Region (shape) representation

Shape Features

Topological

- connectivity
- Euler number
- number of holes
- skeleton

Geometric

- perimeter
- area
- max-min radii
- roundness
- symmetry

Moment-based

- centre of mass
- X, Y Feret
- bounding rectangle
- best-fit ellipse

© IE PŁ

Pixel neighbourhood on a grid

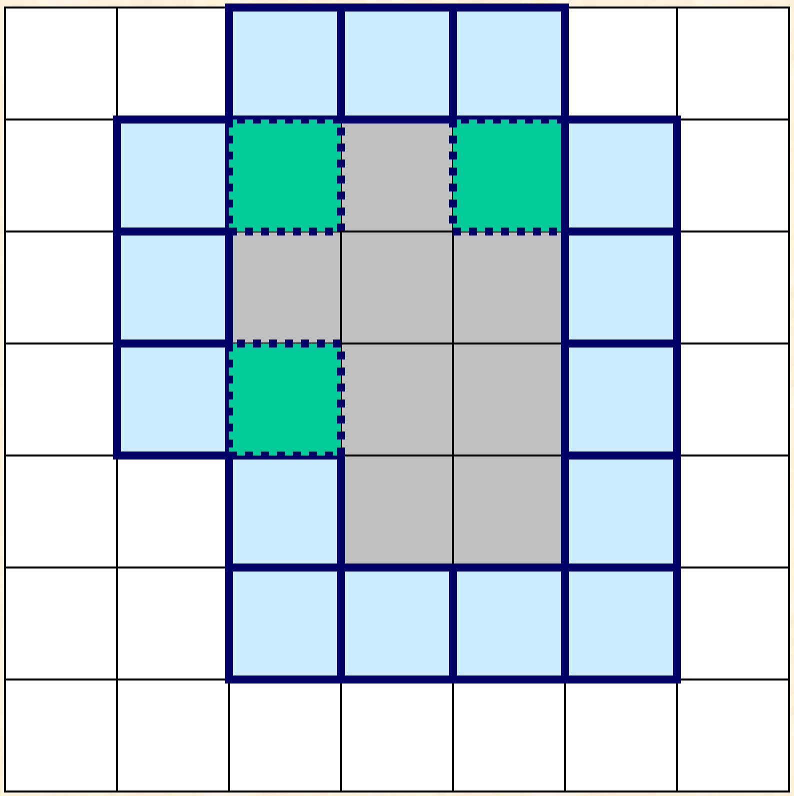
5	6	7
4	x(i,j)	0
3	2	1

Direct neighbour
(D-neighbour) of x(i,j): pixels that share common sides with x(i,j), i.e., [0, 2, 4, 6].

Non-direct neighbours
(N-neighbour) of x(i,j) : pixels that share common vertices with x(i,j), i.e., [1, 3, 5, 7].

© IE PŁ

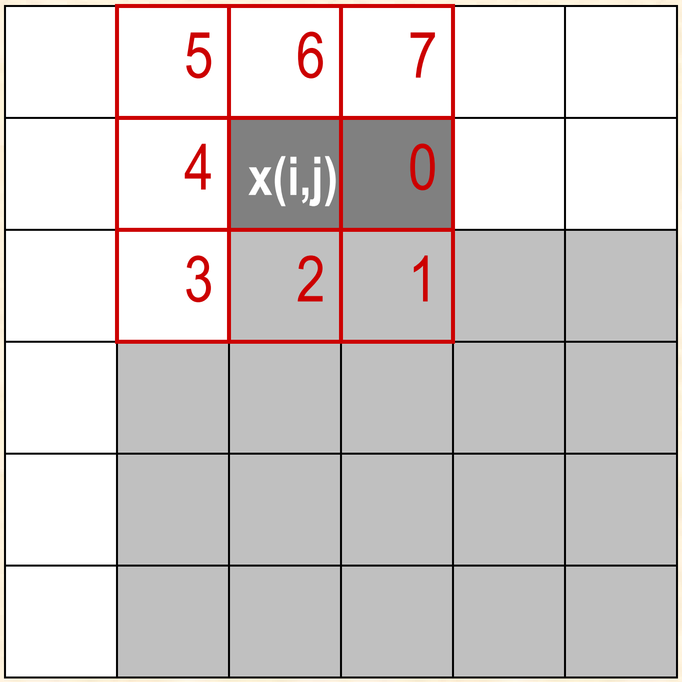
Boundary detection



D-contour: pixels belonging to the object that have at least one neighbour that does not belong to the object

Contour (N-contour): pixels belonging to the object that have at least one D-neighbour that does not belong to the object

Boundary tracking



Coordinates of the next contour pixel:

s=0 -> i:=i+1;
s=1 -> i:=i+1; j:=j+1;
s=2 -> j:=j+1;
s=3 -> i:=i-1; j:=j+1;
...

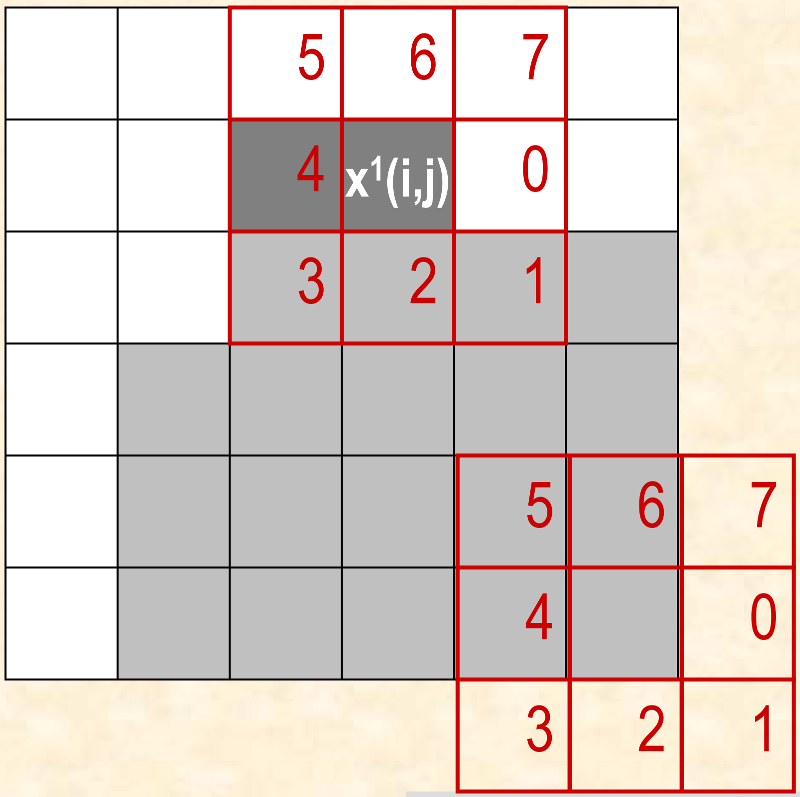
Indication of the neighbour for which search for the next contour pixel starts:

