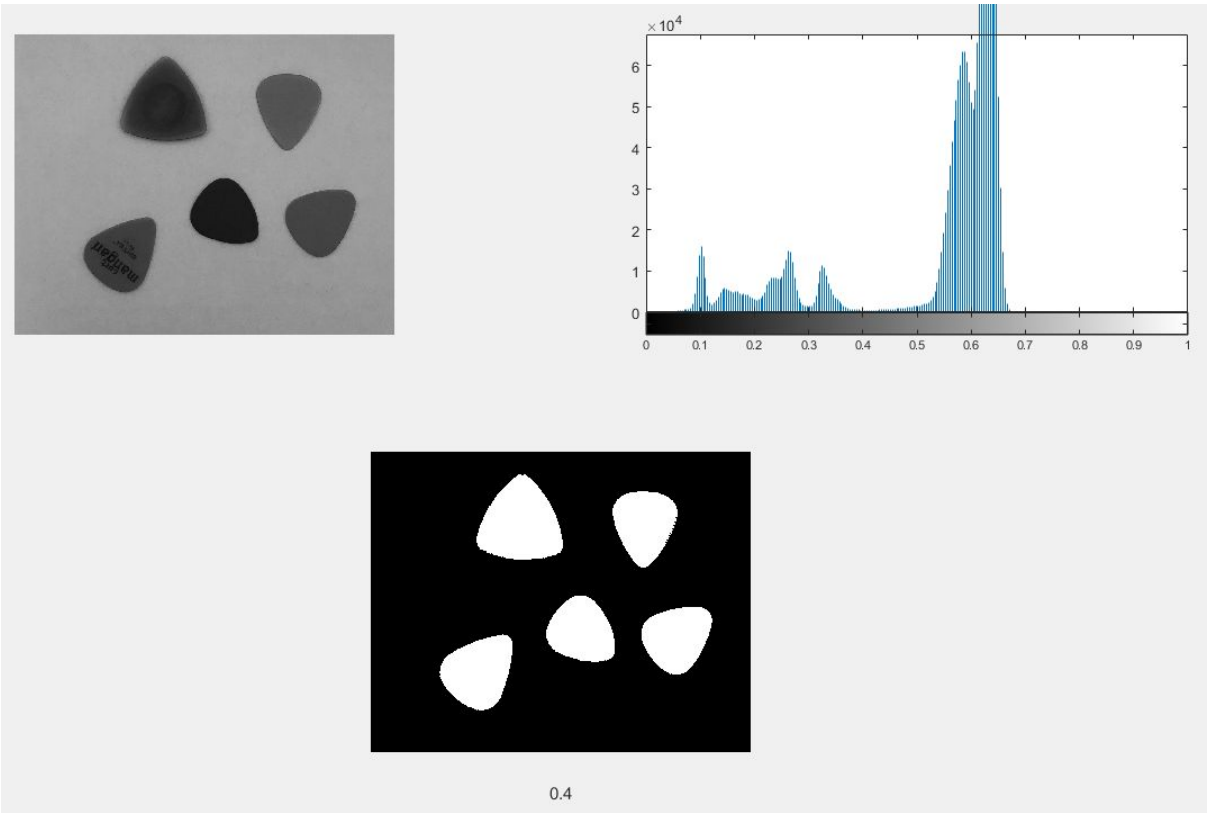
Mateusz Warmuz

25.11.2020

1. Acquire a set of images (at least 2, as usually achromatic 8-bit depth, i.e., 256 possible grey levels, with the resolution of at least 512×512 pixels), which are suitable for binarization. The best images are images presenting simple objects (e.g., text, coins) on a homogeneous background. Images should have varied content, e.g., IMG1 pens, IMG2 rice grains. Acquire means take the pictures yourself.

The second image is not quite good, due to large sizes which causes shadows. Guitar pick is very useful due to flat size.

2. Binarize the acquired images. Compare obtained binarization results with your understanding (interpretation) of these images. Pay special attention to the number of objects, their shapes (including the shape of their contours) and their sizes.

