

# Concepțe și aplicații în Vederea Artificială

Bogdan Alexe

[bogdan.alexe@fmi.unibuc.ro](mailto:bogdan.alexe@fmi.unibuc.ro)

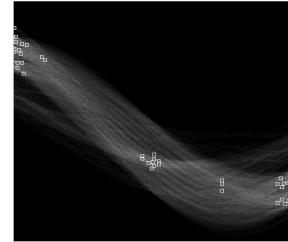
Radu Ionescu

[radu.ionescu@fmi.unibuc.ro](mailto:radu.ionescu@fmi.unibuc.ro)

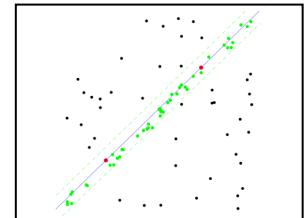
Curs optional  
anii III/IV, semestrul I, 2024-2025

# Cursul trecut

- Aplicație: detectarea liniilor cu
  - transformata Hough



- RANSAC



- Compararea contururilor

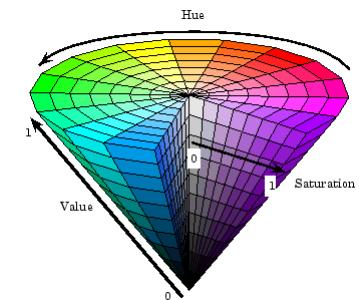
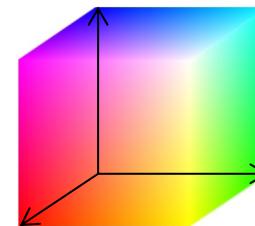
5 5

# Cursul de azi

- Compararea contururilor

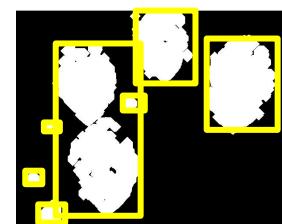
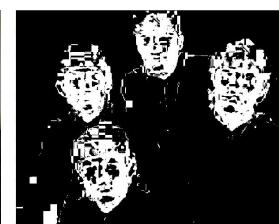


- Spațiile de culori RGB și HSV



- Imagini binare

- aplicarea unui prag
- operatori morfologici
- componente conexe
- proprietăți



- Recunoașterea claselor de obiecte

- privire de ansamblu
- ce este o clasă de obiecte?



# Distanța Chamfer

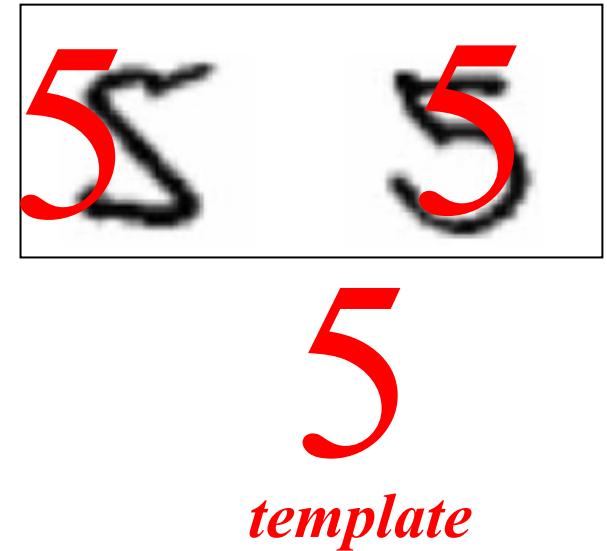
- Distanța medie dintre punctele unui **template  $T$**  și o mulțime de puncte (e.g. **puncte cu intensitate = 0**, puncte cu anumite caracteristici) dintr-o imagine I.

$$d_{chamfer}(\textcolor{red}{T}, I) = \frac{1}{|T|} \sum_{t \in T} d_I(\textcolor{red}{t})$$

$I$  = mulțime de puncte dintr-o imagine

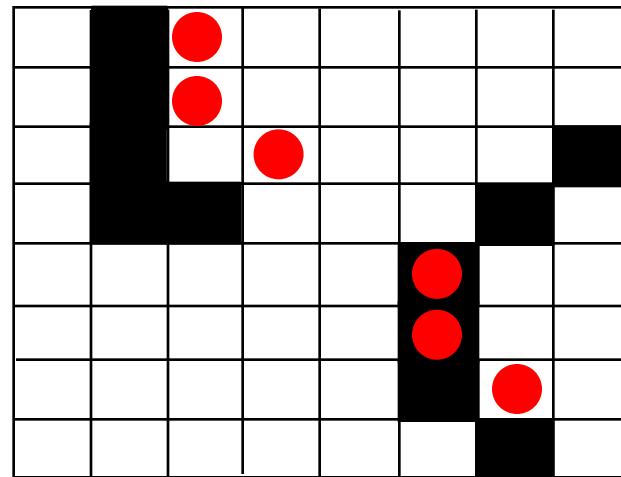
$T$  = mulțime de puncte ale unui **template**

