1.12.2020

Robot Vision

RV ex6 Measurements of geometrical features of 2D objects

Mateusz Warmuz

1.12.2020

1. Prepare programs in MATLAB for measuring objects's area and perimeter. For finding object contour you can use methods presented in RV 5, task 6.

Code:

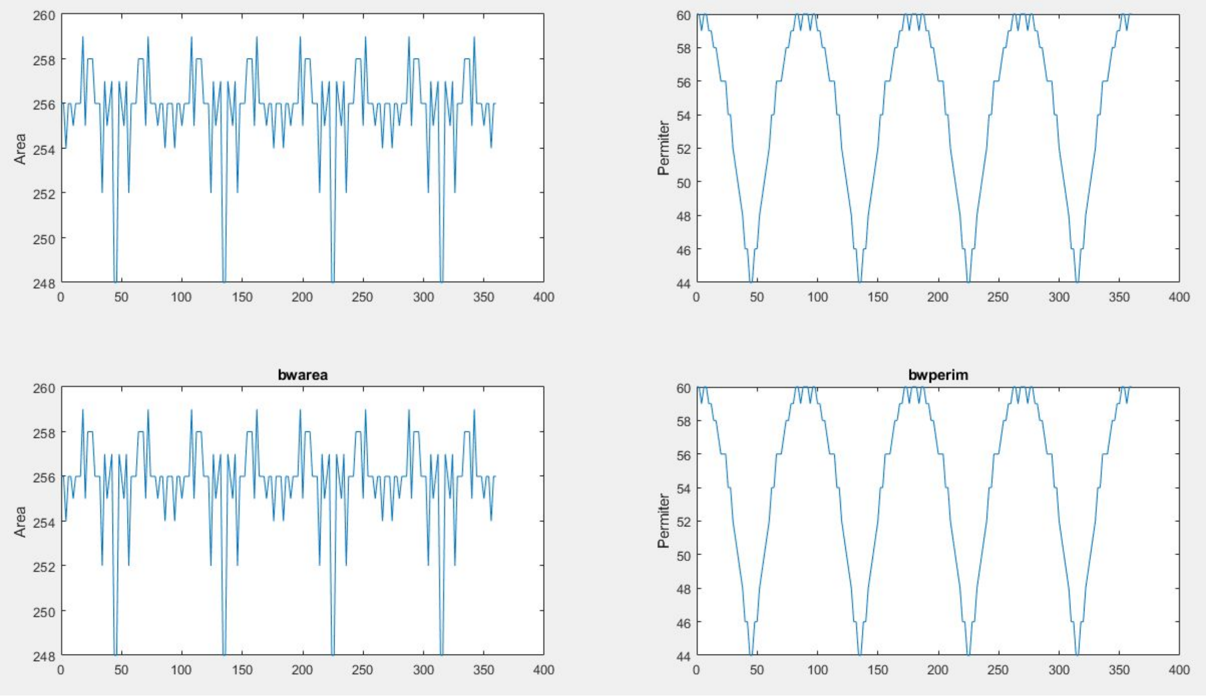
```
photo=imread('kostki_przyciente.jpg');
A = rgb2gray(photo);
grey = rgb2gray(photo);
threshold =0.4;

grey = double(grey)/255;

B = grey <= threshold;


Area = area(B);

se = strel('disk',1);
E = imerode(B,se);

contour = (B-E);
imshow(contour);

Perimeter = perimeter(contour);

function count2 = perimeter(image)
    count2 = 0;
    for row = 1:size(image,1)
        for col = 1:size(image,2)
            if(image(row,col)>0)
                count2 = count2 + 1;
            end
        end
    end
end

function count1 = area(image)
    count1 = 0;
    for row = 1:size(image,1)
        for col = 1:size(image,2)
            if(image(row,col)>0)
                count1 = count1 + 1;
            end
        end
    end
end
```

