Sistemas de Visão e Percepção Industrial

4-Processamento a Médio Nível

Morfologia binária

Universidade de Aveiro
Vitor Santos, 2022-2023

Sumário

Referências

- W. Burger, Chap. 10
- E. R. Davies, Chap. 8
- R. Gonzalez, Chap. 9
- M. Sonka, Chap. 11

Operações morfológicas básicas

Visão por computador – O médio nível

- Operadores morfológicos (binários)
  - Operadores para lidar com as partes da imagem divididas em "objetos" e "fundo" (background)
  - Um objeto é um conjunto de pixels que partilha uma mesma propriedade – como o estarem ligados entre si (conexão 4- ou 8-, indicando estar ligado a 4 ou a 8 vizinhos)
- O elemento estruturante de um operador morfológico
  - Vizinhança de efetividade do operador: (o pixel central (a vermelho) em geral também é parte ativa no elemento estruturante, mas pode não ser, como se verá adiante)
    - N4: 
    N8: 
    - Poderão ser definidos outros elementos estruturantes, ex.:

Operadores morfológicos – Erosão e dilatação

- Erosão (erode)
  - Tomar cada pixel de objeto (valor 1) que esteja N-ligado (N4, N8, ...) a um pixel de fundo (valor 0) e colocar esse pixel do objeto a 0.
  - Em geral, reduz as dimensões do objeto
- Dilatação (dilate)
  - Tomar cada pixel de fundo (valor 0) e colocá-lo como objeto (valor 1) se estiver N-ligado (N4, N8, ...) a um pixel de objeto (com valor 1).
  - Em geral, aumenta as dimensões do objeto

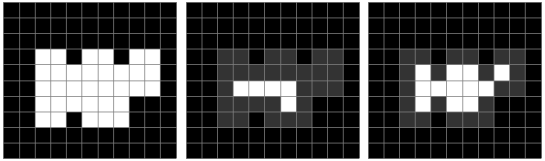


Imagem original e erosões com N8 e N4, respetivamente

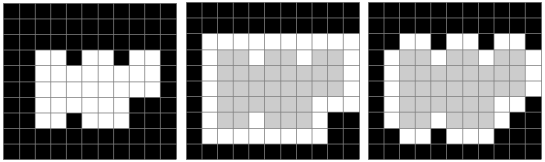


Imagem original e dilatações com N8 e N4, respetivamente

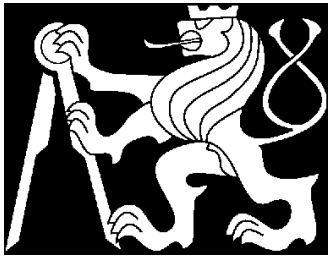
NB. Nas figuras foi mantido a sombreado a representação da imagem original apenas para se perceber melhor. Efetivamente, esses pixels a sombreado (cinzento) são pretos nas imagens de erosões e brancos nas imagens de dilatações!

Operadores Morfológicos – Fecho e abertura

- Fecho (closing)
  - Operação de dilatação seguida de erosão
  - Um resultado comum interessante é o de fundir objetos que inicialmente só estariam ligados por um único pixel.
  - "Suaviza" o objeto pelo exterior do seu contorno
- Abertura (opening)
  - Operação de erosão seguida de dilatação
  - Um resultado comum interessante é o de separar objetos que inicialmente só estariam ligados por poucos pixels, nomeadamente, linhas de um só pixel de "largura".
  - "Suaviza" o objeto pelo interior do seu contorno

Exemplos em Matlab – dilatação/erosão

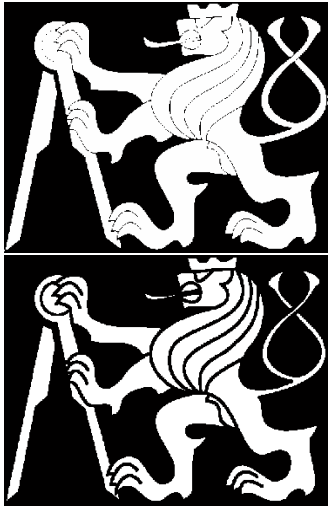
- A função `bwmorph( )` e operadores associados



- `bwmorph()` usa N8 como elemento estruturante.

