Procesarea Imaginilor Digitale

Curs - Morfologie Matematică

Universitatea ”Transilvania” din Brașov

Cuprins

Morfologia matematică - Introducere

- oferă instrumente de preprocesare a imaginii.

Morfologia matematică - Introducere

- oferă instrumente de preprocesare a imaginii.
- permite extragerea de componente sau schelete.

Morfologia matematică - Introducere

- oferă instrumente de preprocesare a imaginii.
- permite extragerea de componente sau schelete.
- are la bază teoria mulțimilor și operațiilor logice.

Morfologia matematică - Introducere

- oferă instrumente de preprocesare a imaginii.
- permite extragerea de componente sau schelete.
- are la bază teoria mulțimilor și operațiilor logice.
- realizează tranzița către operații de tip *mid-level*.

Morfologia matematică - Introducere

- oferă instrumente de preprocesare a imaginii.
- permite extragerea de componente sau schelete.
- are la bază teoria mulțimilor și operațiilor logice.
- realizează tranzița către operații de tip *mid-level*.
- consideră mulțimi din $\mathbb{Z}^2$ = mulțimi de coordonate.

Notații din teoria mulțimilor

- $C_A = \{w |\ w \notin A\}$ - complementara mulțimii A
- $A - B = \{w | w \in A \text{ și } w \notin B\} = A \cap C_B$
- $\hat{B} = \{w | w = -b, b \in B\}$ - simetricul lui B față de origine
- $(A)_z = \{w | w = a + z, a \in A\}$ - translația lui A cu $z = (z_1, z_2)$

Operații logice cu imagini binare

p	q	NOT p	p AND q	p OR q	p XOR q
0	0				
0	1				
1	0				
1	1				

Table: În imagini binare considerăm 1 = pixel alb, 0 = pixel negru.

Operații logice cu imagini binare

p	q	NOT p	p AND q	p OR q	p XOR q
0	0	1			
0	1	1			
1	0	0			
1	1	0			

Table: În imagini binare considerăm 1 = pixel alb, 0 = pixel negru.

Operații logice cu imagini binare

p	q	NOT p	p AND q	p OR q	p XOR q
0	0	1	0		
0	1	1	0		
1	0	0	0		
1	1	0	1		

Table: În imagini binare considerăm $1 =$ pixel alb, $0 =$ pixel negru.

Operații logice cu imagini binare

p	q	NOT p	p AND q	p OR q	p XOR q
0	0	1	0	0	
0	1	1	0	1	
1	0	0	0	1	
1	1	0	1	1	

Table: În imagini binare considerăm 1 = pixel alb, 0 = pixel negru.

Operații logice cu imagini binare

p	q	NOT p	p AND q	p OR q	p XOR q
0	0	1	0	0	0
0	1	1	0	1	1
1	0	0	0	1	1
1	1	0	1	1	0

Table: În imagini binare considerăm $1 =$ pixel alb, $0 =$ pixel negru.

Operații logice cu imagini binare

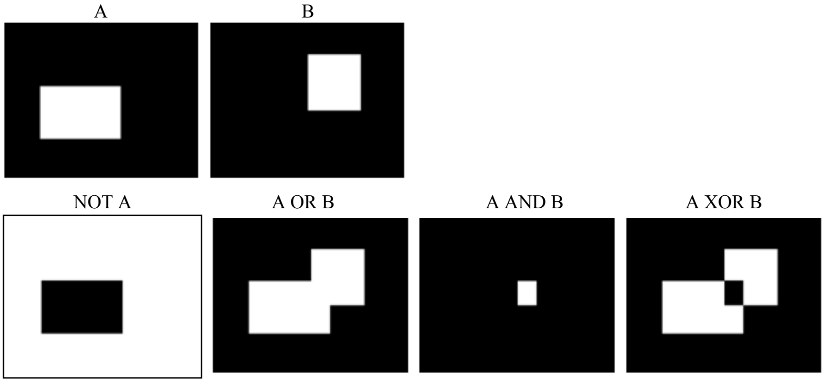


Figure: Imaginile binare conținând mulțimile A și B împreună cu imaginile rezultate prin aplicarea operațiilor logice NOT, AND, OR și XOR asupra acestor imagini.

Dilatarea

- **Definiție matematică**:Fie A și B în $\mathbb{Z}^2$ *dilatarea* lui A cu B este:

$$A \oplus B = \{z | (\hat{B})_z \cap A \neq \emptyset\}$$

- **Element structurant**: B - de obicei $\hat{B} = B$

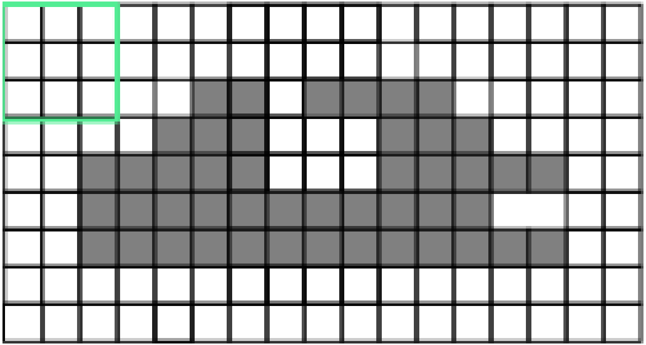
Dilatarea

Practic:

- Se consideră o mască B centrată în $(0,0)$
- Dacă $\exists (s,t) \in B_{xy}$ cu $f(s,t) = 255 \Rightarrow g(x,y) = 255$
- Altfel $g(x,y) = 0$
- B_{xy} = vecinătatea lui (x,y) definită de masca B

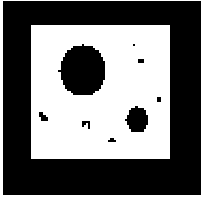
Observație: echivalentă cu operația SAU asupra pixelilor din mască.

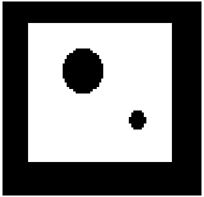
Dilatarea - Exemplu



Dilatarea - Exemplu

1	1	1
1	1	1
1	1	1





$$g(x,y) = \begin{cases} 255, & \text{dacă } f(x,y) = 255 \\ 255, & \text{dacă } \exists (x_1, y_1) \in B_{xy} \text{ a. î. } f(x_1, y_1) = 255 \\ 0 & \text{altfel} \end{cases}$$

$$B_{xy} = \{(x-1, y-1), (x, y-1), (x+1, y-1), (x-1, y), (x, y),$$
$$(x+1, y), (x-1, y+1), (x, y+1), (x+1, y+1)\}$$

Dilatarea - Diferite elemente structurante

1	1	1
1	1	1
1	1	1

a)

0	1	0
1	1	1
0	1	0

b)

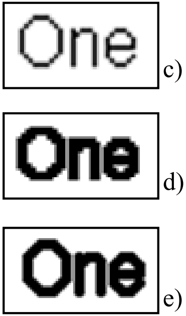


Figure: a) Element structurant dreptunghiular; b) Element structurant circular; c) Imagine originală; d) Imagine dilatată cu a); e) Imagine dilatată cu b).

Dilatarea - Proprietăți

- ...

Dilatarea - Proprietăți

- ...
- Expandează obiectele din imagine.

Dilatarea - Proprietăți

- ...
- Expandează obiectele din imagine.
- Umple mici goluri.

Dilatarea - Proprietăți

- ...
- Expandează obiectele din imagine.
- Umple mici goluri.
- Reduce fragmentarea contururilor.

Erodarea

- **Definiție matematică**:Fie A și B în $\mathbb{Z}^2$ *Erodarea* lui A cu B este:

$$A \ominus B = \{z | (\hat{B})_z \subseteq A\}$$

- **Element structurant**: B - de obicei $\hat{B} = B$

Erodarea

Practic:

- Se consideră o mască B centrată în $(0,0)$
- Dacă $\exists(s,t) \in B_{xy}$ cu $f(s,t) = 0 \Rightarrow g(x,y) = 0$
- Altfel $g(x,y) = 255$
- B_{xy} = vecinătatea lui (x,y) definită de masca B

Observație: echivalentă cu operația ȘI asupra pixelilor din mască.