s=0 -> s:=5;
s=1 -> s:=6;
s=2 -> s:=7;
s=3 -> s:=0;
...

Indication of the neighbour for which the next contour pixel resides

Boundary tracking

Next_ij(s,i,j)



Next_s(s,i,j)

Coordinates of the next contour pixel:

s=0 -> i:=i+1;

s=1 -> i:=i+1; j:=j+1;

s=2 -> j:=j+1;

s=3 -> i:=i-1; j:=j+1;

...

Indication of the neighbour for which the next contour pixel resides

Indication of the neighbour for which search for the next contour pixel starts:

s=0 -> s:=5;

s=1 -> s:=6;

s=2 -> s:=7;

s=3 -> s:=0;

...

Boundary tracking - algorithm

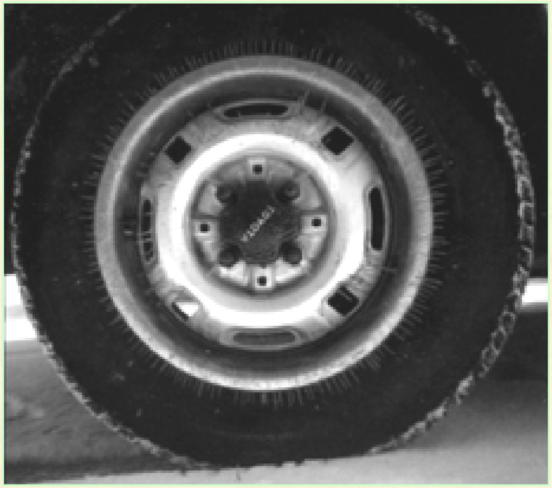
```
{ Contour – number of contour pixels;
Contour_tab – table containing sequence of contour pixels'
neighbourhoods
i,j – coordinates of the starting pixel of the contour}
....
s := 0;
s := Next_s(s,i,j); Next_ij(s,i,j);
Contour := 0; Start_i := i; Start_j := j; Start_s :=s ;
repeat
      s := Next_s(s,i,j);
     Next_ij(s,i,j);  Inc(Contour); Contour_tab[Contour] := s;
until (s=Start_s) and (i=Start_i) and (j=Start_j);
....
```

© IE PŁ

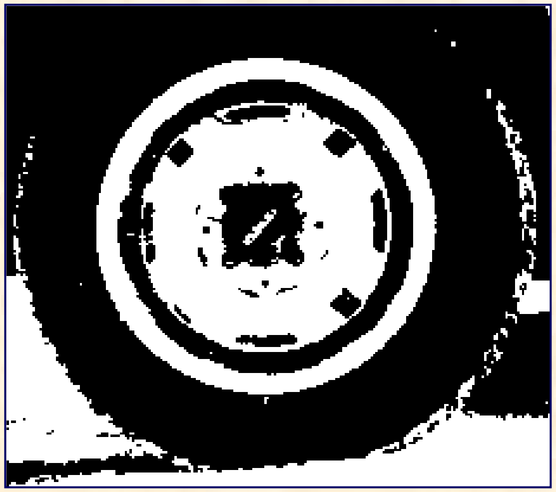
Boundary detection - MATLAB

```
%MATLAB
x = imread('tire.tif');
BW1=im2bw(x,0.2);         %image thresholding
BW2 = bwperim(BW1,8);  %n-contour 8-connected neighbourhood
imshow(x);
figure, imshow(BW1)
figure, imshow(BW2)
```