Dilatação

Erosão



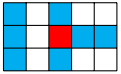
vs_morph_simple.m

Elementos estruturantes

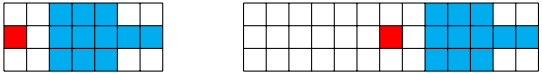
Outros elementos estruturantes

- O "centro" do elemento estruturante
  - O elemento estruturante pode ter qualquer geometria
  - A sua ação pode ficar "centrada" em torno de qualquer dos seus pontos (o centro) Exemplos:

    - NB: Em Matlab o centro só pode ser no "centro" geométrico da matriz que o representa!
- O elemento estruturante não tem de ser contíguo
  - A zona de ação é determinada pela parte ativa (não nula) da matriz que o representa.



  - Pode haver casos de centro "fora" da região delimitada (em Matlab só é possível estendendo o elemento estruturante com zeros! [Dir.]).

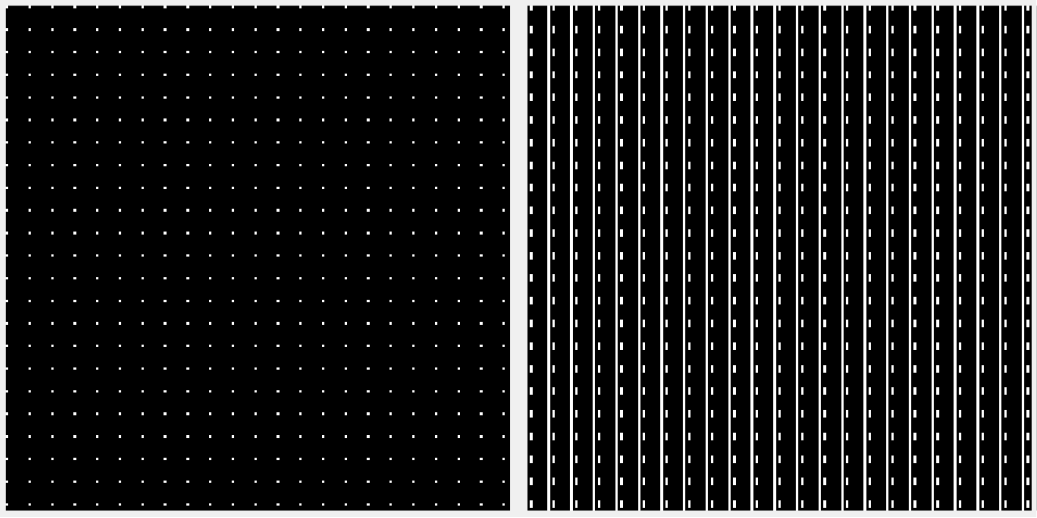


Exemplos com outros elementos estruturantes

- Para elementos estruturantes mais complexos no Matlab devem-se usar-se funções especiais em vez de bwmorph()
  - `imdilate`, `imerode`, `imclose`, `imopen`.
- Exemplo de erode com dois elementos estruturantes diferentes:
  - Esq.: `se=[1 1 1 1 1 1 1]` Dir.: `se=[1 1 1;1 1 1;1 1 1]`

Ainda outro exemplo

- Como se obtém a imagem da direita a partir da esquerda por dilatação?



vs_dil.m

Operações morfológicas mais complexas

Operação de Hit-and-miss

- Operação morfológica indicada para deteção de determinados padrões e que usa dois elementos estruturantes:
  - O **Hit** (para indicar a conformidade positiva – quais as partes do padrão que devem ser 1)
  - O **Miss** (para indicar a conformidade negativa – quais as partes do padrão que devem ser 0)
    - Em geral: `Miss = ~Hit`, mas há casos de partes *"don't care"* no elemento estruturante. Isso significa que esses valores do `Miss` que não obedecem a `~Hit` e serão também 0.