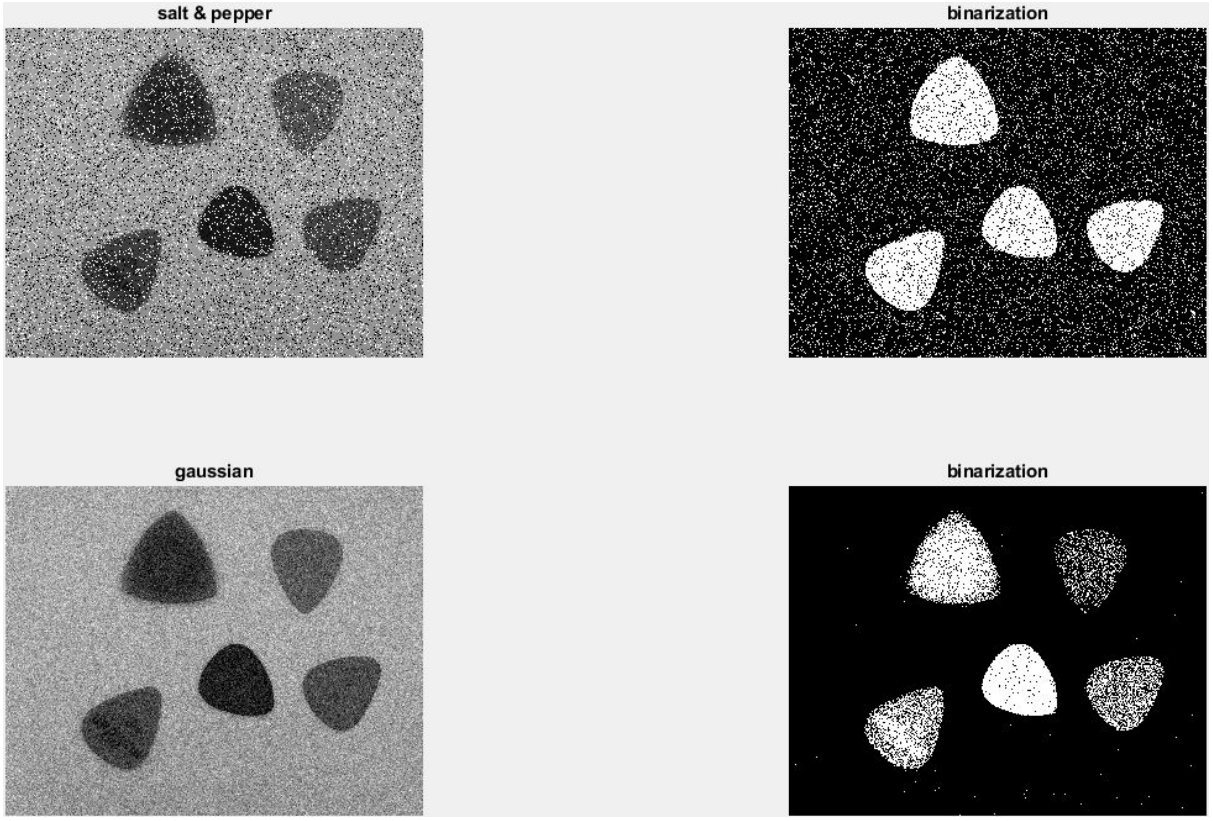


My threshold is 0.4

code:

```
photo=imread('kostki.jpg');
A = rgb2gray(photo);
grey = rgb2gray(photo);
threshold =0.4;

grey = double(grey)/255;

subplot(2,2,1);
imshow(grey);

subplot(2,2,2);
imhist(grey);
B = grey <= threshold;

subplot(2,2,[3,4]);
imshow(B);
xlabel(threshold);
```