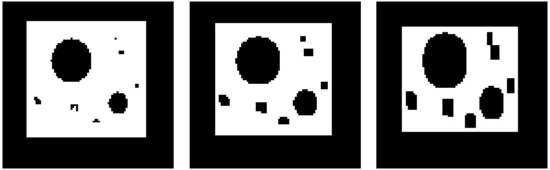
Erodrea - Exemplu

1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1

a)

1	1	1
1	1	1
1	1	1

b)



c) d) e)

Erodarea - Proprietăți

- ...

Erodarea - Proprietăți

- ...
- Reduce obiectele din imagine.

Erodarea - Proprietăți

- ...
- Reduce obiectele din imagine.
- Elimină detalii nerelevante mai mici decât elementul structurant (zgomot).

Erodarea - Proprietăți

- ...
- Reduce obiectele din imagine.
- Elimină detalii nerelevante mai mici decât elementul structurant (zgomot).
- Mărește fragmentarea contururilor.

Problemă

Doresc:

- Eliminare goluri, defragmentare contururi.

  sau

- Eliminare zgomot, detalii nerelevante.

NU doresc:
Modificarea dimensiunii obiectelor!

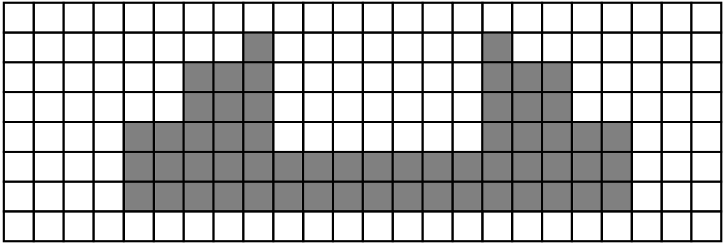
Ce fac?

Opening

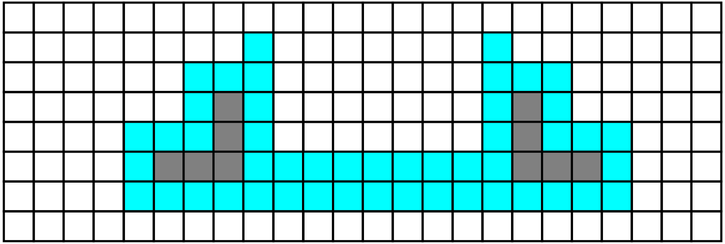
Definiție:

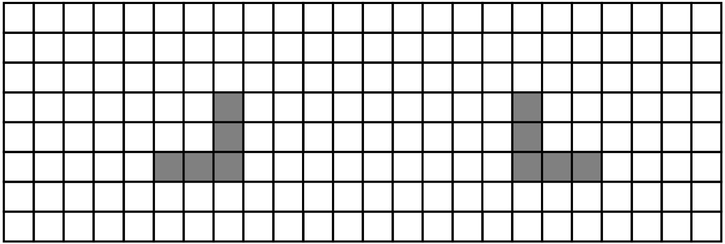
$$A \circ B = (A \ominus B) \oplus B$$

Opening - Exemplu

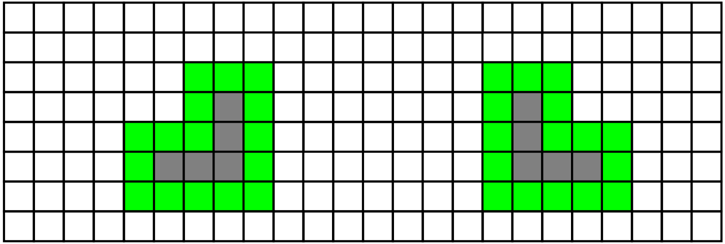


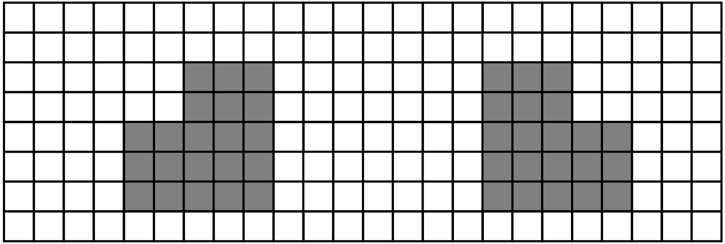
Opening - Exemplu



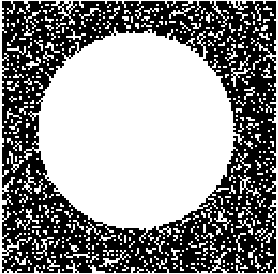


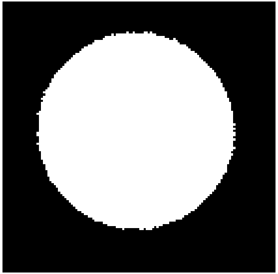
Opening - Exemplu





Opening





Proprietăți:

- Netezește contururile obiectelor.
- Separă obictele legate prin "punți" subțiri.
- Elimină mici protuberanțe.

Opening

(1) $A \circ B \subseteq A$

(2) Dacă $C \subset D$, atunci $C \circ B \subset D \circ B$

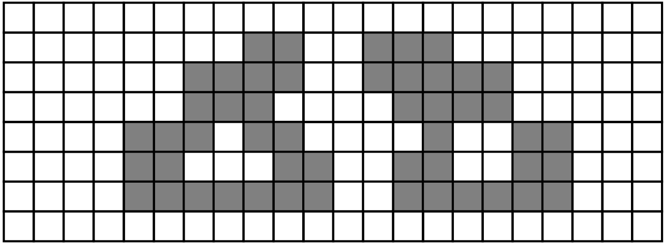
(3) $(A \circ B) \circ B = A \circ B$

Closing

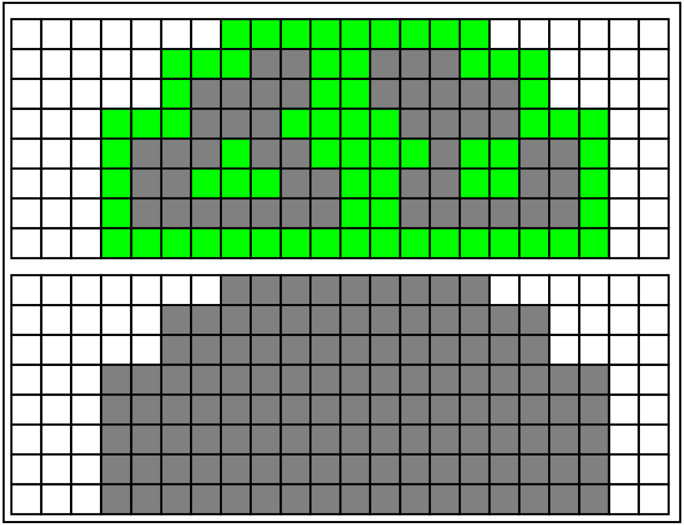
Definiție:

$$A \bullet B = (A \oplus B) \ominus B$$

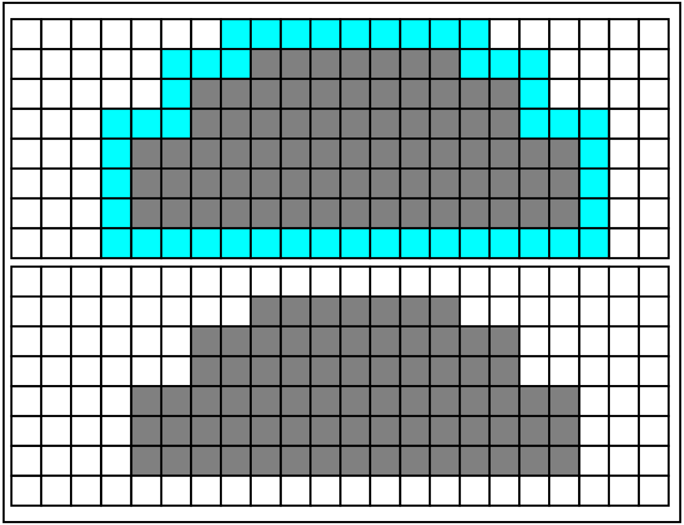
Closing - Exemplu



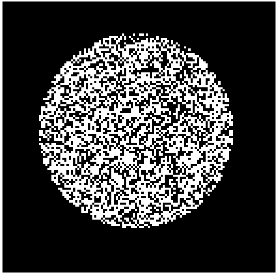
Closing - Exemplu

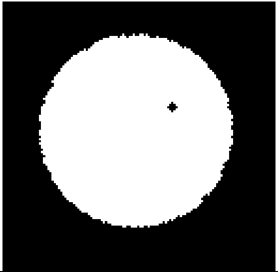


Closing - Exemplu



Closing





Proprietăți:

- Netezește contururile obiectelor.
- Umple goluri mici în obiecte și pe contururi.
- Unește obiecte aflate la distanță mică.