Hit para cruzes

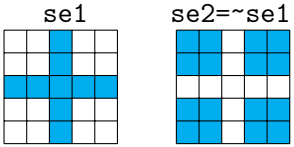
Miss para cruzes (estrito)

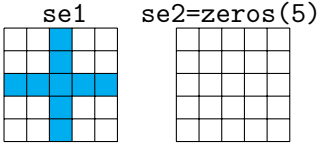
Miss para cruzes (permissivo)

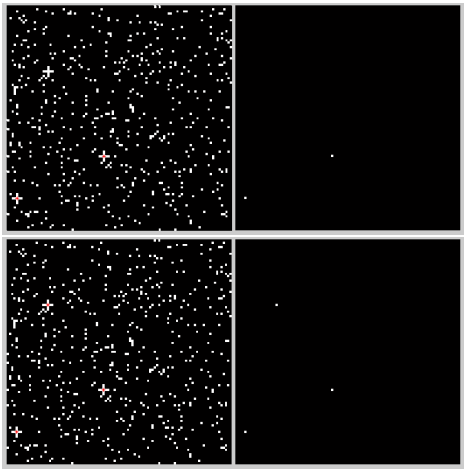
Partes *"don't care"* para o padrão de Miss permissivo. Ou seja, os pixels correspondentes serão desprezados na operação de deteção pelo elemento `Miss`

Exemplos de Hit and Miss

- Com elementos estruturantes permissivos e não permissivos



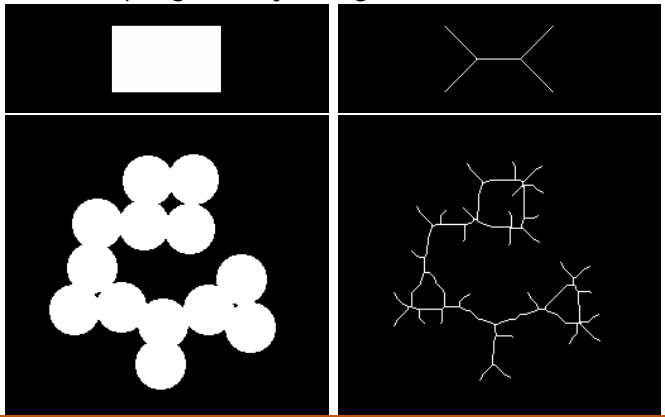




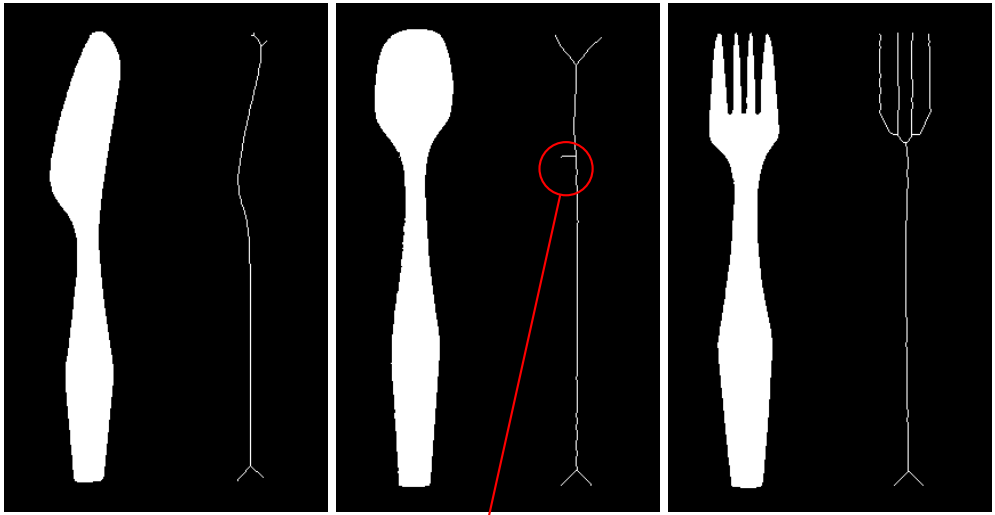
vs_hitmiss.m

Operadores Morfológicos - Esqueletização

- Operação de determinação do esqueleto
- Definição de esqueleto:
  - Objeto filiforme (1 pixel de largura) ...
  - .. que passa pelo "meio" do objeto
  - ... e que preserva a topologia do objeto original