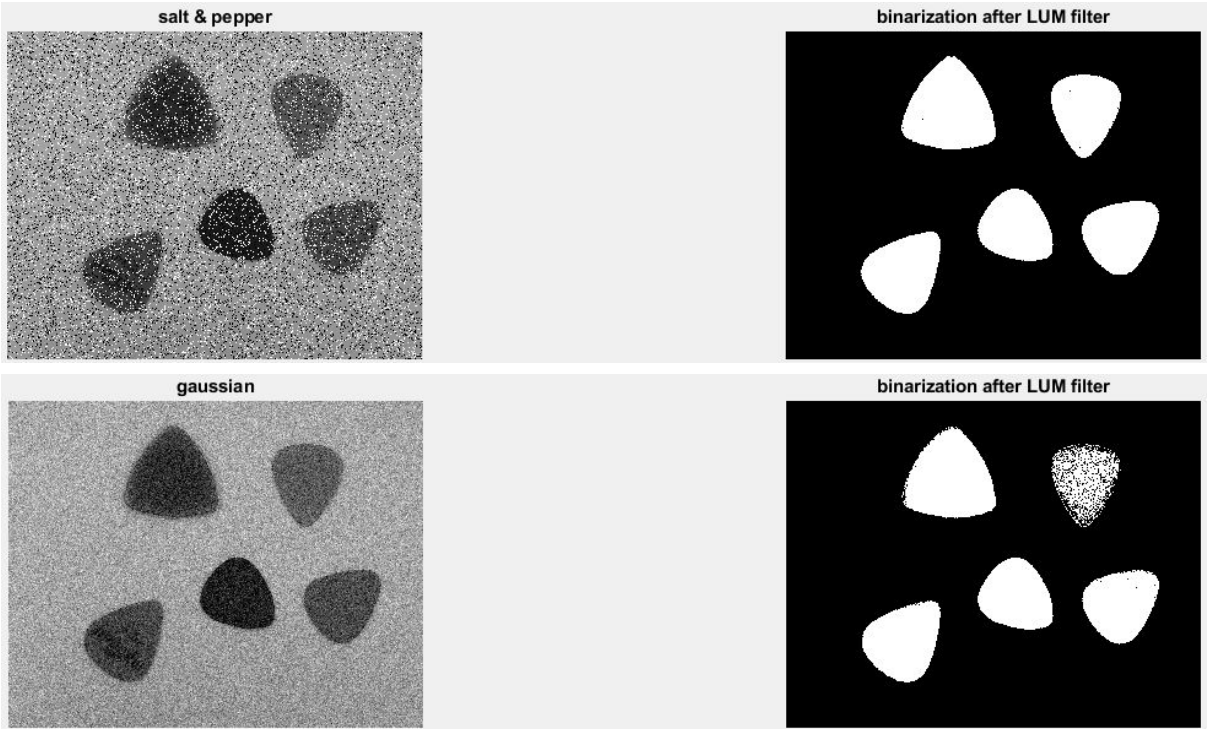
3. Test how image noise (test Gaussian and salt&peprer) influence binarization results. Check if the usage of noise reduction filters change the binarization results.



code:

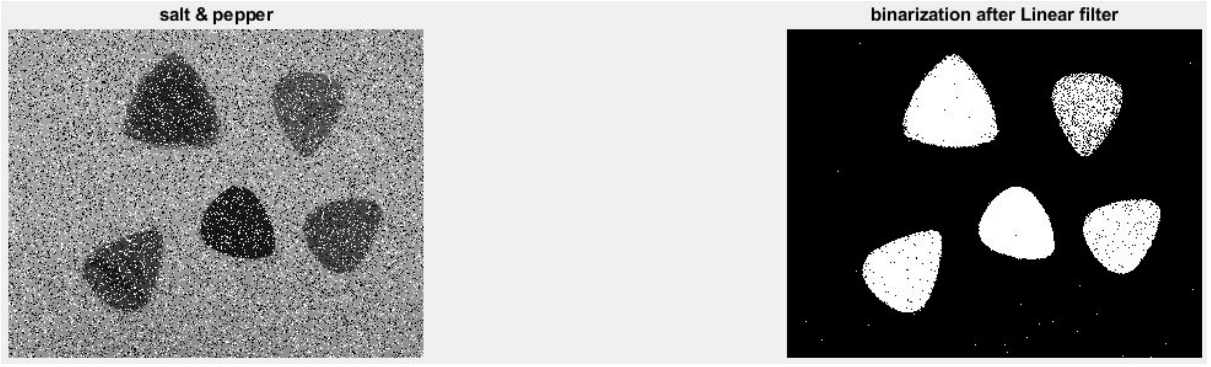
```
photo=imread('kostki.jpg');
A = rgb2gray(photo);
grey = rgb2gray(photo);
threshold =0.4;

grey = double(grey)/255;

greysanp = imnoise(grey,'salt & pepper',0.2);
greygau = imnoise(grey,'gaussian',0.025);

B = grey <= threshold;

sandp = imnoise(grey,'salt & pepper',0.2);
gaussian = imnoise(grey,'gaussian',0.025);

sandp = sandp <= 0.3 ;
gaussian = gaussian <=0.3;


subplot(2,2,1);
imshow(greysanp);
title('salt & pepper');
subplot(2,2,2);
imshow(sandp);
title('binarization');

subplot(2,2,3);
imshow(greygau);
title('gaussian');
subplot(2,2,4);
imshow(gaussian);
title('binarization');
```