Closing

(1) $A \subseteq A \bullet B$

(2) Dacă $C \subset D$, atunci $C \bullet B \subset D \bullet B$

(3) $(A \bullet B) \bullet B = A \bullet B$

Aplicații

- Îmbunătățirea imaginii binare prin:

Aplicații

- Îmbunătățirea imaginii binare prin:
  - eliminarea de zgomot;

Aplicații

- Îmbunătățirea imaginii binare prin:
    - eliminarea de zgomot;
    - umplerea golurilor;

Aplicații

- Îmbunătățirea imaginii binare prin:
  - eliminarea de zgomot;
  - umplerea golurilor;
  - defragmentarea contururilor;

Aplicații

- Îmbunătățirea imaginii binare prin:
  - eliminarea de zgomot;
  - umplerea golurilor;
  - defragmentarea contururilor;
- Extragerea contururilor obiectelor.

Aplicații

- Îmbunătățirea imaginii binare prin:
  - eliminarea de zgomot;
  - umplerea golurilor;
  - defragmentarea contururilor;
- Extragerea contururilor obiectelor.
- Extragerea de componente conexe.

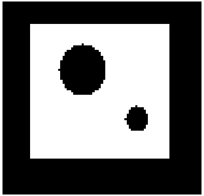
Aplicații

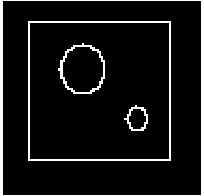
- Îmbunătățirea imaginii binare prin:
    - eliminarea de zgomot;
    - umplerea golurilor;
    - defragmentarea contururilor;
- Extragerea contururilor obiectelor.
- Extragerea de componente conexe.
- Extragerea scheletelor.

Aplicații

- Îmbunătățirea imaginii binare prin:
  - eliminarea de zgomot;
  - umplerea golurilor;
  - defragmentarea contururilor;
- Extragerea contururilor obiectelor.
- Extragerea de componente conexe.
- Extragerea scheletelor.
- etc.

Extragerea contururilor

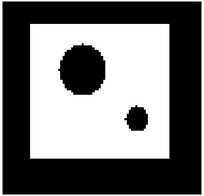


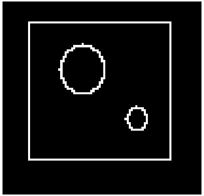


Variante

...?

Extragerea contururilor



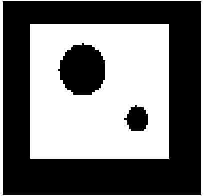


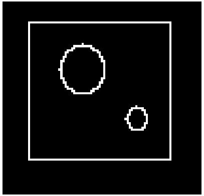
Variante

...?

1. $contur(A) = A - (A \ominus B)$

Extragerea contururilor



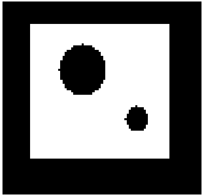


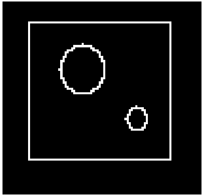
Variante

...?

1. $contur(A) = A - (A \ominus B)$
2. $contur(A) = (A \oplus B) - A$

Extragerea contururilor



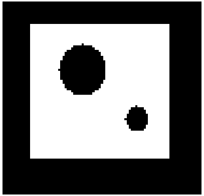


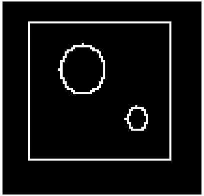
Variante

...?

1. $contur(A) = A - (A \ominus B)$
2. $contur(A) = (A \oplus B) - A$
3. $contur(A) = XOR(A, A \ominus B)$

Extragerea contururilor





Variante

...?

1. $contur(A) = A - (A \ominus B)$
2. $contur(A) = (A \oplus B) - A$
3. $contur(A) = XOR(A, A \ominus B)$
4. $contur(A) = XOR(A, A \oplus B)$

Extragerea componentelor conexe

- Se poate realiza morfologic prin dilatare.
- Se consideră:
  - mulțimea componentelor conexe A;
  - $Y \subset A$ componentă conexă;
  - $p \in Y$;
- Componenta Y se obține iterativ la convergența șirului Y_k:

$$Y_k = (Y_{k-1} \oplus B) \cap A, k = 1, 2, 3, \ldots$$

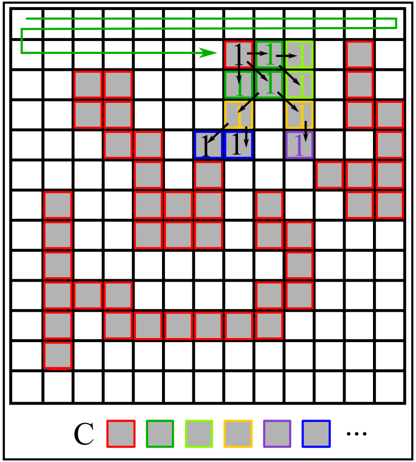
$Y_0 = p$, B element structurant potrivit.

Extragerea componentelor conexe - Practic - Alg1

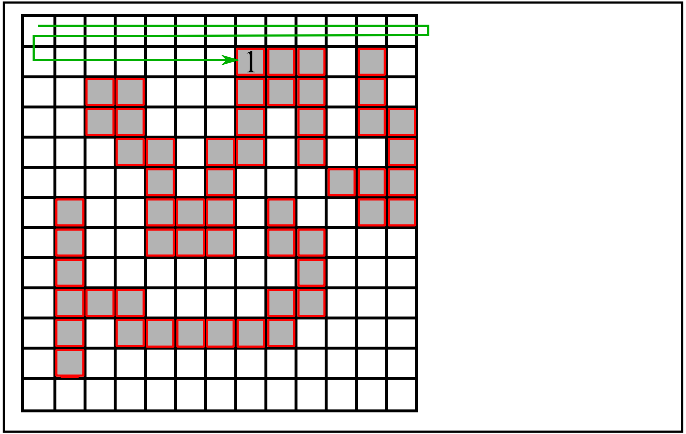
Notații: C - coadă de pixeli, p - pixel etichetat.