Exemplos de Esqueletização



Porquê?

vs_skel_spoon.m

Operadores Morfológicos – "Thinning"

- A variante "thinning" da esqueletização é baseada na erosão condicionada (menos exigente computacionalmente)
- Um pixel não é erodido nas seguintes condições:
  - Se for isolado - condição C1:

  - Se removê-lo afetar a conectividade – condição C2:

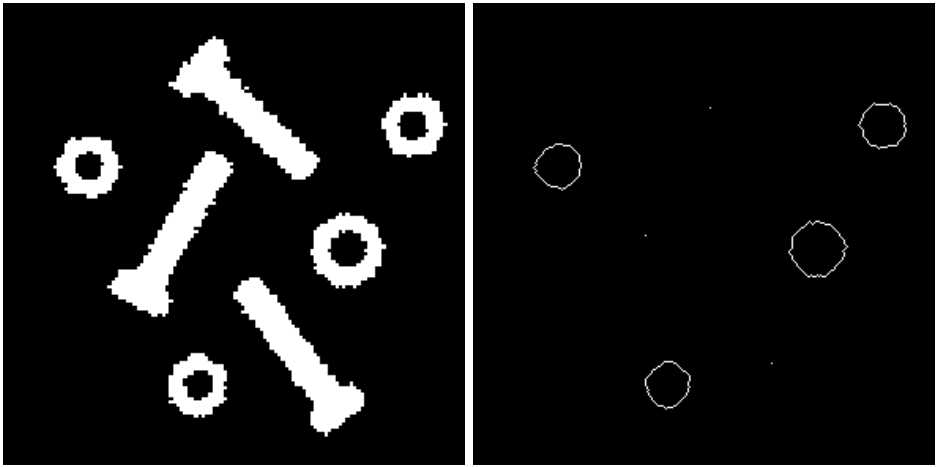
  - Se removê-lo afetar o comprimento da linha – condição C3:

Variantes de Esqueletização

- No algoritmo de "thinning" há diversas combinações das condições C1, C2 e C3.
  - Só condição C1
    - Esqueleto reduzido a um pixel
  - Só condição C2
    - Esqueleto reduzido a linhas fechadas se houver objetos com buracos. Objetos sem buracos desaparecem.
  - C1 + C2
    - O esqueleto é constituído por pixels isolados para objetos sem "buracos" e por contornos fechados para objetos com "buracos".
  - C1 + C2 + C3
    - Ter-se-á o esqueleto completo tradicional

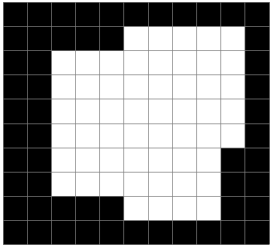
Exemplo da operação de "shrink" em Matlab

- Equivale ao "thinning" com C1 e C2
  - `bw2=bwmorph(bw1, 'shrink', inf)`

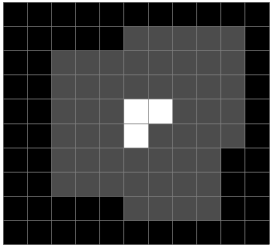


Operadores Morfológicos – Reconstrução

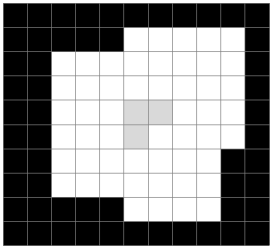
- Reconstrução ou propagação – `imreconstruct()` em Matlab
  - Operação de dilatação sucessiva de um objeto "semente" (por exemplo, um esqueleto) até aos limites dados por uma imagem "máscara" usando um dado elemento estruturante.