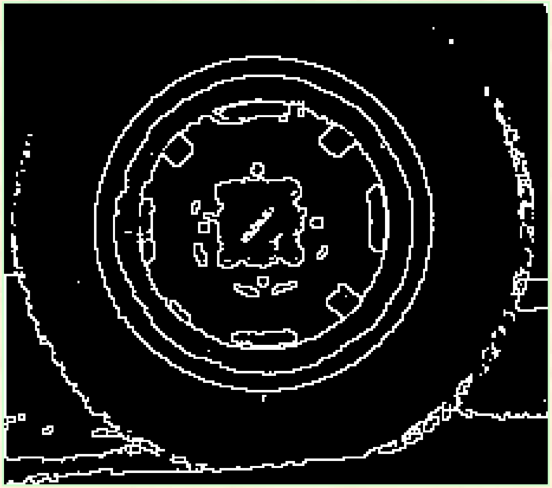
x



BW1



BW2



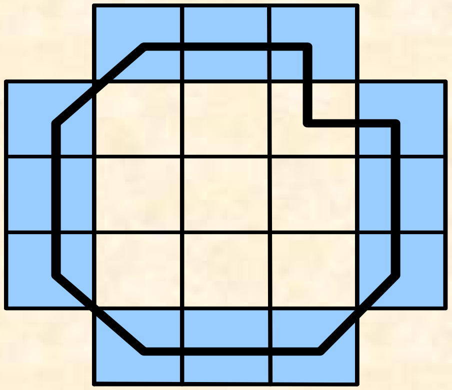
M. Strzelecki, P. Strumiłło

Geometric Features

Perimeter - length of objects boundary

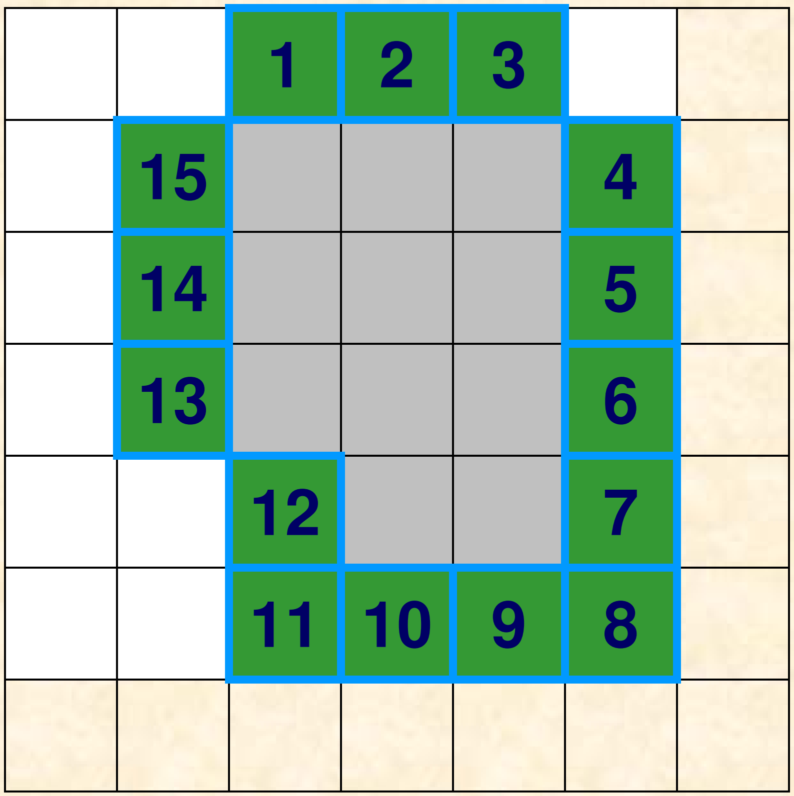
$$T = \int \sqrt{x^2(t) + y^2(t)} dt$$

For a discrete grid contour length is not just the number of boundary pixels!



Contour length

1. Contour length is the number of boundary pixels

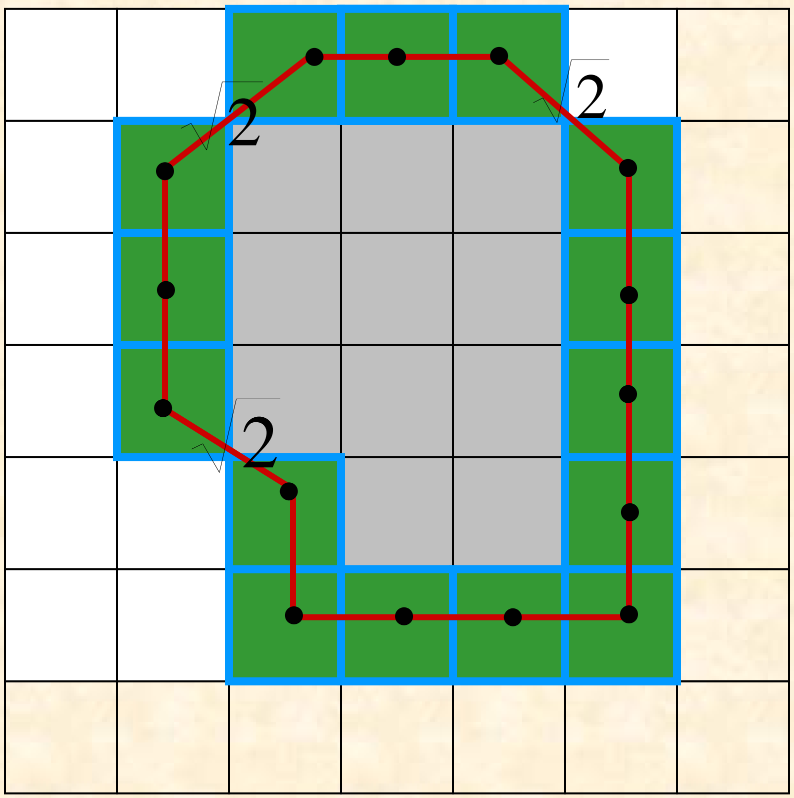


$L = 15$

Expect large error !

Contour length

2. Contour length is the sum of line segments lenghts connecting pixel centres. Pixel size is 1 x 1.



$$L = 12+3\sqrt{2}$$

A better method than the first.