Algoritm: Inițial coada $C = \{p\}$

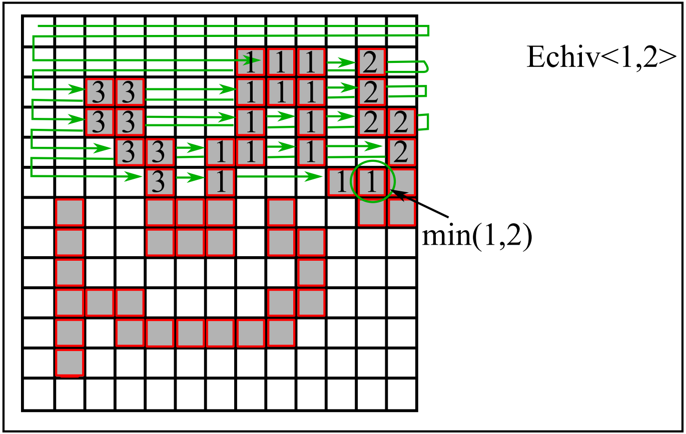
cât timp $C \neq \emptyset$
    $q \Leftarrow C$
    pentru q_i vecin al lui q, $i = \overline{1,8}$
        dacă $q_i \in A$ şi q_i nemarcat atunci
            marchează q_i cu eticheta corespunzătoare lui q
            $q_i \Rightarrow C$
        sfârşit dacă
    sfârşit pentru
sfârşit cât timp



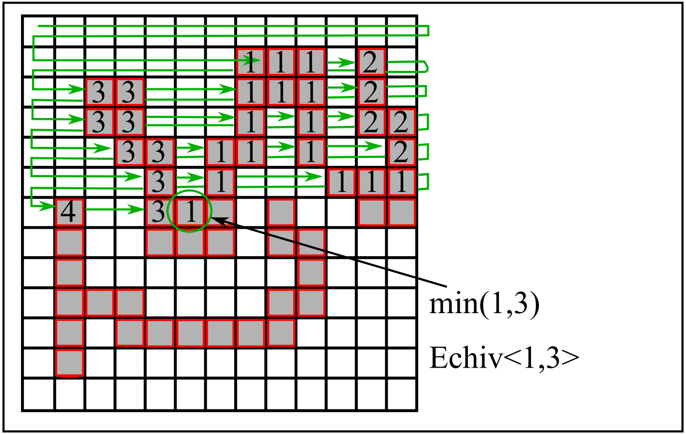
Extragerea componentelor conexe - Practic - Alg2



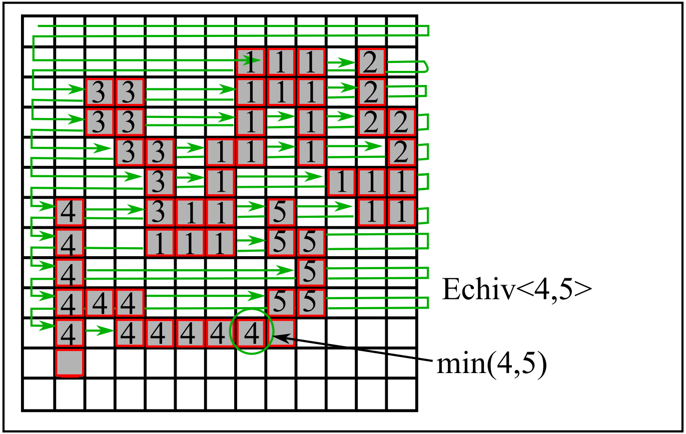
Extragerea componentelor conexe - Practic - Alg2



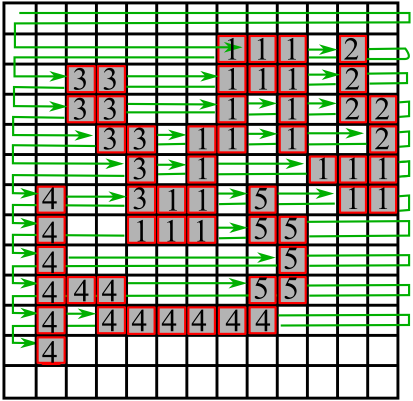
Extragerea componentelor conexe - Practic - Alg2



Extragerea componentelor conexe - Practic - Alg2



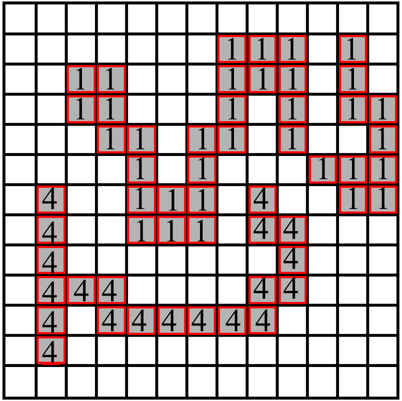
Extragerea componentelor conexe - Practic - Alg2



Echiv<1,2,3>

Echiv<4,5>

Extragerea componentelor conexe - Practic - Alg2



Echiv<1,2,3>

Echiv<4,5>

Păstrarea listelor de echivalență cu mulțimi disjuncte

Disjoint Set - structură de date $S = \{S_1, S_2, \ldots, S_3\}$

- $S_1, S_2, \ldots, S_3$ = mulțimi dinamice disjuncte
- fiecare S_i este identificat printr-un reprezentant

Operații:

- MAKE_SET(x) - crează o mulțime nouă în S care conține doar elementul x
- UNION(x,y) - reunește două mulțimi disjuncte S_x și S_y într- o nouă mulțime
- FIND_SET(x) - determină reprezentantul mulțimii care îl conține pe x

Păstrarea listelor de echivalență cu mulțimi disjuncte

Implementare

- Liste înănțuite - nu este foarte eficient (vezi Cormen)
- Pădure - mulțime de arbori

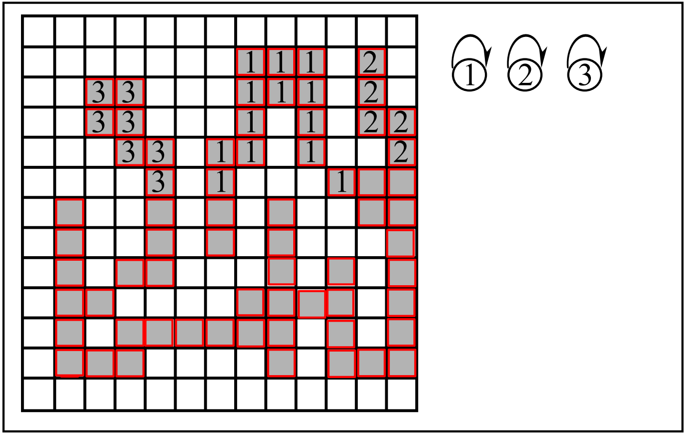
Păstrarea listelor de echivalență cu mulțimi disjuncte

Implementare Fiecare arbore este alcătuit din noduri - fiecare nod x:

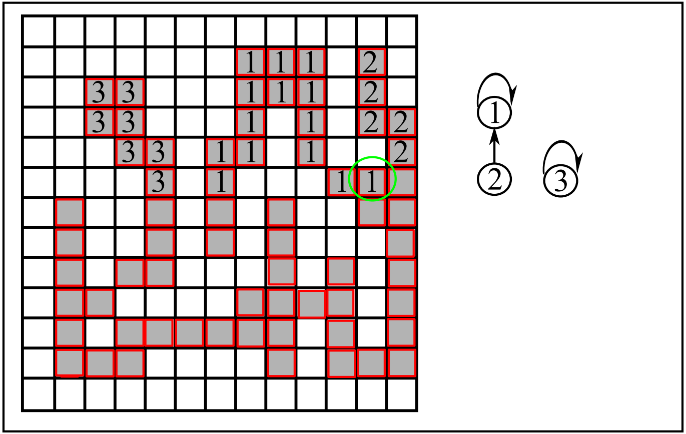
- - are un câmp eticheta
- - are un câmp de legătură către părinte.

Rădăcina - legată de ea însăși = reprezentantul mulțimii.