Máscara (*mask*)



Semente (*marker* ou *seed*)

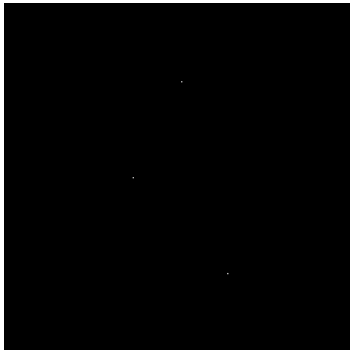


Reconstrução

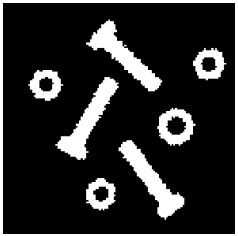
- Notas
  - Nalguma literatura, a "máscara" é um contorno fechado e não uma região cheia como se exige no caso do Matlab.
  - A reconstrução só ocorre no objeto dentro do qual se encontra a semente!
  - Nas imagens, os pixels a cinzento são ilustrativos (na Semente são pretos, e na Reconstrução são brancos)

imreconstruct() em Matlab

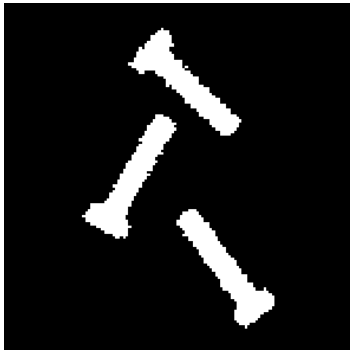
```
porcas= imread('porcas.png');
porcas=imbinarize(porcas);
porcas_s = bwmorph(porcas,'shrink', inf);
ppi = filter2([1 1 1; 1 -8 1; 1 1 1], porcas_s);
marker= (abs(ppi)==8);
ppr=imreconstruct(marker, porcas);
```



Marker (semente)



Mask (máscara)



Resultado do imreconstruct

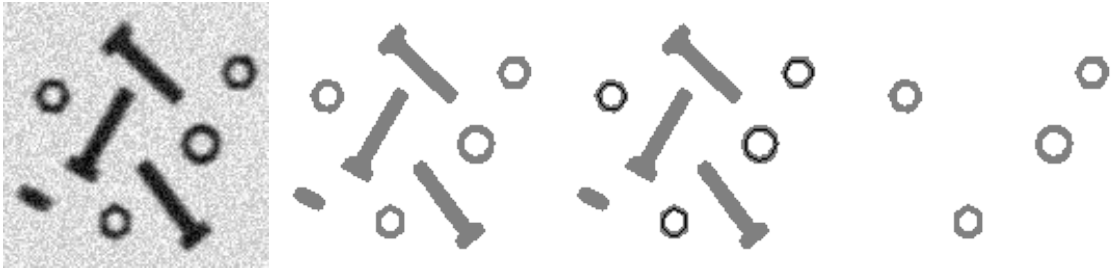
Operações morfológicas de alto nível

Operações de morfologia binária

- Alguns exemplos comuns de operações morfológicas
  - Isolar objetos com buracos
  - Preencher os buracos em objetos
  - Remover objetos nos bordos da imagem
  - Cálculo do Exo-esqueleto
  - Separar objetos que se tocam
  - Etc.

Isolar objetos com buracos

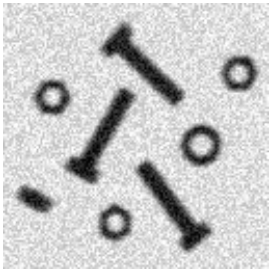
1. Binarizar a imagem de forma a ter objetos brancos.
2. Obter o "esqueleto" sem preservar os pixels terminais (shrink)
3. Eliminar pontos isolados do esqueleto (ficam os anéis)
4. Propagar o restante esqueleto (os anéis) até à máscara obtida em 1.