Calculation of Area = 86886 and Perimeter = 1098

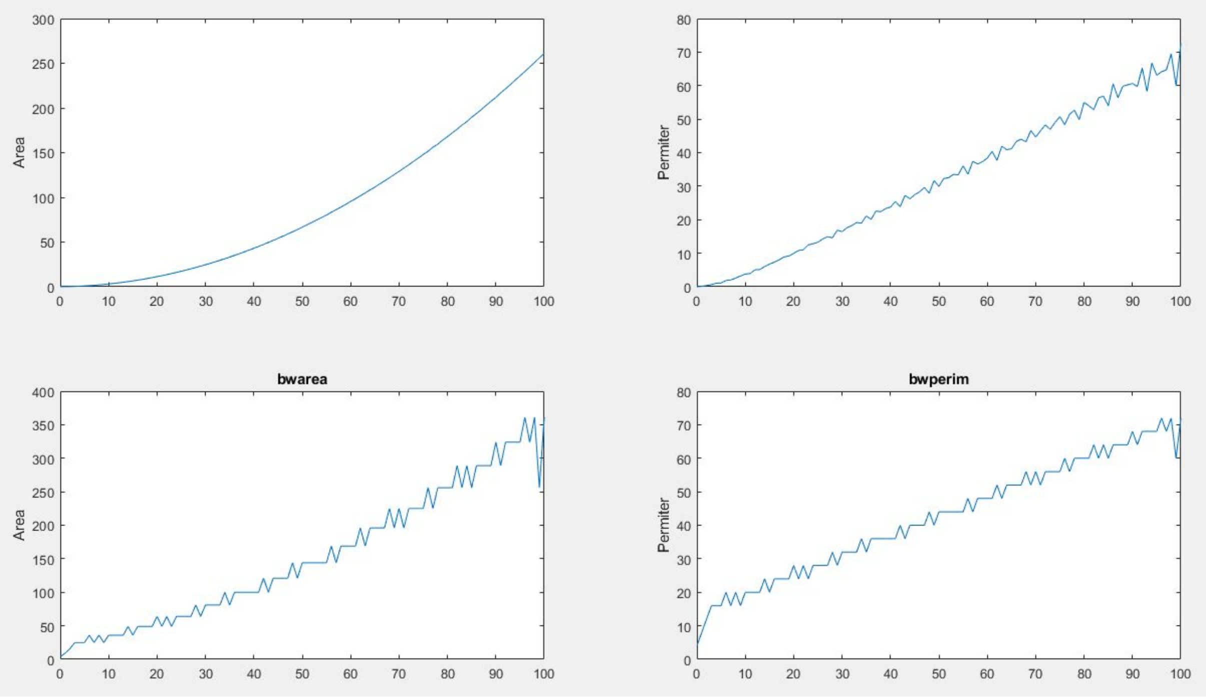
1.12.2020

2. Analyse how a change of object size and its rotation influence on measurements of objects' perimeter and area. Use imrotate and imresize to create charts presenting a change of that measure in a function of angle of rotation and size change. Compare numerical and theoretical values for perimeter and area for simple geometrical figures (e.g. right triangle, rectangle, square, circle). Repeat this analysis using bwarea and bwperim.

Rotation



Resize



1.12.2020

We can observe significant changes in area and perimeter in rotation and resizing objects. First two plots are my method of calculating area and perimeter and two on the bottom are matlab functions (bwarea an bwperim).
We can observe for rotation that the worst perimeter and area is around 43 which this value is really close to 60/sqrt(2). It means that in the worst degree (45, 135, 225 and 315) we have a diagonal which causes such error.
In the resize method we can observe that my method for calculation area and perimeter is much smoother than matlab function. We can observe some strange jumps of area and perimeter in matlab function (bwperim and bwarea).