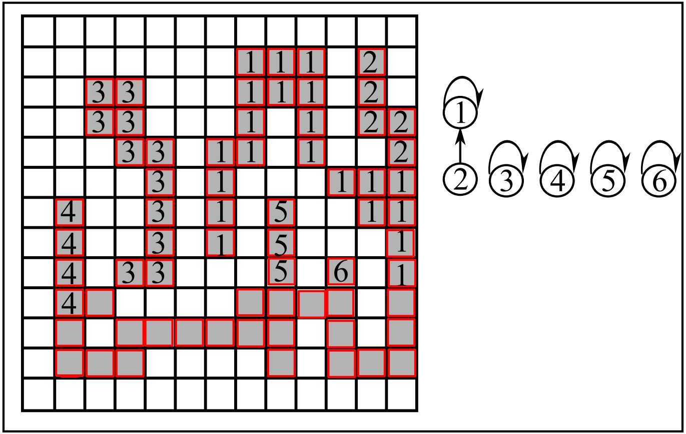
Extragerea componentelor conexe - Disjoint Set - Exemplu



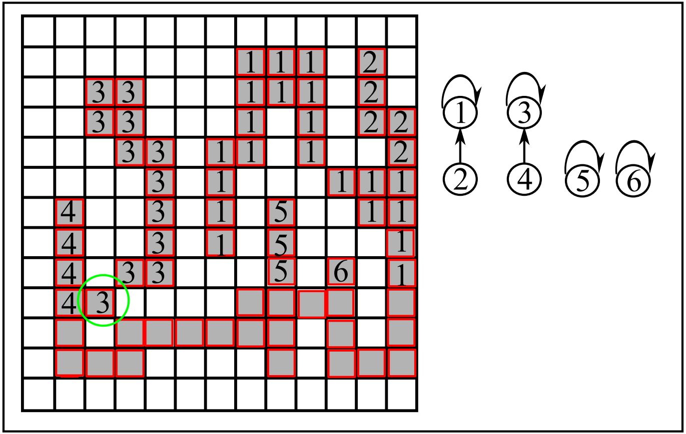
Extragerea componentelor conexe - Disjoint Set - Exemplu



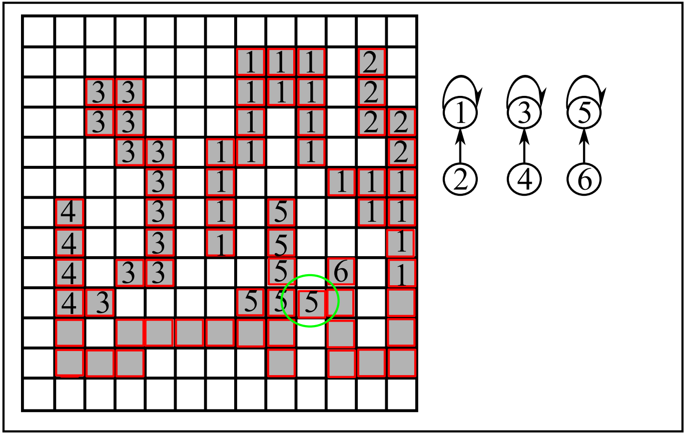
Extragerea componentelor conexe - Disjoint Set - Exemplu



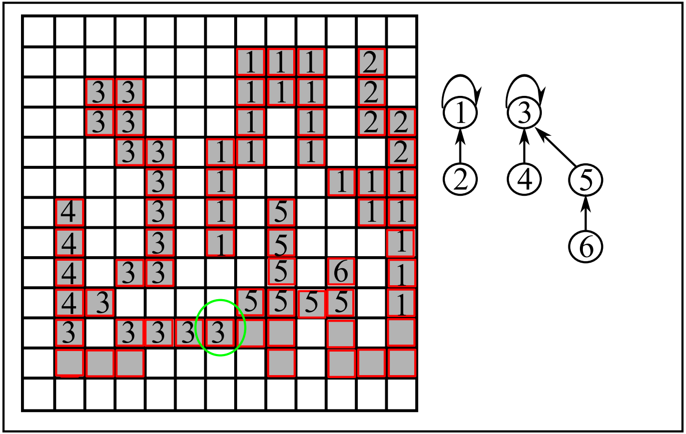
Extragerea componentelor conexe - Disjoint Set - Exemplu



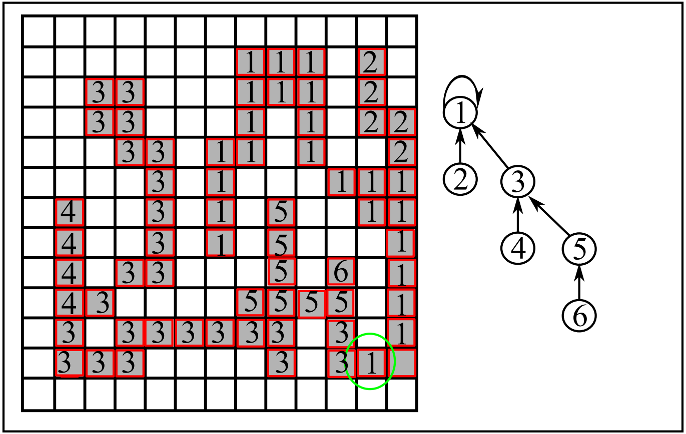
Extragerea componentelor conexe - Disjoint Set - Exemplu



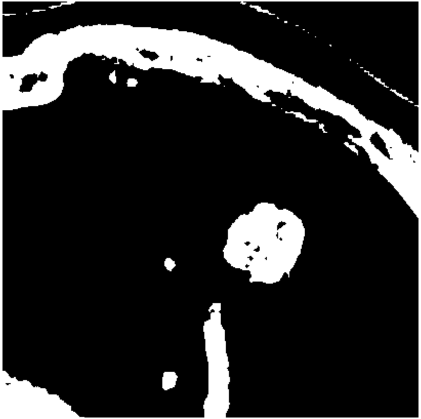
Extragerea componentelor conexe - Disjoint Set - Exemplu

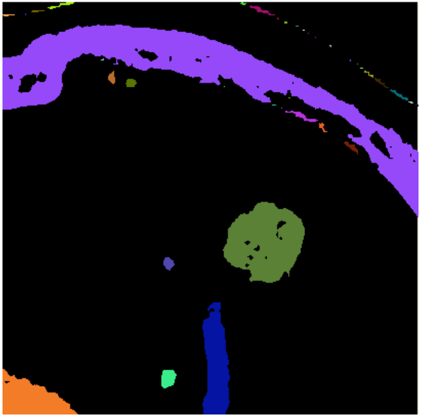


Extragerea componentelor conexe - Disjoint Set - Exemplu



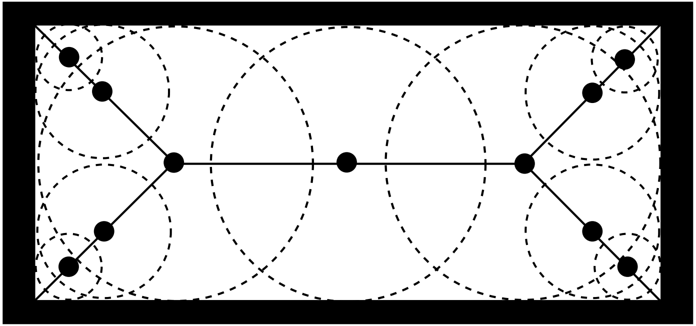
Extragerea componentelor conexe - Exemplu



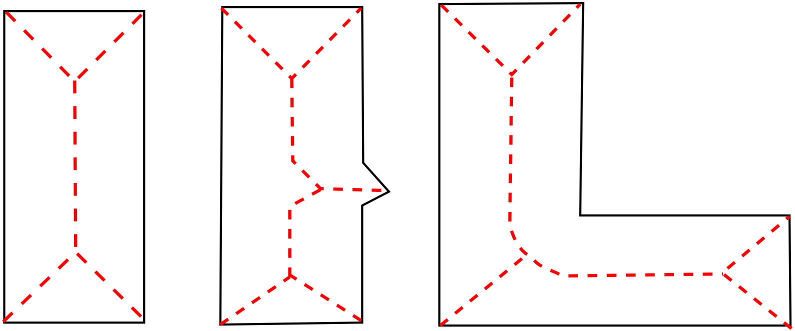


Schelete

Scheletul $S(A)$ al unei mulțimi/obiect A = mulțimea de puncte $p \in S(A)$, p centrul unui disc de rază maximă conținut în A, tangent la A în cel puțin două puncte de contur distincte.



Schelete - Exemple



Schelete - Utilizare

De obicei asupra imaginilor binare pentru:

- Recunoaștere.

Schelete - Utilizare

De obicei asupra imaginilor binare pentru:

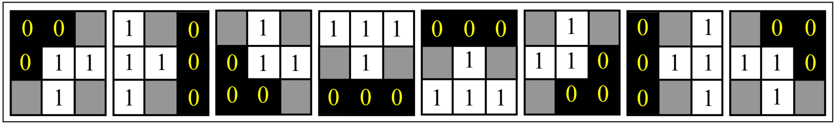
- Recunoaștere.
- Descriere.