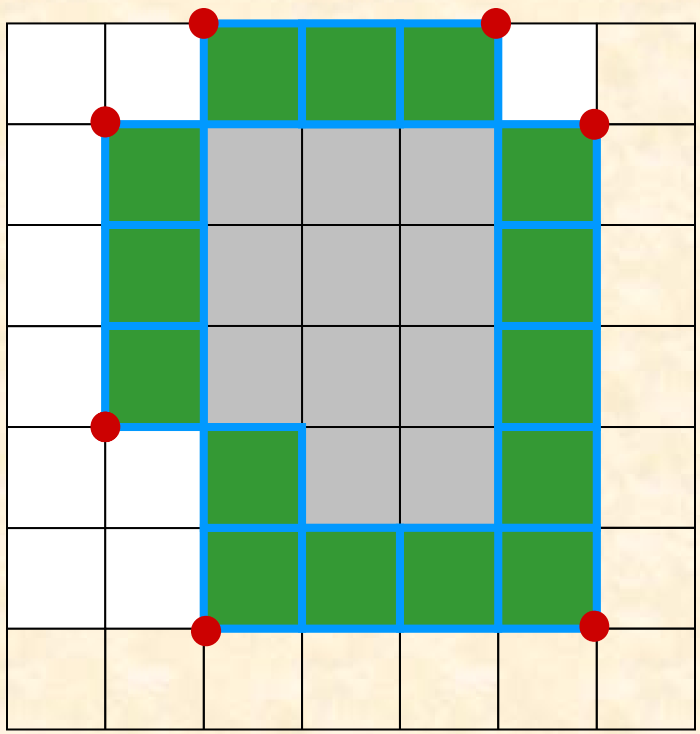
Contour length

3. Contour length is estimated from:

$$\mathbf{O = \mathit{a}N_B - \mathit{b}N_W}$$

$\mathbf{N_B}$ - number of external sides of contour pixels

$\mathbf{N_A}$ - number of contour vertices



$$a = \frac{\pi(1+\sqrt{2})}{8} \qquad b = \frac{\pi}{8\sqrt{2}}$$

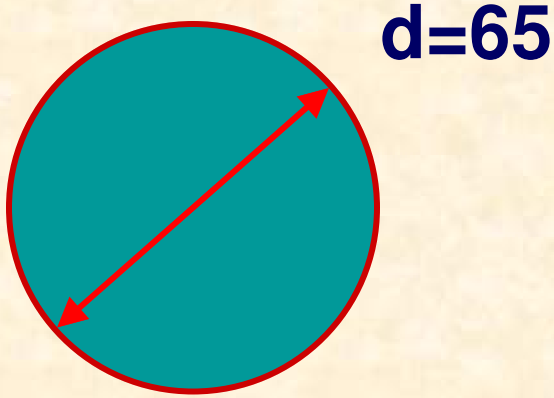
$$\boxed{L = 22a - 7b = ?}$$

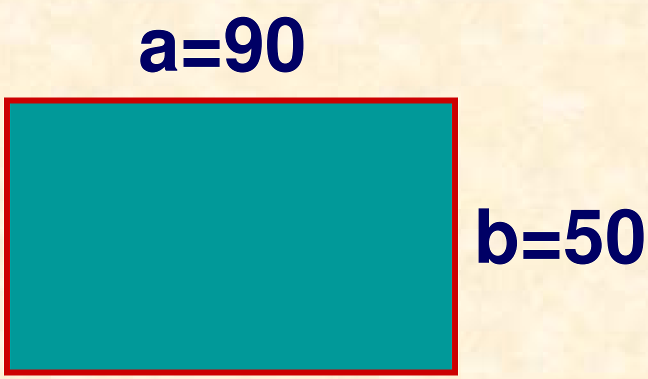
This is an optimum method for a hypothetical shape having boundaries in all directions.

© IE PŁ

Comparison of methods

METHOD	(1)	(2)	(3)	True length
Rectangle	276,0	276,0	263,2	280
Circle	180,0	211,5	203,5	204,2





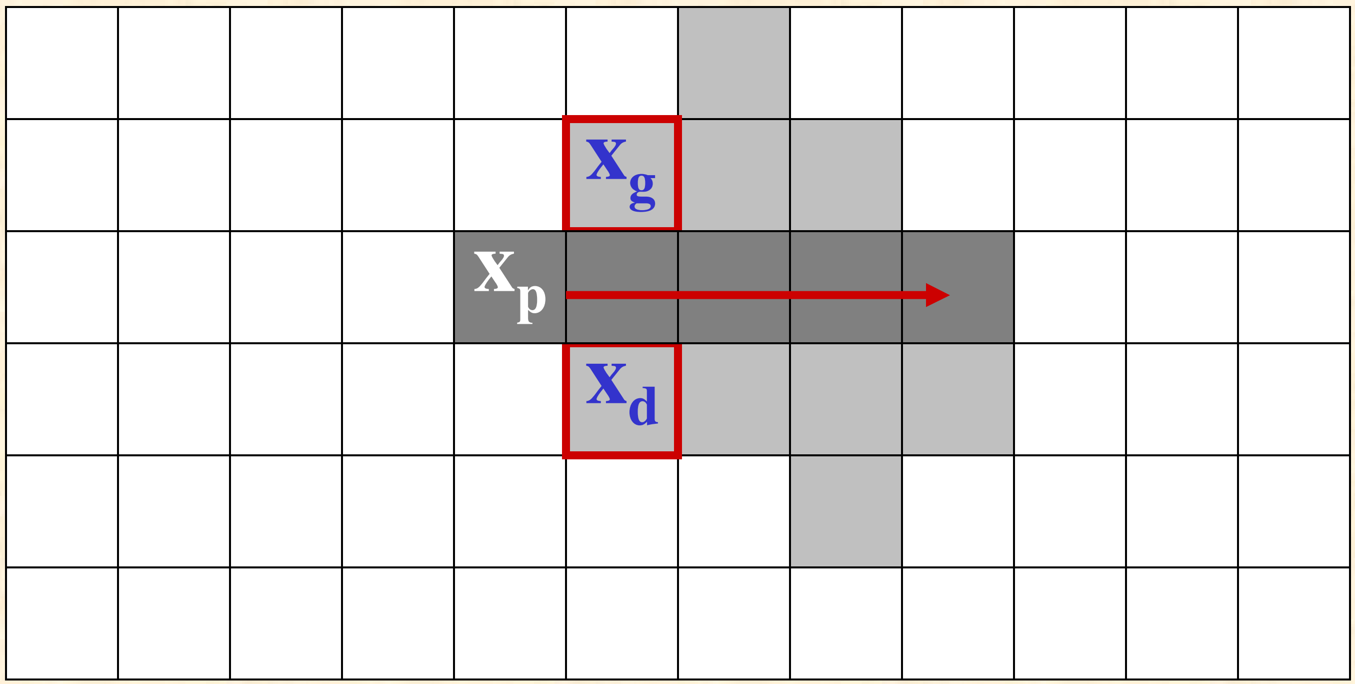
Area

$$A = \iint_R dx\,dy = \int_{\partial R} y(t)\frac{dx(t)}{dt}dt - \int_{\partial R} x(t)\frac{dy}{dt}dt$$

where R and ∂R denote object region and its boundary, respectively (e.g., for a circle of unit radius $x(t)=sin(t)$, $y(t)=cos(t)$).