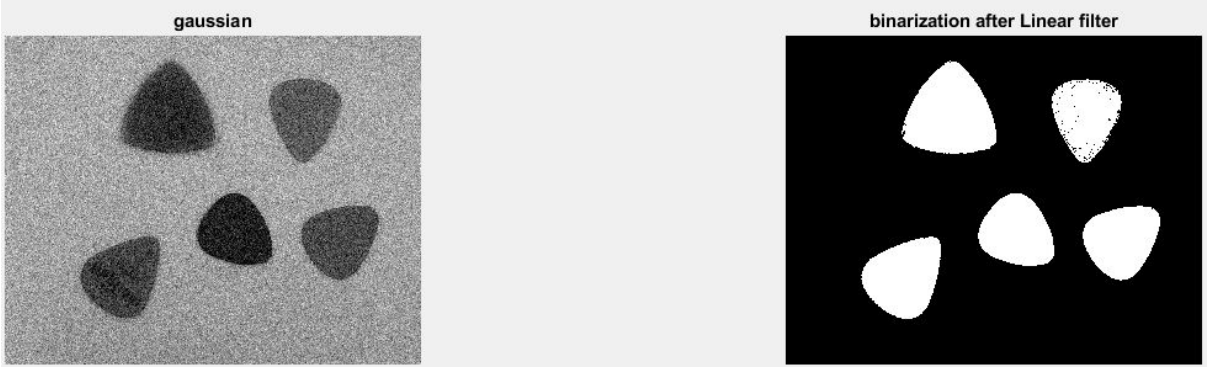
noise reduction filters



code:

```
photo=imread('kostki.jpg');
A = rgb2gray(photo);
grey = rgb2gray(photo);
threshold =0.4;
grey = double(grey)/255;

greysanp = imnoise(grey,'salt & pepper',0.2);
greygau = imnoise(grey,'gaussian',0.025);

B = grey <= threshold;

maskrow = 5;
maskcol = 5;
h = ones(maskrow,maskcol)/25;
s1 = size(h,1)/2+0.5; %row
s2 = size(h,2)/2+0.5; %column
k = 3;
for row = (s1):size(greygau,1)-(s1)
   for col = (s2):size(greygau,2)-(s2)
      B = greygau((row-s1+1):(row+s1-1),(col-s2+1):(col+s2-1));
      x0 = B(s1,s2);
      C = sort(reshape(B,1,[]));
      median = C(size(C,2)/2+0.5);
      l = C((size(C,2)/2+0.5)-k);
      u = C((size(C,2)/2+0.5)+k);
      if x0>l
         x0=l;
      end
      if x0<u
         x0=u;
      end
       D(row-1,col-1) = x0;
   end
end
```