$d_I(\textcolor{red}{t})$  = distanța minimă dintre **punctul  $t$**  și un punct din I  
(Manhattan, Euclidiană)



# Distanța Chamfer

Puncte din  $I$  (2D)

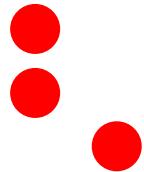


Distanța Manhattan:

$$\frac{1}{3}(1 + 1 + 2) = 1.33$$

$$\frac{1}{3}(0 + 0 + 1) = 0.33$$

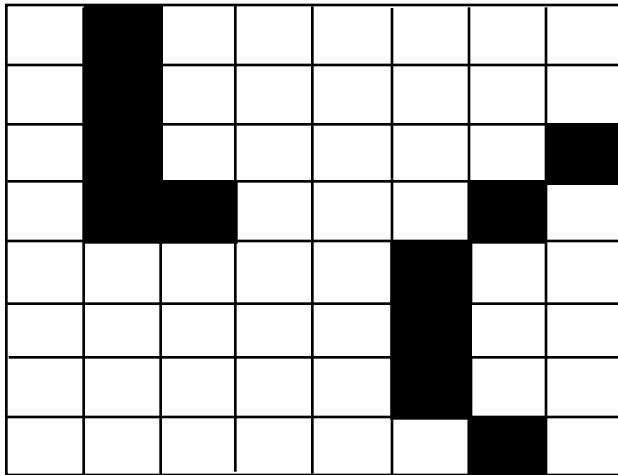
$T$



$$d_{chamfer}(T, I) = \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

# Transformata Distanță (distance transform)

Puncte din  $I$  (2D)



Transformata Distanță

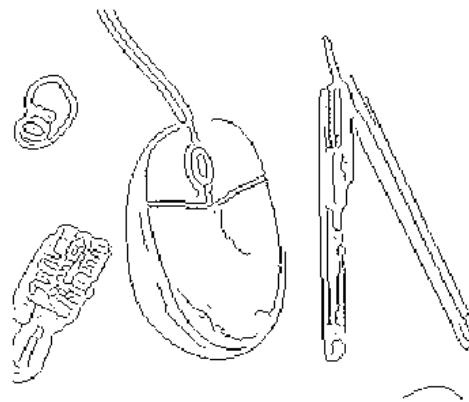
1	0	1	2	3	4	3	2
1	0	1	2	3	3	2	1
1	0	1	2	3	2	1	0
1	0	0	1	2	1	0	1
2	1	1	2	1	0	1	2
3	2	2	2	1	0	1	2
4	3	3	2	1	0	1	2
5	4	4	3	2	1	0	1

**Transformata Distanță** (TD) este o funcție care pentru fiecare pixel  $p$  asignează un număr pozitiv  $TD(p)$  corespunzând distanței de la  $p$  la cel mai apropiat punct din mulțimea  $I$

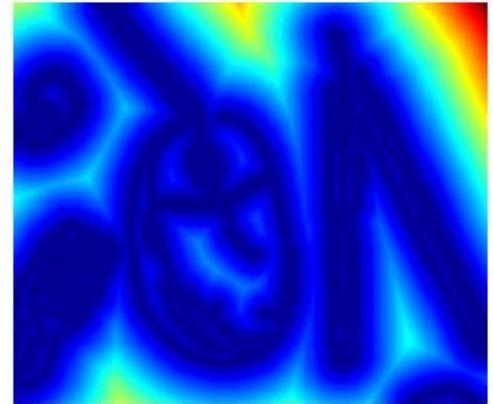
# Transformata Distanță



Imagine inițială



muchii



transformata distanță

Calculăm o singură dată TD(),  
apoi citim din acest tabel  
distanța la cel mai apropiat  
punct de pe muchie.

Valoarea la  $(x,y)$  indică cât de  
departe  $(x,y)$  este de cel mai  
apropiat punct de pe muchie  
**albastru – valori mici,**  
**roșu – valori mari**

**OpenCV: cv.distanceTransform**

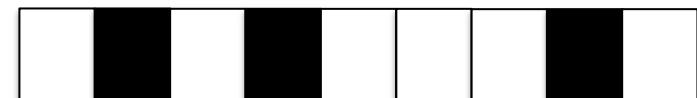
# Transformata Distanță – 1D

Două treceri de complexitate  $O(n)$

## 1. Inițializare

```
for j = 1:n  
    TD(j) = {  
        0, dacă j este în I  
        ∞, altfel  
    }  
end
```

*Exemplu*



∞	0	∞	0	∞	∞	∞	0	∞
---	---	---	---	---	---	---	---	---

## 2. Trecere înapoi (vecin stânga)

```
for j = 2:n  
    TD(j) = min(TD(j),TD(j-1)+1)  
end
```

∞	0	1	0	1	2	3	0	1
---	---	---	---	---	---	---	---	---

## 3. Trecere înapoi (vecin dreapta)

```
for j = n-1:-1:1  
    TD(j) = min(TD(j),TD(j+1)+1)  
end
```

1	0	1	0	1	2	1	0	1
---	---	---	---	---	---	---	---	---

# Transformata Distanță – 2D

Analog cu 1D

1. Inițializare

$$TD(i,j) = \begin{cases} 0, & \text{dacă } (i,j) \text{ este în } I \\ \infty, & \text{altfel} \end{cases}$$