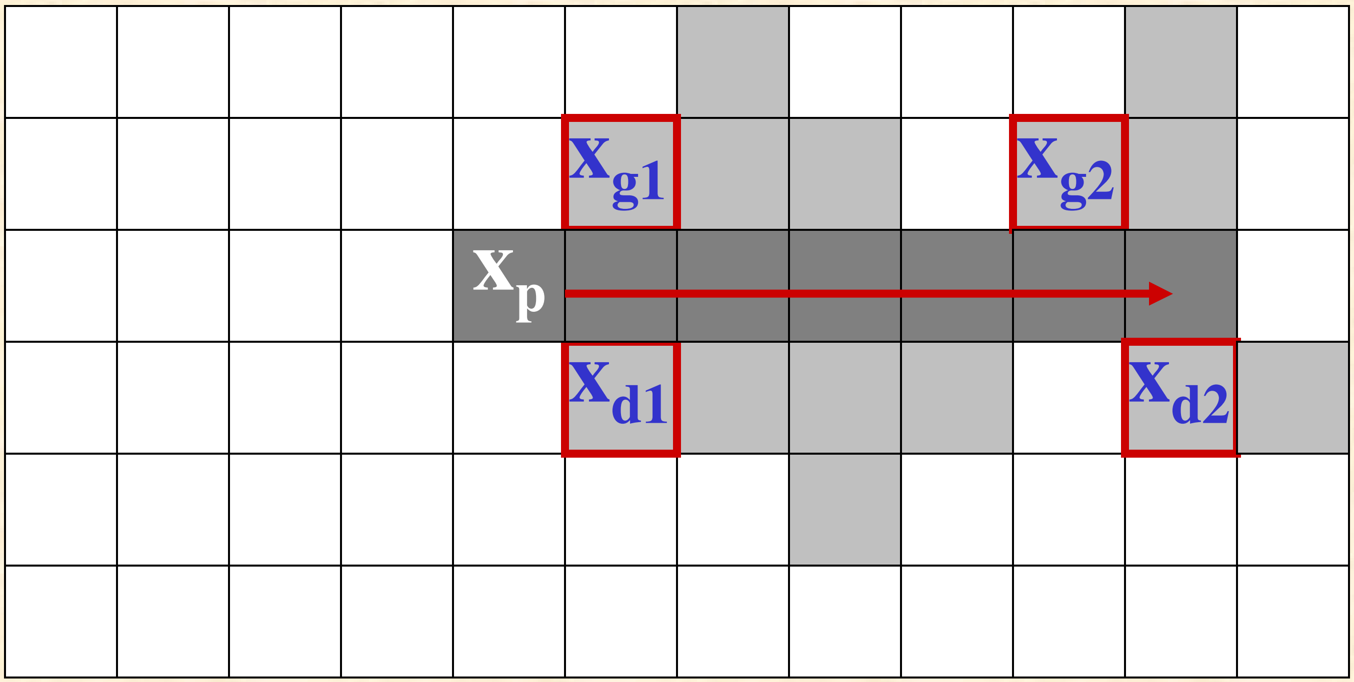
%MATLAB
help bwarea

Region filling algorithm



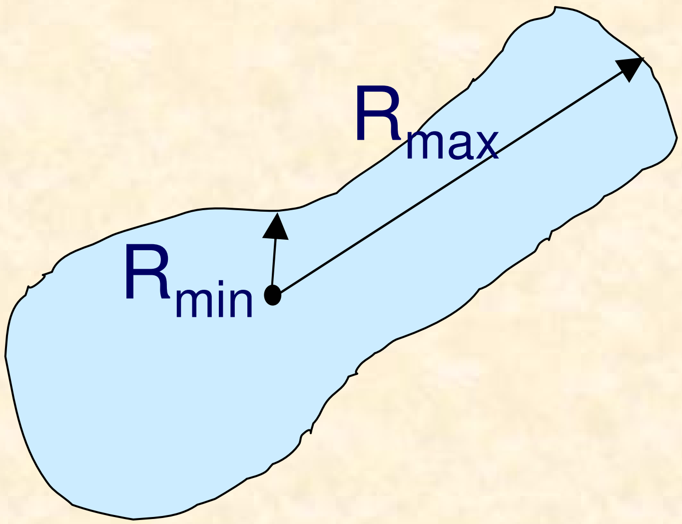
Coordinates x_g, x_d, are stored and filled after finishing with first filled row etc.

Region filling algorithm- concave regions



Concave object ?

Radii



Radii R_{min}, R_{max} are the minimum and maximum distances, respectively, to the boundary from the centre of region mass. The ratio R_{max}/R_{min} (sometimes called object aspect ratio).