Schelete - Utilizare

De obicei asupra imaginilor binare pentru:

- Recunoaștere.
- Descriere.
- Codificare.

Scheletare - Algoritm1 - 8 Măști

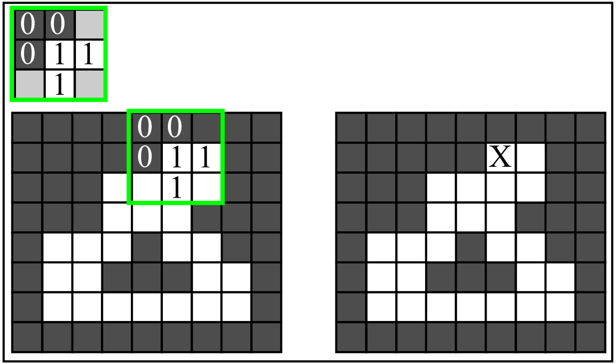


Pentru fiecare mască

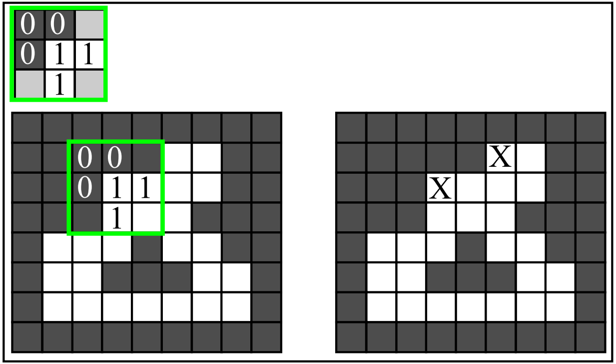
1. Plasează masca centrată pe fiecare pixel (x, y)
2. Marchează spre ștergere (x, y) dacă masca se potrivește pe vecinătatea lui (x, y)
3. După parcurgerea imaginii cu masca curentă, șterge pixelii marcați

Repetă pașii până nu mai apar modificări.