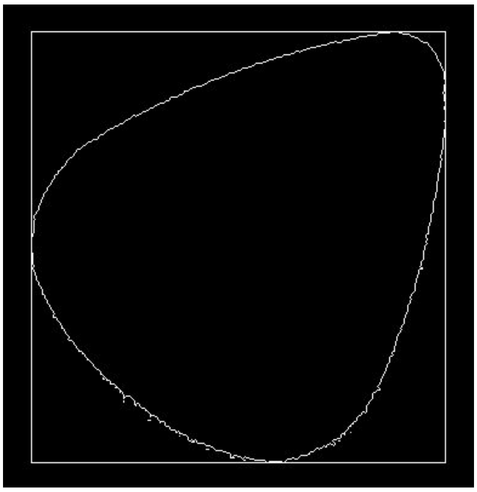
Code:

```
a = 16;

Area_real = a^2;
Permiter_real = 4*a;

background = a + 100;
image = zeros(background);

start = (background-a)/2;
finish = start-1 + a;
image(start:finish,start:finish) = 1;

Area_numericaly = area(image);

se = strel('disk',1);
E = imerode(image,se);

contour = (image-E);

Permiter_numericaly = perimeter(contour);

% rotation

alfa_max = 360;
delta = 2;

vector = 0:delta:alfa_max;
amount = size(vector,2);

vector1 = zeros(1,amount);
vector2 = zeros(1,amount);

for i=1:amount
   degree = vector(i);
   image_rotation = imrotate(image, degree);
   contour_rotation = image_rotation - imerode(image_rotation, se);
   contour_bwperim = bwperim(image_rotation);

   vector1(i) = sum(image_rotation(:));
   vector2(i) = sum(contour_rotation(:));

   vector3(i) = bwarea(image_rotation(:));
   vector4(i) = sum(contour_bwperim(:));

end
```

```
figure;
subplot(2,2,1);
plot(vector,vector1);
ylabel('Area');
subplot(2,2,2);
plot(vector,vector2);
ylabel('Permiter');
subplot(2,2,3);
plot(vector,vector3);
ylabel('Area');
title('bwarea');
subplot(2,2,4);
plot(vector, vector4);
ylabel('Permiter');
title('bwperim');

%resize

vector = 0:100;
amount = size(vector,2);

for i = 1:amount
    image_resize = imresize(image,i/(amount-1));
    contour_resize = image_resize - imerode(image_resize, se);
    contour_bwperim = bwperim(image_resize);

    vector5(i) = sum(image_resize(:));
    vector6(i) = sum(contour_resize(:));

    vector7(i) = bwarea(image_resize(:));
    vector8(i) = sum(contour_bwperim(:));
end

figure;
subplot(2,2,1);
plot(vector, vector5);
ylabel('Area');

subplot(2,2,2);
plot(vector, vector6);
ylabel('Permiter');

subplot(2,2,3);
plot(vector,vector7);
ylabel('Area');
title('bwarea');

subplot(2,2,4);
plot(vector, vector8);
ylabel('Permiter');
title('bwperim');

function count2 = perimeter(image)
    count2 = 0;
    for row = 1:size(image,1)
        for col = 1:size(image,2)
            if(image(row,col)>0)
                count2 = count2 + 1;
            end
        end
    end
end

function count1 = area(image)
    count1 = 0;
    for row = 1:size(image,1)
        for col = 1:size(image,2)
            if(image(row,col)>0)
                count1 = count1 + 1;
            end
        end
    end
end
```

3. Propose and implement in MATLAB an algorithm for calculating following measures of 2D object size:

- Feret box and its horizontal and vertical diameter



Code:

```
countour_rotation = imrotate(contour, 0);


x_max = 0;
x_min = size(countour_rotation,1);

y_max = 0;
y_min = size(countour_rotation,2);



for row = 1:size(countour_rotation,1)
    for col = 1:size(countour_rotation,2)
        if(countour_rotation(row,col) == 1)
            if(row > y_max)
                y_max = row;

            end
            if(row < y_min)
                y_min = row;

            end
            if(col > x_max)
                x_max = col;
            end
            if(col < x_min)
                x_min = col;

            end
        end
    end
end

line = countour_rotation;

line(y_min, x_min:x_max) = 1;
line(y_max, x_min:x_max) = 1;
line(y_min:y_max, x_min) = 1;
line(y_min:y_max, x_max) = 1;

imshow(line);
```

- diagonal of Feret box

Code:

```
diagonal = sqrt((x_max-x_min)^2+(y_max-y_min)^2);
```

The length of Feret box diagonal of my object will be 492.

- diagonal of oriented Feret box

Code:

```
alfa_max = 360;
delta = 1;

vector = 0:delta:alfa_max;
degree = size(vector,2);
mindiag = 1000000;

countour_rotation1 = imrotate(contour, 0);


x_max1 = 0;
x_min1 = size(countour_rotation1,1);

y_max1 = 0;
y_min1 = size(countour_rotation1,2);


for i = 1:degree

    countour_rotation1 = imrotate(contour, i);


    x_max1 = 0;
    x_min1 = size(countour_rotation1,1);

    y_max1 = 0;
    y_min1 = size(countour_rotation1,2);


    for row = 1:size(countour_rotation1,1)
        for col = 1:size(countour_rotation1,2)
```

```
            if(countour_rotation1(row,col) == 1)
                if(row > y_max1)
                    y_max1 = row;

                end
                if(row < y_min1)
                    y_min1 = row;

                end
                if(col > x_max1)
                    x_max1 = col;

                end
                if(col < x_min1)
                    x_min1 = col;

                end
            end
        end
    end

    diag1 = sqrt((x_max1-x_min1)^2+(y_max1-y_min1)^2);

    if(mindiag > diag1)
        mindiag = diag1;
    end

end
```