Compactness

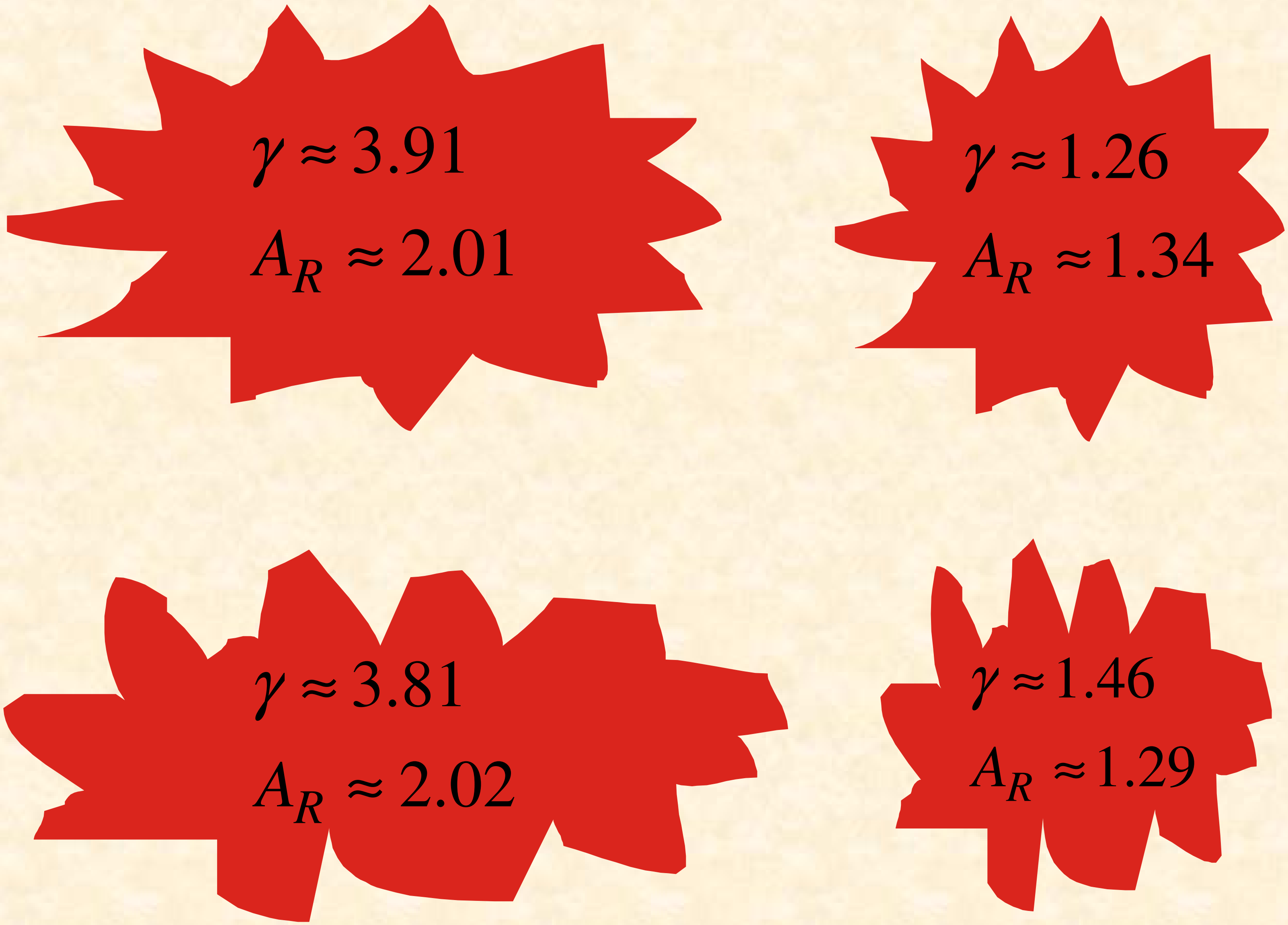
Roundness (compactness) - is a measure of how region shape is different from a circular shape

$$\boxed{\gamma = \frac{(\text{boundary length})^2}{4\pi(\text{area})}}$$

For a circular boundary γ is minimum and equals 1, e.g. for a square $\gamma_{\square} = 4/\pi > 1$.