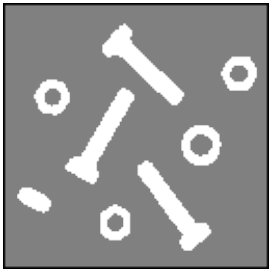


- Nas ilustrações os objetos não estão a branco para tornar as figuras mais ligeiras!

Preencher buracos em objetos

1. Binarizar a imagem de forma a ter objetos brancos.
2. Complementar a imagem (negar) para ter fundo branco.
3. Definir o bordo (caixilho) da imagem como semente com valor branco
4. Propagar o bordo (semente) até aos limites da imagem obtida em 2.
5. Tomar a imagem propagada e invertê-la





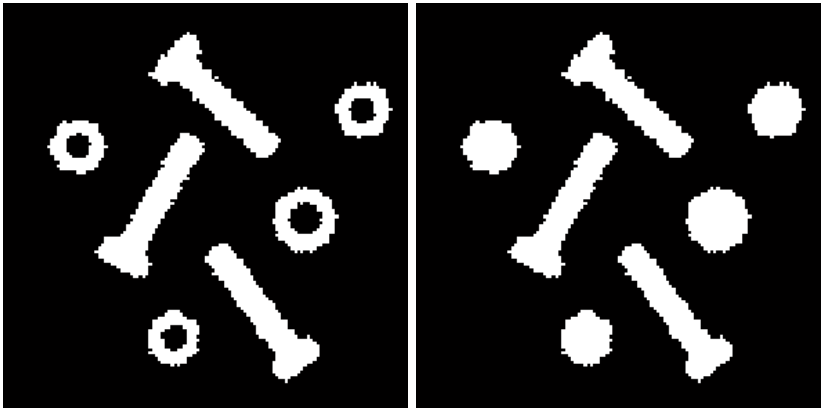

- Obs. Algumas aplicações de software têm já funções para fazer esta operação num passo único (como o caso do Matlab)
- Nas ilustrações os objetos não estão a branco para tornar as figuras mais ligeiras!

Código Matlab para remover furos

```
%Fill holes
close all
A=imread('porcas.png');
B=imbinarize(A);
%=========================
S=logical(zeros(size(B)));
S(:,1)=1;
S(:,end)=1;
S(1,:)=1;
S(end,:)=1;
B=~B;
C=imreconstruct(S, B);
C=~C;
imshow(C)
```

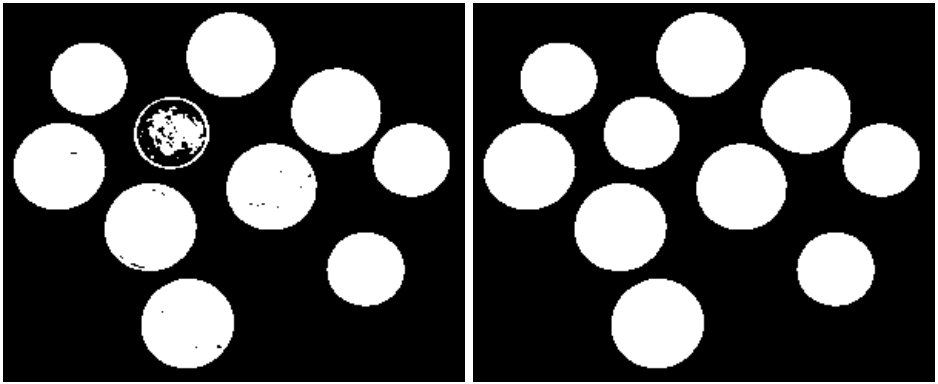
Código compacto!

```
%Fill holes
A=imread('porcas.png');
B=imbinarize(A);
C=imfill(A, 'holes');
imshow(C);
```