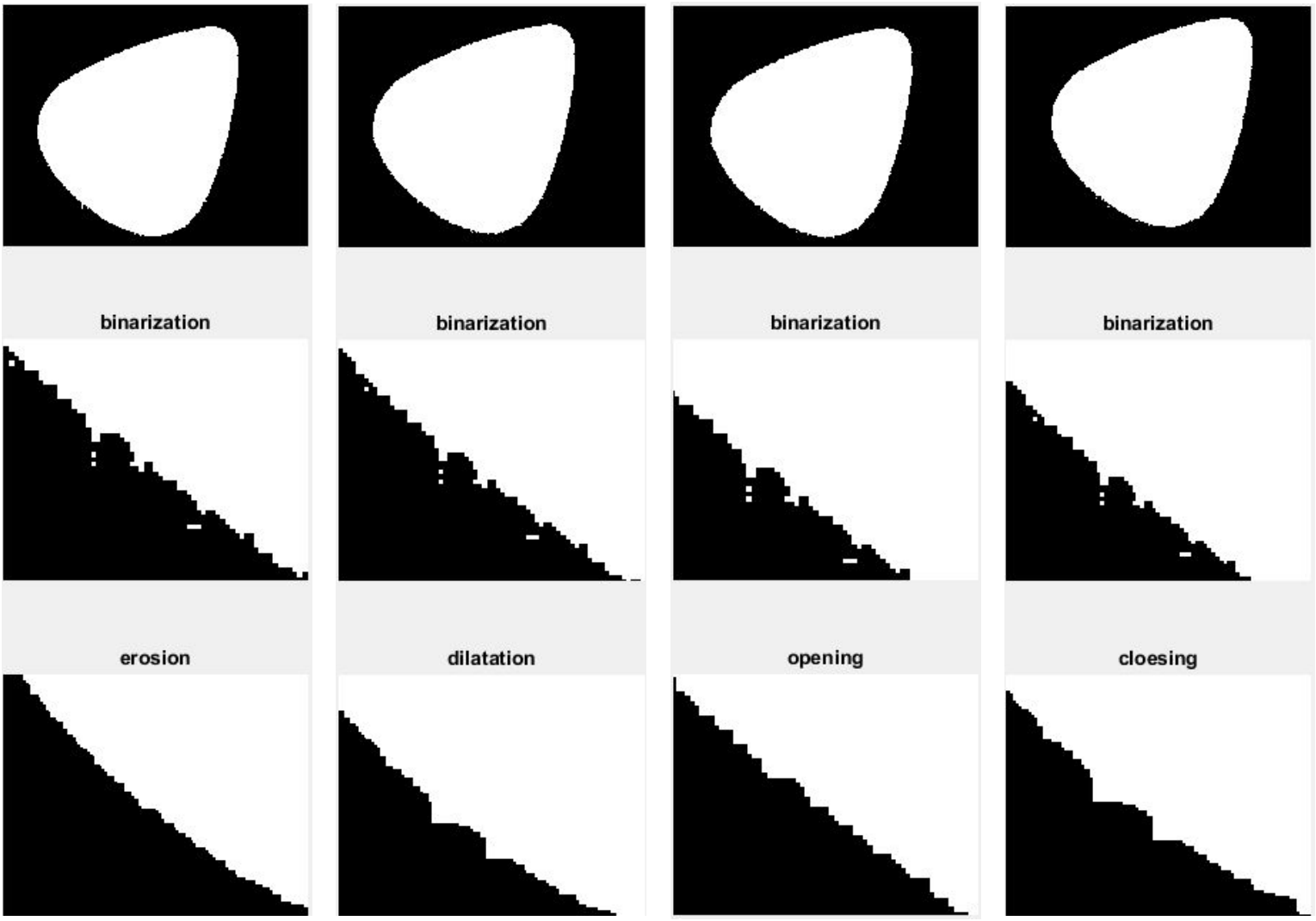


code:

```
photo=imread('kostki.jpg');
A = rgb2gray(photo);
grey = rgb2gray(photo);
threshold =0.4;
grey = double(grey)/255;
greysanp = imnoise(grey,'salt & pepper',0.2);
greygau = imnoise(grey,'gaussian',0.025);

B = grey <= threshold;

maskrow = 5;
maskcol = 5;
h = ones(maskrow,maskcol)/25;
point = size(h,1)/2-0.5;

s1 = size(h,1)/2+0.5; %row
s2 = size(h,2)/2+0.5; %column
k = 3;

for row = (point+1):size(greygau,1)-(point+1)
    for col = (point+1):size(greygau,2)-(point+1)
        grey2(row,col)=conv2(greygau((row-point):(row+point),(col-point):(col+point)),h,
'valid')/sum(h,'all');
    end
end

grey2 = grey2 <= threshold;
subplot(2,2,1);
imshow(greygau);
title('gaussian');
subplot(2,2,2);
imshow(grey2);
title('binarization after Linear filter');
```

We can see that filters cope with noisy images. The best turned out LUM filter which very clearly gives the shape of the object.