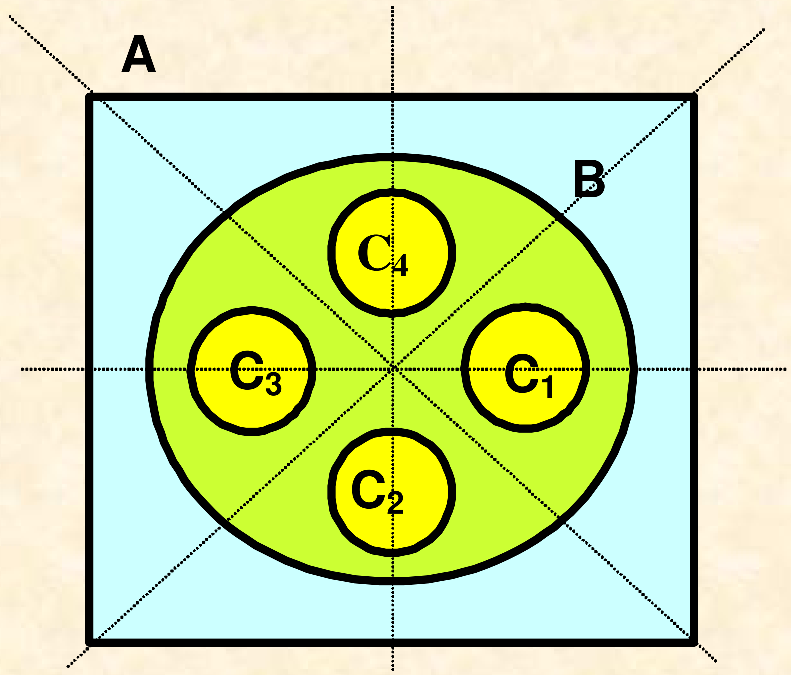
© IE PŁ

Compactness coefficients



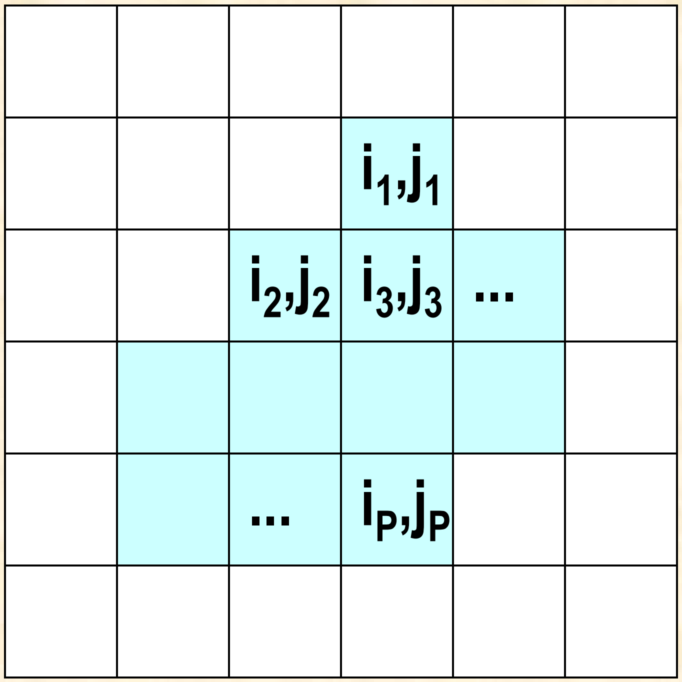
Symmetry

There are two common types of symmetry of shapes, rotational (radial) and mirror. Other types of symmetry are two-fold, four-fold, etc.



Square A has 4-fold symmetry, Circle B is rotationally symmetric, Small circles C_1, … C_4 have 4-fold symmetry

Centre of mass (centroid)

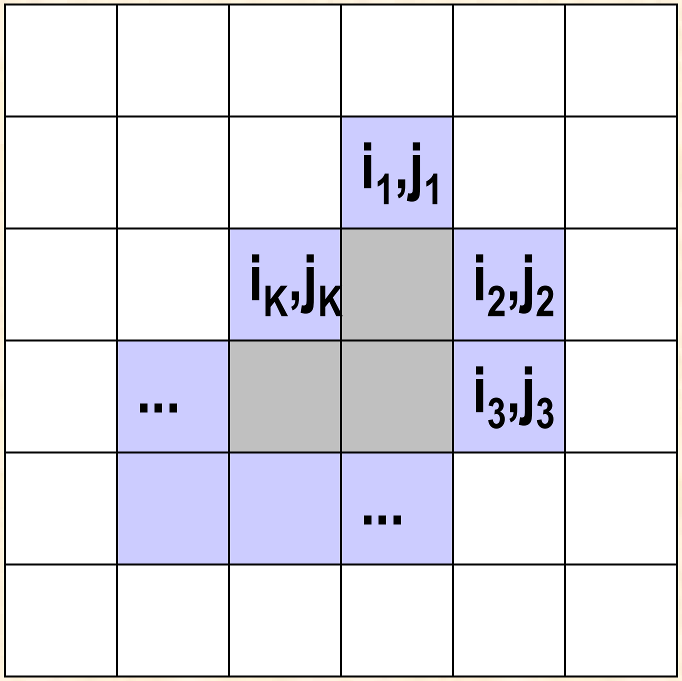


$$SC_i = \frac{1}{P}\sum_{k=1}^{P} i_k, \qquad SC_j = \frac{1}{P}\sum_{k=1}^{P} j_k$$

$$P - number\ object\ \ pixels$$

Note that centre of mass coordinates can be non-integer numbers

Centre of mass (centroid)



$$SC_i = \frac{1}{K}\sum_{l=1}^{K} i_l, \qquad SC_j = \frac{1}{K}\sum_{l=1}^{K} j_l$$

$$K - number\ \ of\ \ contour\ pixels$$

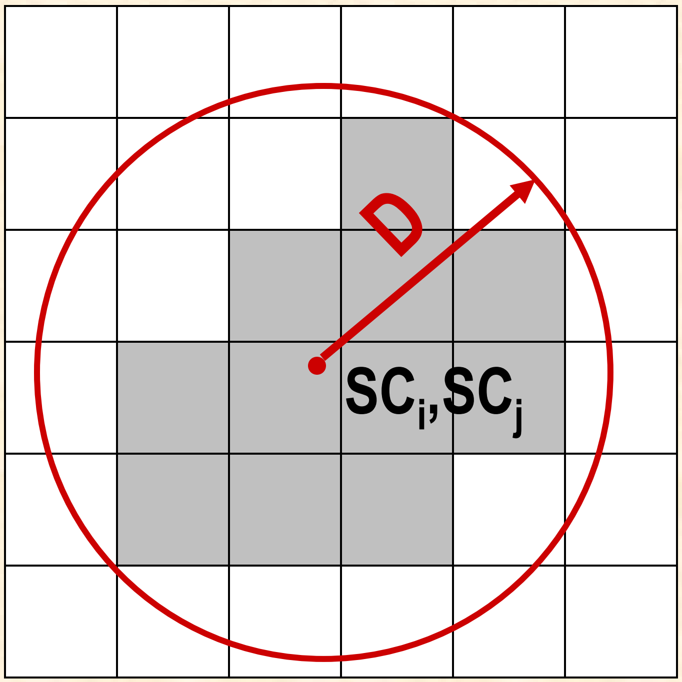
Centre of mass is calculated just on the basis of the contour points

M. Strzelecki, P. Strumiłło

Calculation of the centroid

```
{ Contour – number of contour pixels;
Contour_tab – table containing sequence of contour pixels' neighbourhoods
i,j – coordinates of the starting pixel of the contour}
....
XCenter:=0; YCenter:=0;
for n:=1 to Contour do
begin
   Xcenter := Xcenter + i; Ycenter := Ycenter + j;
   Next_ij(Contour_tab[n],i,j)
end;
XCenter:=XCenter div Contour;
YCenter:=YCenter div Contour;
....
```