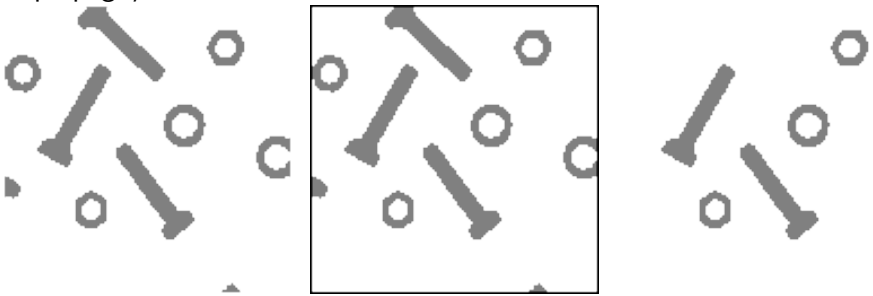
Outro exemplo com código compacto

```
BW4 = imbinarize(imread('coins.png'));
BW5 = imfill(BW4,'holes');
figure, imshow(BW4), figure, imshow(BW5)
```



Remover objetos que tocam no bordo da imagem

1. Binarizar a imagem de forma a ter objetos brancos (máscara)
2. Criar um objeto semente que é o bordo da imagem em branco
3. Propagar a semente até à máscara (preenche objetos ligados ao bordo)
4. O que não foi propagado não está no bordo – é essa a imagem final, ou seja, a imagem após a propagação!



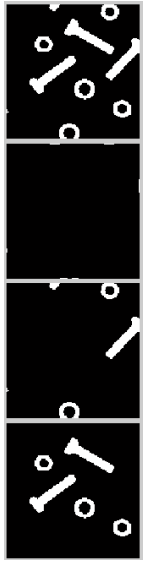
- Obs. Algumas aplicações de software têm já funções para fazer esta operação num passo único (como o caso do Matlab)
- Nas ilustrações os objetos não estão a branco para tornar as figuras mais ligeiras!

Remoção de objetos no bordo da imagem

```
A=imread('porcas01.png');
B=imbinarize(A, 0.5);
subplot(1,4,1), imshow(B)

S=false(size(B));
S(:,1)=1; S(:,end)=1;
S(1,:)=1; S(end,:)=1;
S=(S&B);
subplot(1,4,2),imshow(S)

M=imreconstruct(S, B);
subplot(1,4,3); imshow(M)
N=B & ~M;
subplot(1,4,4); imshow(N)
```

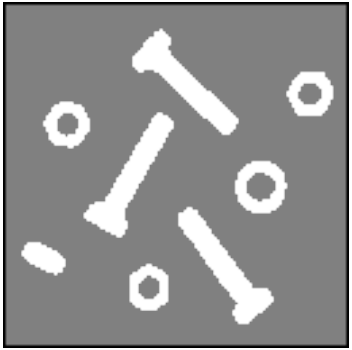


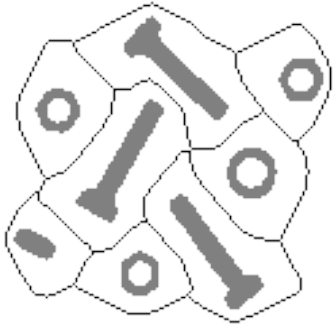
- Ou simplesmente apenas...

```
N=imclearborder(B);
```

Obtenção do Exo-esqueleto

- Definição de exo-esqueleto: esqueleto do "fundo" que contém os objetos, "criando regiões"
- Binarização de forma a ter os objetos a branco.
- Complementar a imagem (negação)
- Cálculo do esqueleto pela metodologia que elimina as linhas terminais (condições C1 e C2 na definição da operação de "thinning")