- object diameter,

Code:

```
X_max = 0;
X_min = size(image,1);
Y_max = 0;
Y_min = size(image,2);
vector = [];  %storing in two element vector position of pixel image
maximalDiameter = 0;

for row = 1:size(image,1)
    for col = 1:size(image,2)
        if(image(row,col) == 1)
            vector = [vector [row,col]'];
        end
    end
end

for row = 1:size(vector,2)
    for col = 1:size(vector,2)
        diam = sqrt((vector(1,row) - vector(1,col))^2 + (vector(2,row) - vector(2,col))^2);
        if diam > maximalDiameter
            maximalDiameter = diam;
            X_max = vector(1,row);
            X_min = vector(1,col);
            Y_max = vector(2,row);
            Y_min = vector(2,col);
        end
    end
```

```
end
```

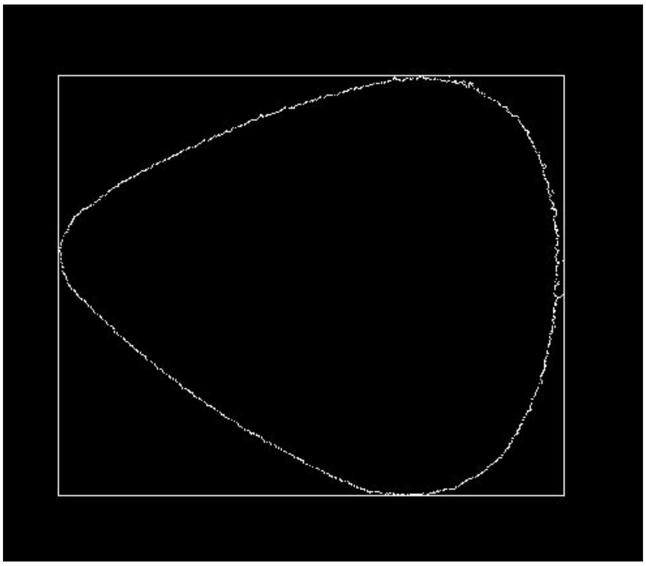
- morphological thickness.

Code:

```
C = B;
count2 = 0;
while sum(C(:))>1
   C = imerode(C, se);
   count2 = count2 + 1;
end
```

with mask se = strel('disk', 1) morphological thickness will be 158.

The bwferet instruction can be used only for comparison purposes. The usage of instructions for object size calculation implemented in the Image Processing Toolbox is forbidden. Check the robustness of the implemented methods on the change of orientation and position of objects.



Comparing my implemented method to bwferet function the results are quite the same.

4. Compare the size-based order of objects for all tested size measures use images from directory IMAGES_2.

	Feret box and its horizontal and vertical diameter	diagonal of Feret box	diagonal of oriented Feret box	object diameter	morphological thickness
circle_1	113,113	159.8061	60.6712	159.8061	13
circle_2	134,134	189.5046	134.9518	189.5046	15
cross	119,119	168.294	120.6690	139.3018	4
dumbbell_1	67,90	112.2007	96.3846	105.5509	5
dumbbell_2	76,86	114.7693	98.5951	107.0187	5
ellipse_1	114,45	122.5602	114.1096	122.9309	6
ellipse_2	60,53	80.0562	60.6712	80.0562	7
ellipse_3	144,88	168.7602	144.3468	168.2914	11
rectangle_1	84,64	105.6030	92.1954	92.3580	3
rectangle_2	95,34	100.9009	95.0842	95.5667	3
square_1	9,9	12.7279	12.7279	12.7279	2
square_2	59,59	83.4386	83.4386	83.4386	8
square_3	179,179	253.1442	253.1442	253.1442	23
trapeze_1	80,76	110.3449	87.7268	109.6221	6
trapeze_2	88,73	114.3372	90.2441	111.0720	6

5. Explain how the morphological thickness depends on the size and shape of the mask (structuring element). Define the type of objects for which the morphological thickness is such a good size measure as others' features.

We have different shapes of masks which cause different calculations of morphological thickness. For example the fastest will be a sphere mask and the worse turned out line mask. Of course not only the mask metter, also size plays a significant role in thickness. When we increase the thickness which causes a small amount of operations. So the principle is that dilation by some large structuring elements can be computed faster by dilation with a sequence of smaller structuring elements.