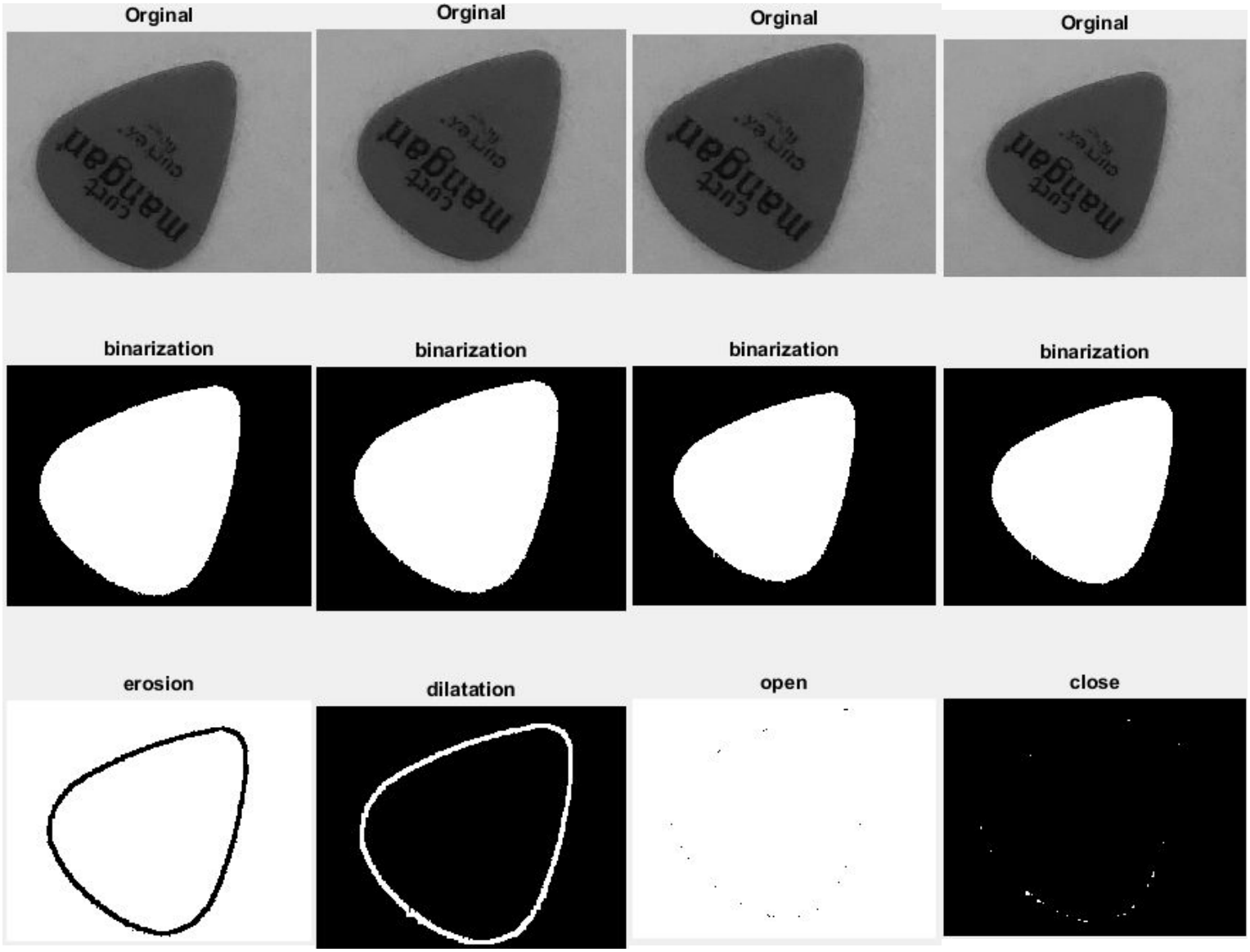
4. Use a sequence of fundamentals operations of mathematical morphology (i.e. erosion, dilatation, opening and closing) and arithmetic operation to improve the previously obtained binarization results. Use different morphological structuring elements.



The dilation operation makes an object grow by size. The extent to which it grows depends on the nature and shape of the structuring element. The erosion operation is a complement of the dilation operation in context with the operation effect. Erosion operation causes objects to lose its size. The opening of an image is a combinational operation of erosion and dilation and closing is the reverse of opening.