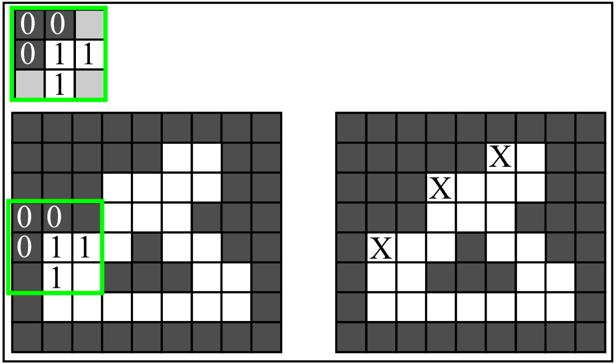
Scheletare- Exemplu



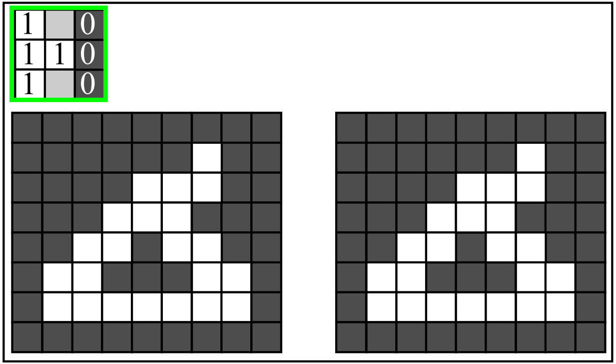
Scheletare- Exemplu



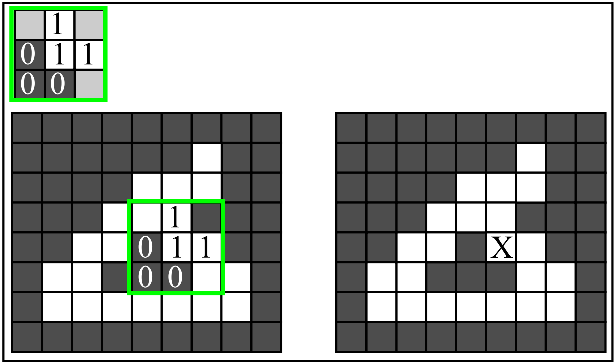
Scheletare- Exemplu



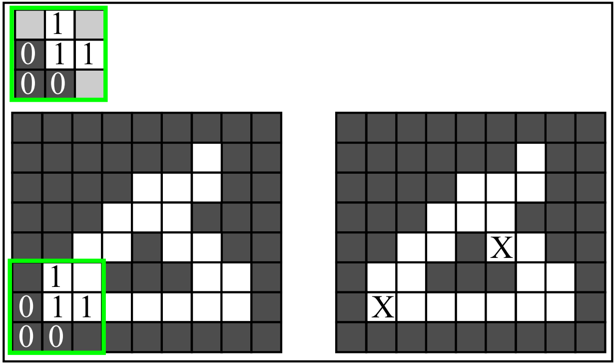
Scheletare- Exemplu



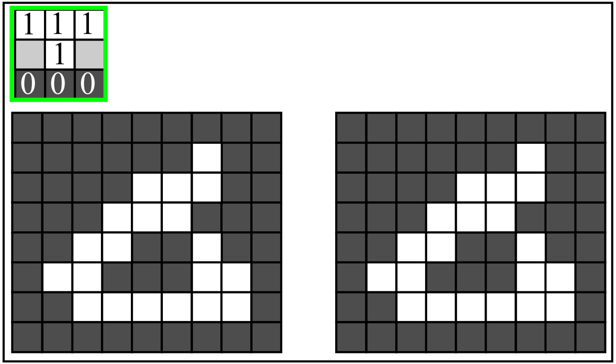
Scheletare- Exemplu



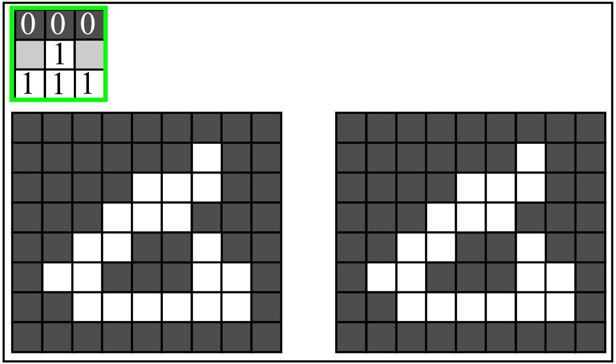
Scheletare- Exemplu



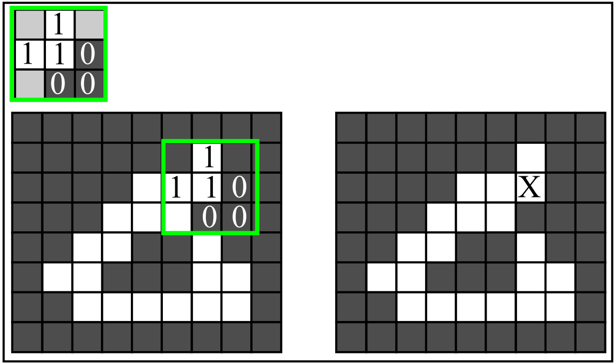
Scheletare- Exemplu



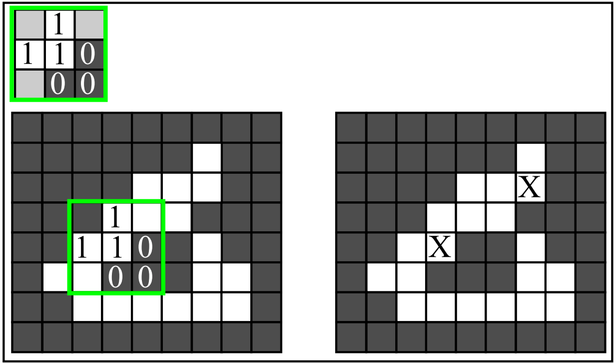
Scheletare- Exemplu



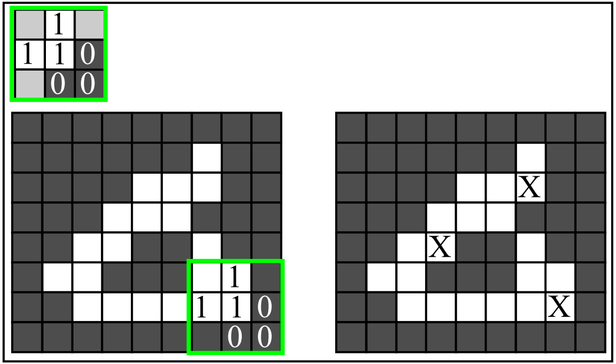
Scheletare- Exemplu



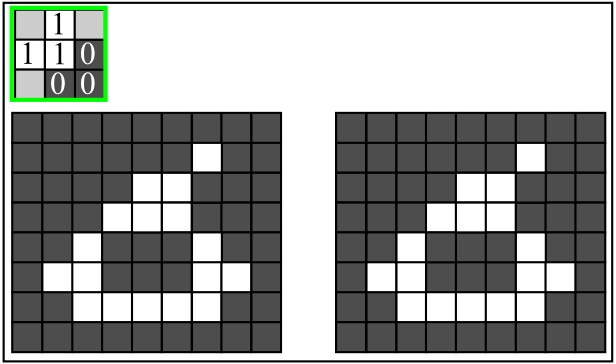
Scheletare- Exemplu



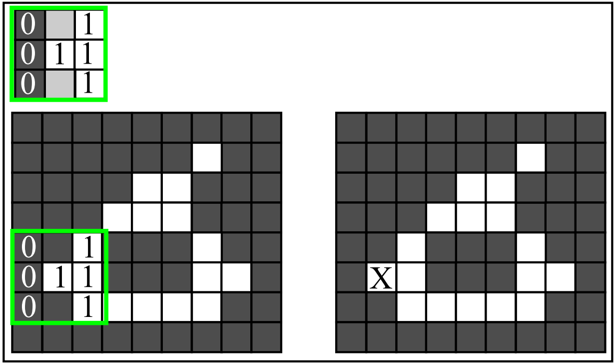
Scheletare- Exemplu



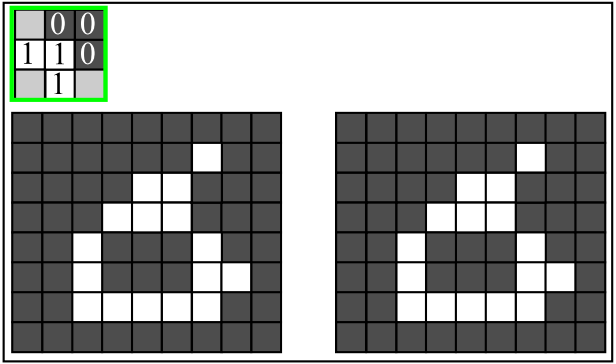
Scheletare- Exemplu



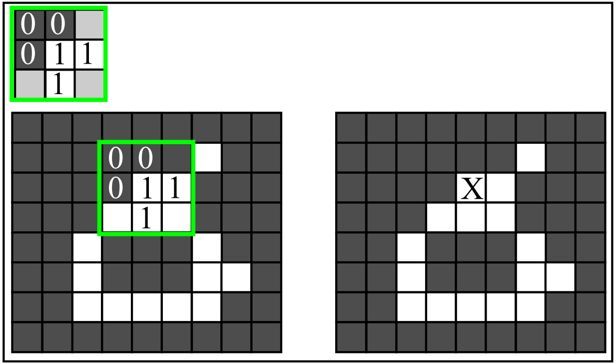
Scheletare- Exemplu



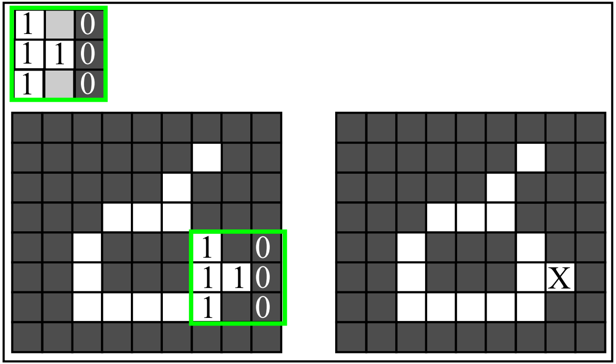
Scheletare- Exemplu



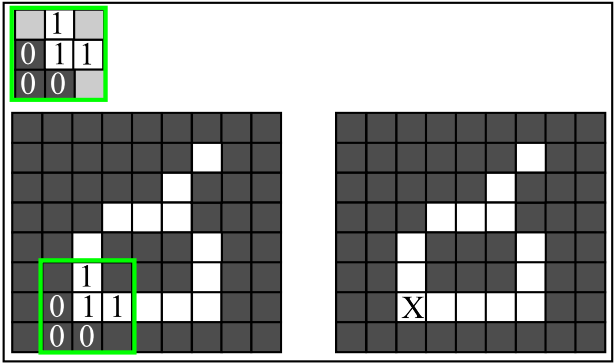
Scheletare- Exemplu



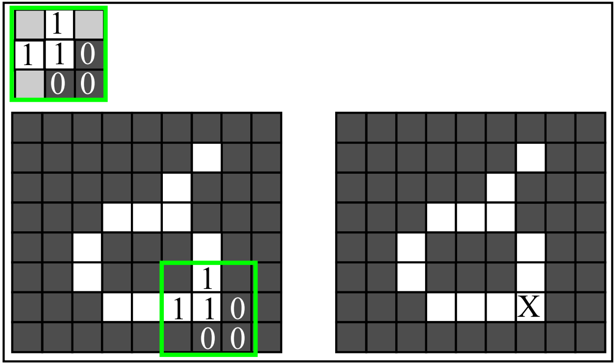
Scheletare- Exemplu



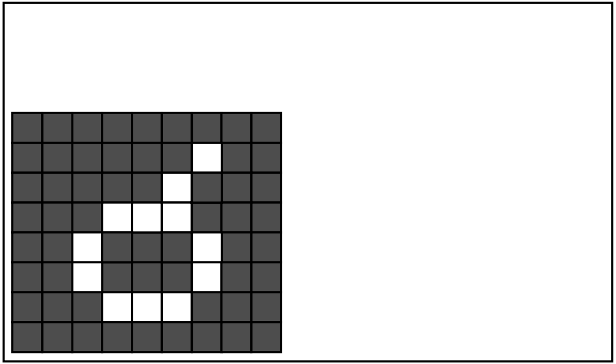
Scheletare- Exemplu



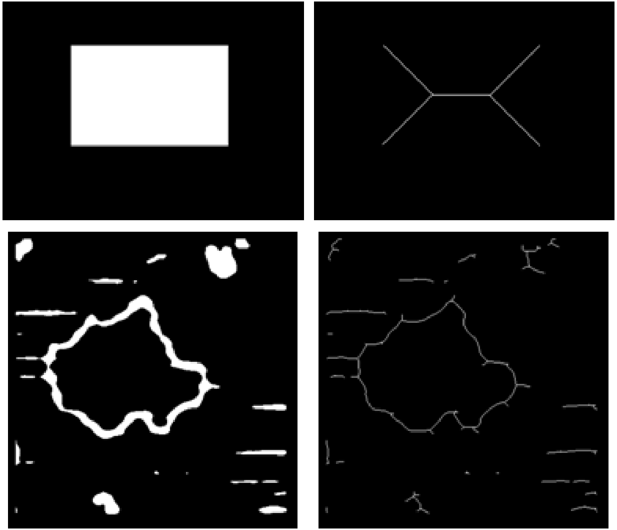
Scheletare- Exemplu



Scheletare- Exemplu



Scheletare - Exemple



Scheletare - Algoritm Zhang-Suen

P_9	P_2	P_3
P_8	P_1	P_4
P_7	P_6	P_5

Notații:

- P_1 - Pixelul procesat.
- $A(P_1)$ - Nr. de tranziții de la negru la alb în șirul de pixeli $P_2, P3, \ldots, P_9$.
- $B(P_1)$ - Nr. de pixeli albi în șirul de pixeli $P_2, P3, \ldots, P_9$.