Conectividade de regiões

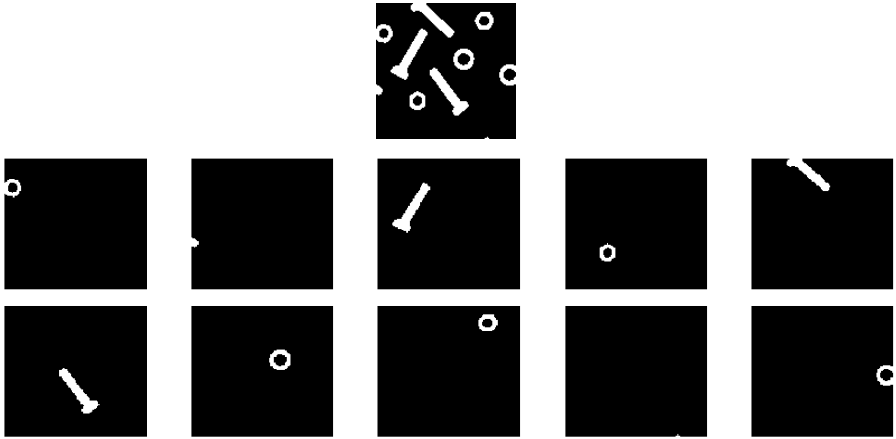
- Uma imagem binária pode ser catalogada em regiões baseada na conectividade (4- ou 8-) dos pixels.

0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	1	0	0	0	0
0	1	1	1	0	1	1	1	0	0
1	1	1	1	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0
0	0	1	1	1	0	0	0	1	0
0	0	0	1	0	0	0	1	1	1
0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	2	0	0	0	0
0	1	1	1	0	2	2	2	0	0
1	1	1	1	0	0	2	0	0	0
0	1	0	0	0	0	0	0	0	0
0	0	0	3	3	3	0	0	0	0
0	0	3	3	3	0	0	0	4	0
0	0	0	3	0	0	0	4	4	4
0	0	0	3	0	0	0	0	4	0
0	0	0	0	0	0	0	0	0	0

- Em Matlab isso faz-se de forma automática com a função `bwlabel()`.

Separação de objetos – Exemplo em Matlab



- Como fazer...? Numa forma compacta de código ☺...?

```
p=imread('porcas01.png'); [L,num]=bwlabel(p);
for i=1:num; subplot(2,num/2,i), imshow(L==i); end;
```

Propriedades dos objetos

- Tendo os objetos separados, podem-se então estudar as propriedades individuais
  - Area
  - Perímetro
  - Rectângulo envolvente
  - Eixos de elipse envolvente
  - Extremos (pixels extremos)
  - Factor de forma
  - Etc.
- A abordar mais tarde...

Funções em MatLab

- BW2 = BWMORPH(BW1,OPERATION,N) applies the operation N times. N can be Inf, in which case the operation is repeated until the image no longer changes.
- OPERATION is a string that can have one of these values:
  - 'bothat' Subtract the input image from its closing
  - 'branchpoints' Find branch points of skeleton
  - 'bridge' Bridge previously unconnected pixels
  - 'clean' Remove isolated pixels (1's surrounded by 0's)
  - 'close' Perform binary closure (dilation followed by erosion)
  - 'diag' Diagonal fill to eliminate 8-connectivity of background
  - 'dilate' Perform dilation using the structuring element ones(3)
  - 'endpoints' Find end points of skeleton
  - 'erode' Perform erosion using the structuring element ones(3)
  - 'fill' Fill isolated interior pixels (0's surrounded by 1's)
  - 'hbreak' Remove H-connected pixels
  - 'majority' Set a pixel to 1 if five or more pixels in its 3-by-3 neighborhood are 1's
  - 'open' Perform binary opening (erosion followed by dilation)
  - 'remove' Set a pixel to 0 if its 4-connected neighbors are all 1's, thus leaving only boundary pixels
  - 'shrink' With N = Inf, shrink objects to points; shrink objects with holes to connected rings
  - 'skel' With N = Inf, remove pixels on the boundaries of objects without allowing objects to break apart
  - 'spur' Remove end points of lines without removing small objects completely
  - 'thicken' With N = Inf, thicken objects by adding pixels to the exterior of objects without connected previously unconnected objects
  - 'thin' With N = Inf, remove pixels so that an object without holes shrinks to a minimally connected stroke, and an object with holes shrinks to a ring halfway between the hole and outer boundary
  - 'tophat' Subtract the opening from the input image