25.11.2020

Arithmetic operations

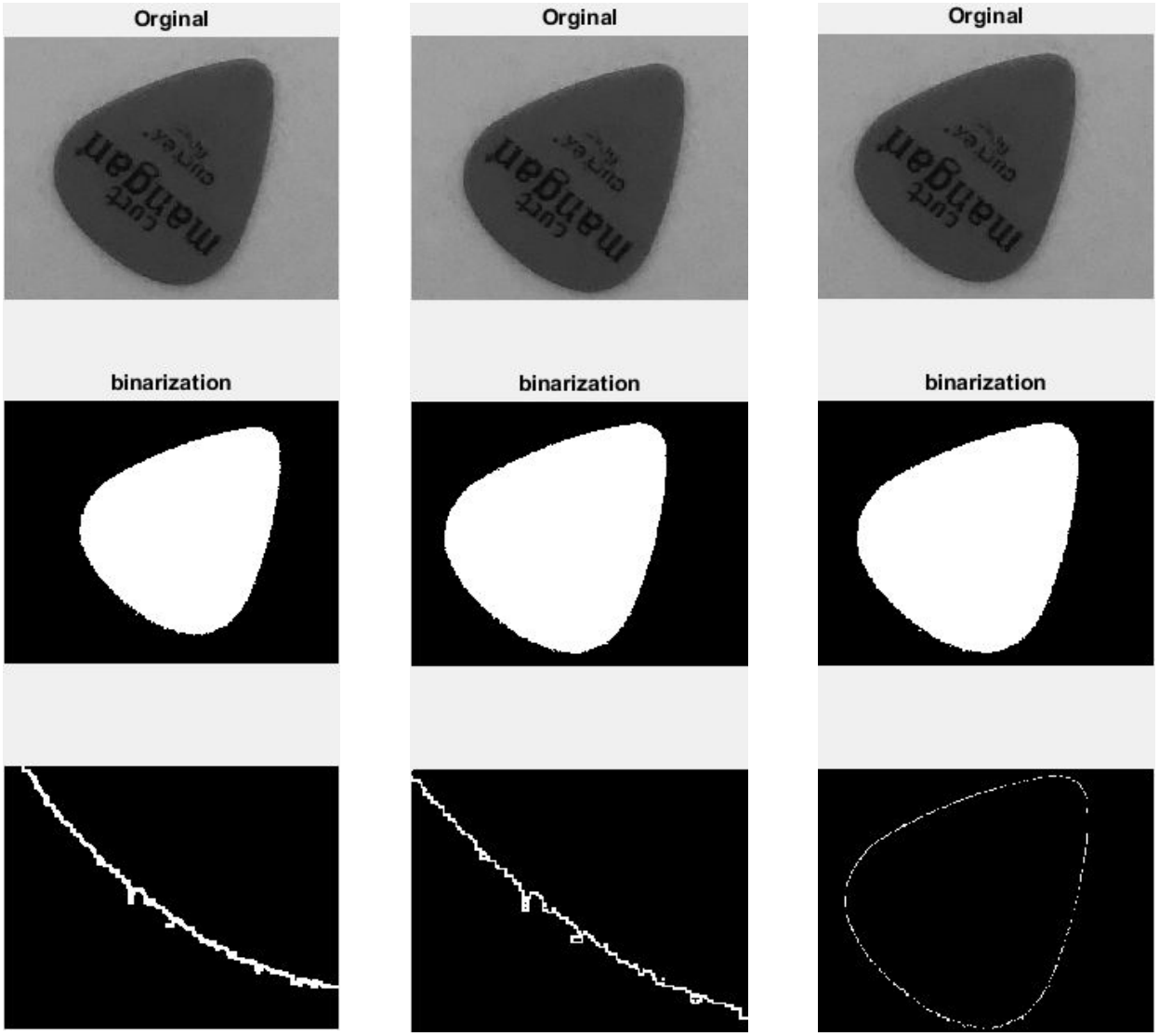


We can notice the difference between how binarize images differ from morphology operations. In erosion we can see how objects lose in size and in dilatation how objects grow. We aren't able to see the difference between opening/closing and binarization. It means that that method is very ‘safe’ in changing the size of the object.