Scheletare - Algoritm Zhang-Suen

P_9	P_2	P_3
P_8	P_1	P_4
P_7	P_6	P_5

Se iterează Pas 1 și Pas 2 cât timp apar modificări în imagine:

Pas 1

Se marchează pixelii albi $P_1 = P(x, y)$ pentru care sunt îndeplinite simultan condițiile:

1. $A(P_1) = 1$
2. $2 \leq B(P_1) \leq 6$
3. $P_2 \wedge P_4 \wedge P_6 = 0$
4. $P_4 \wedge P_6 \wedge P_8 = 0$

După parcurgerea întregii imagini se șterg pixelii marcați.

Scheletare - Algoritm Zhang-Suen

P_9	P_2	P_3
P_8	P_1	P_4
P_7	P_6	P_5

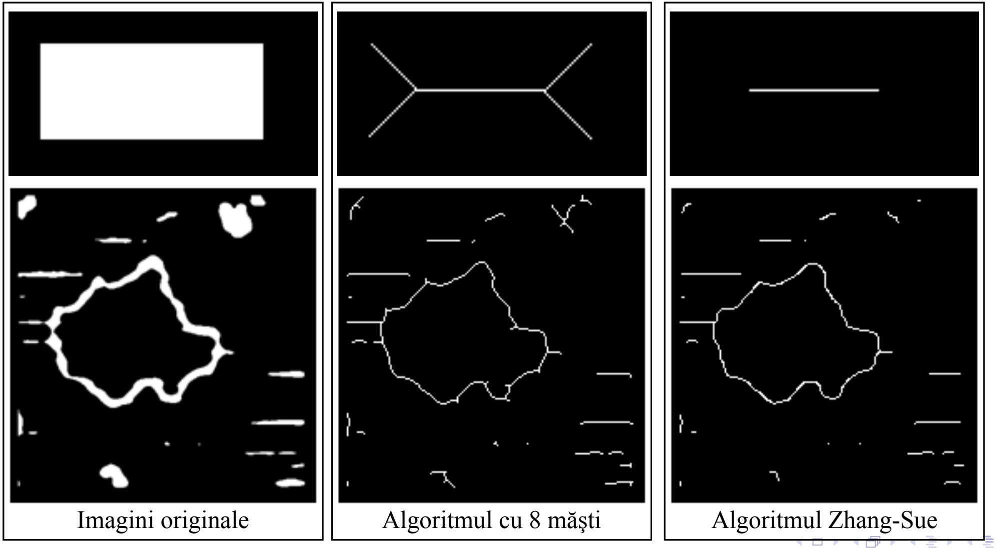
Pas 2

Se marchează pixelii albi $P_1 = P(x, y)$ pentru care sunt îndeplinite simultan condițiile:

1. $A(P_1) = 1$
2. $2 \leq B(P_1) \leq 6$
3. $P_2 \wedge P_4 \wedge P_8 = 0$
4. $P_2 \wedge P_6 \wedge P_8 = 0$

După parcurgerea întregii imagini se șterg pixelii marcați.
Observație: alternativă la condiția 2: $3 \leq B(P_1) \leq 6$.

Scheletare - Algoritm Zhang-Suen



Imagini originale | Algoritmul cu 8 măști | Algoritmul Zhang-Sue

Operații morfologice pentru imagini în tonuri de gri

- În cazul imaginilor binare: operații logice pe vecinătăți.
- În cazul imaginilor în tonuri de gri: operații de maxim și minim pe vecinătăți.

Operații morfologice pentru imagini în tonuri de gri

0	0	0
0	0	0
0	0	0

0	1	0
1	2	1
0	1	0

Dilatarea imaginii $f(x,y)$ cu masca $b(s,t)$, $s = \overline{-k,k}, t = \overline{-k,k}$:

$$(f \oplus b)(x,y) = max\{f(x-s, y-t) + b(s,t)|$$

$$s = \overline{-k,k}, t = \overline{-k,k}\}$$

Dilatare - Exemplu

Dilatare - Exemplu

Dilatare - Proprietăți

(1) Dacă toate valorile din masca b sunt pozitive, imaginea rezultat este mai deschisă decât imaginea originală.

(2) Detalii închise sunt reduse sau eliminate.

Operații morfologice pentru imagini în tonuri de gri

0	0	0
0	0	0
0	0	0

0	1	0
1	2	1
0	1	0

Erodarea imaginii $f(x, y)$ cu masca $b(s, t)$, $s = \overline{-k, k}, t = \overline{-k, k}$:

$$(f \ominus b)(x, y) = min\{f(x - s, y - t) - b(s, t)|$$

$$s = \overline{-k, k}, t = \overline{-k, k}\}$$

Erodare - Exemplu

Erodare - Exemplu

Erodare - Proprietăți

(1) Dacă toate valorile din masca b sunt pozitive, imaginea rezultat est mai închisă decât imaginea originală.

(2) Detalii deschise sunt reduse sau eliminate.

Opening și Closing

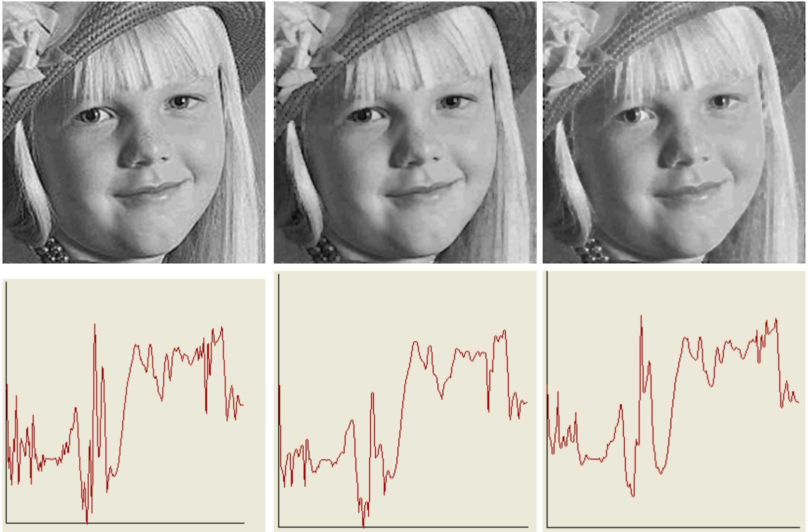
Opening:

$$f \circ b = (f \ominus b) \oplus b$$

Closing:

$$f \bullet b = (f \oplus b) \ominus b$$

Opening și Closing - Exemple



Opening Closing

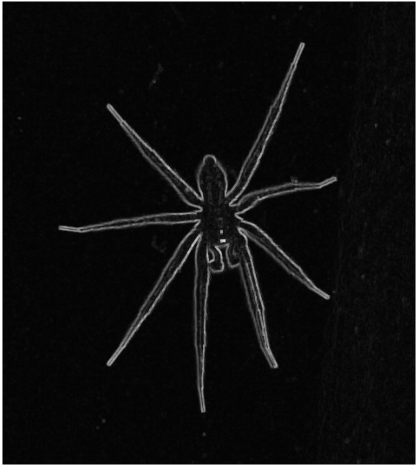
Opening și Closing - Aplicații

- **Opening**: - elimină mici detalii deschise, păstrând nivelul mediu de gri și detalii deschise mai mari neschimbate.
- **Closing**: - elimină mici detalii de culoare închisă.

Netezire morfologică

Procedeu: $g = (f \circ b) \bullet b$

Gradient morfologic



Procedeu: $g = (f \oplus b) - (f \ominus b)$

Rezumat

- Noțiuni introductive - operații logice cu regiuni binare.
- Operații morfologice pe imagini binare
  - Dilatarea
  - Erodarea
  - Opening și Closing
  - Extragerea contururilor: operatorul XOR
  - Detectarea componentelor conexe
  - Schelete
- Operații morfologice pe imagini în tonuri de gri
  - Dilatare și Erodare
  - Opening și Closing
  - Netezire morfologică
  - Gradiet morfologic