Maximum diameter



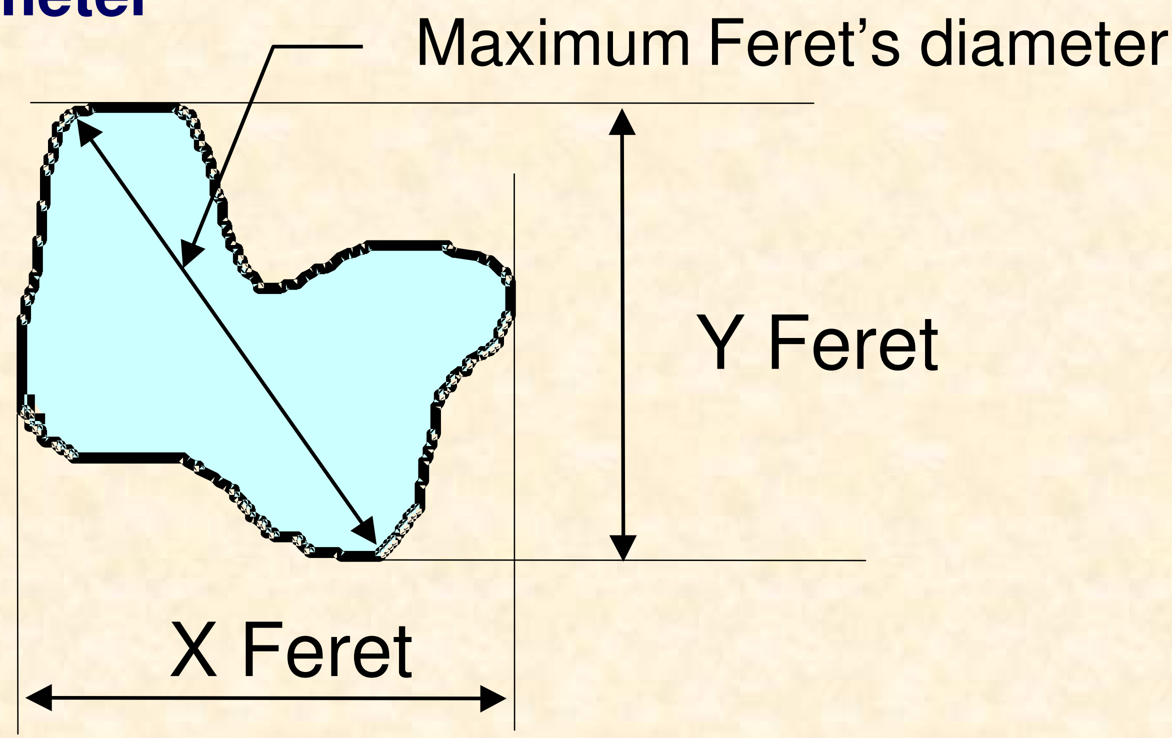
$$D = 2\max\left(\sqrt{(i_k - SC_i)^2 + (j_k - SC_j)^2}\right) \qquad k = 1,\ 2,\ldots,P$$

P – *number of object pixels*

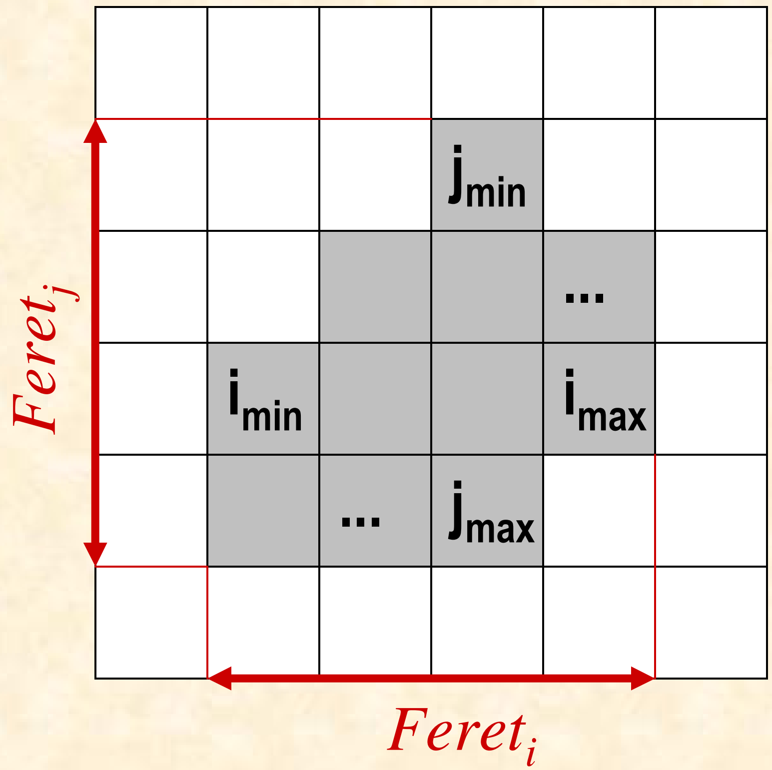
Ferets

Region orientation can be identified from its projection onto X and Y axes, these are termed X **Feret** and Y **Feret**, correspondingly.

Maximum Feret's diameter is the line between two object points that are farthest apart.



Finding Ferets



$$Feret_i = \max(i_k - i_l),\ \ k,l = 1,\ 2,\ldots,P$$

$$Feret_j = \max(j_k - j_l),\ \ k,l = 1,\ 2,\ldots,P$$

$$P - number\ of\ object\ pixels$$

M. Strzelecki, P. Strumiłło

Finding Ferets - algorithm

```
{ FeretX, FeretY - calculated Feret diameters;
Contour_tab – table containing sequence of contour pixels' neighbourhoods
i,j – coordinates of the starting pixel of the contour}
....
FerXMi := N; FerXma := 0; FerYMi := N; FerYMa := 0;
for n:=1 to Edge do begin
   if FerXMi > i then FerXmi := i; if FerYMi > j then FerYmi := j;
   if FerXMa < i then FerXma := i; if FerYMa < j then FerYma := j;
   Next_ij(Contour_tab[n],i,j);
end;
FeretX := FerXMa - FerXMi;    FeretY := FerYMa - FerYMi;
....
```