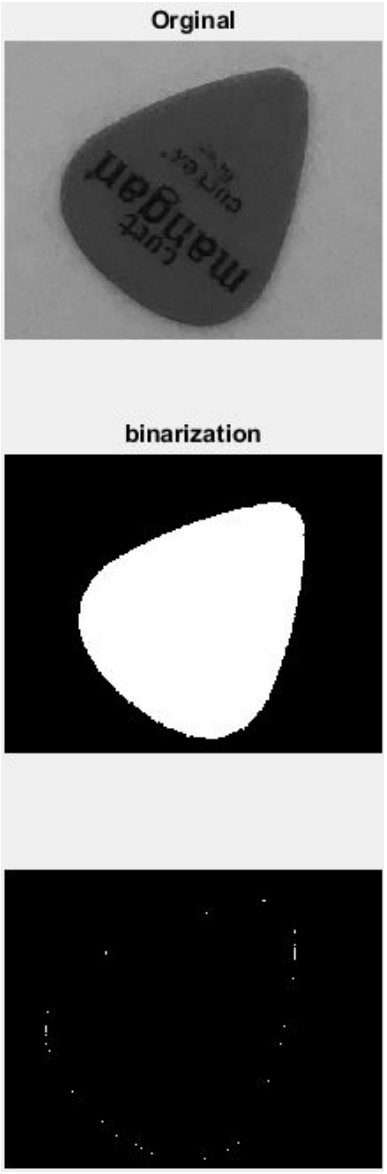
5. Try to formulae some principles of the usage of morphological operations to improve the results of segmentation.

Morphological operations are a very hendfull tool to remove the imperfections in the structure of image. They are used in hole filling, boundary extraction of objects, extraction of connected components, Thinning and thickening and so on. In segmentation, morphological operation will improve the recognition of objects and filter out the noise.

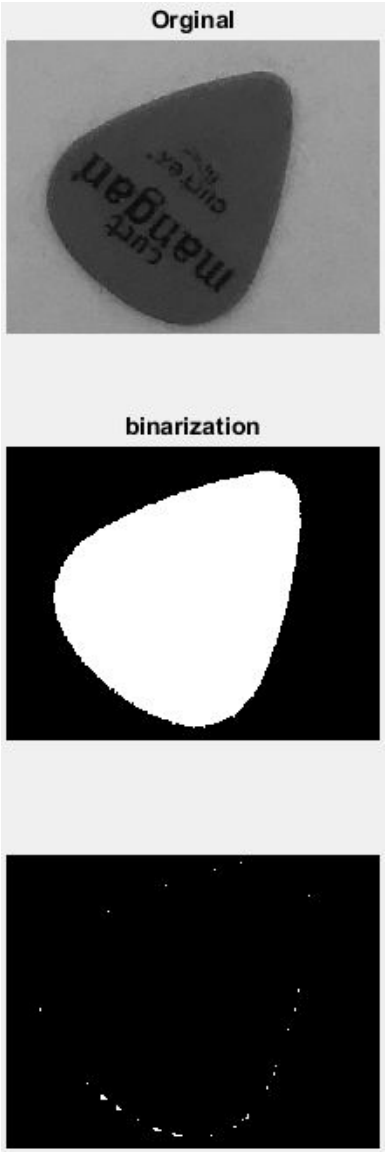
6. Specify which operator is the best edge detector for binary images:
a. $c1 = D(f) - E(f)$, b. $c2 = D(f) - f$, c. $c3 = f - E(f)$,



d. $c4 = f - O(f)$,



e. $c5 = C(f) - f$



In my opinion c. $c3 = f - E(f)$, give the best result. It outlines nicely border without noise as we can see on first two images.

25.11.2020

Code:

```
photo=imread('kostki.jpg');
A = rgb2gray(photo);
grey = rgb2gray(photo);
threshold =0.4;

grey = double(grey)/255;


B = grey <= threshold;
subplot(3,1,1);
imshow(A);
title('Orginal');

subplot(3,1,2);
imshow(B);
title('binarization');


h = ones(15);

E = imerode(B,h);

D = imdilate(B,h);

O = imopen(B,h);

C = imclose(B,h);
subplot(3,1,3);
imshow(C-B);
```