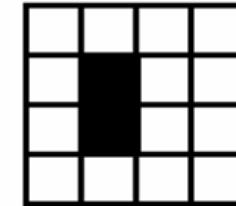
2. Trecere înapoi (vecin stânga, sus)

$$TD(i,j) = \min(TD(i,j), TD(i,j-1)+1, TD(i-1,j)+1)$$

3. Trecere înapoi (vecin dreapta, jos)

$$TD(i,j) = \min(TD(i,j), TD(i,j+1)+1, TD(i+1,j)+1)$$

*Exemplu*



$\infty$	$\infty$	$\infty$	$\infty$
$\infty$	<b>0</b>	$\infty$	$\infty$
$\infty$	<b>0</b>	$\infty$	$\infty$
$\infty$	$\infty$	$\infty$	$\infty$

$\infty$	$\infty$	$\infty$	$\infty$
$\infty$	<b>0</b>	<b>1</b>	$\infty$
$\infty$	<b>0</b>	$\infty$	$\infty$
$\infty$	$\infty$	$\infty$	$\infty$

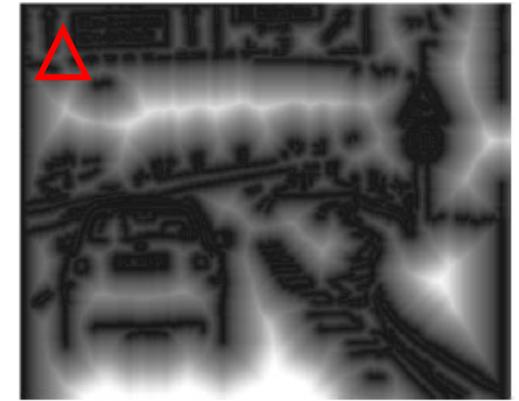
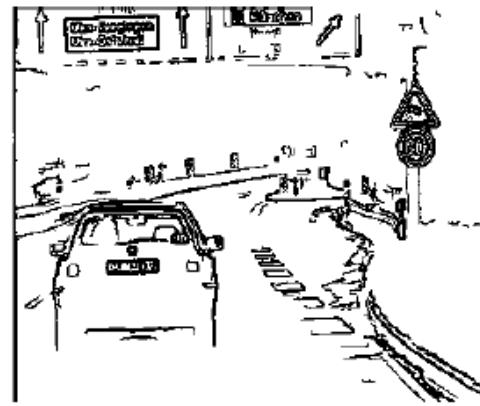
$\infty$	$\infty$	$\infty$	$\infty$
$\infty$	<b>0</b>	1	2
$\infty$	0	1	2
$\infty$	1	2	3

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

# Distanța Chamfer

- distanța medie până la cel mai apropiat punct din  $I$

$$D_{chamfer} = (T, I) = \frac{1}{|T|} \sum_{t \in T} d_I(t)$$



Complexitate implementare naivă:  
 $|T| * |I|^2$  operații

Complexitate transformata distanță:  
 $|T| * |I|$  operații

Imagine cu muchii

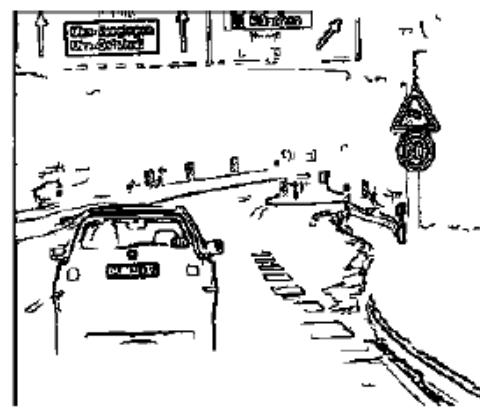
transformata distanță  
alb – distanță mare,  
negru - distanță mică

# Recunoașterea semnelor de circulație

**Recunoaștere** = detectare + clasificare

**Detectare** = localizăm conturul în imagine  
(pe baza de template: triunghi, cerc)

**Clasificare** = analizăm pixelii din interiorul  
conturului și clasificăm semnul de  
circulație într-o clasă (limitare de viteză 60  
km, limitare de viteză 80 km, semafor, etc)



Imagine cu muchii



transformata distanță  
alb – distanță mare,  
negru - distanță mică

# Recunoașterea semnelor de circulație



condiții de zi



condiții de noapte

# Detectarea pietonilor

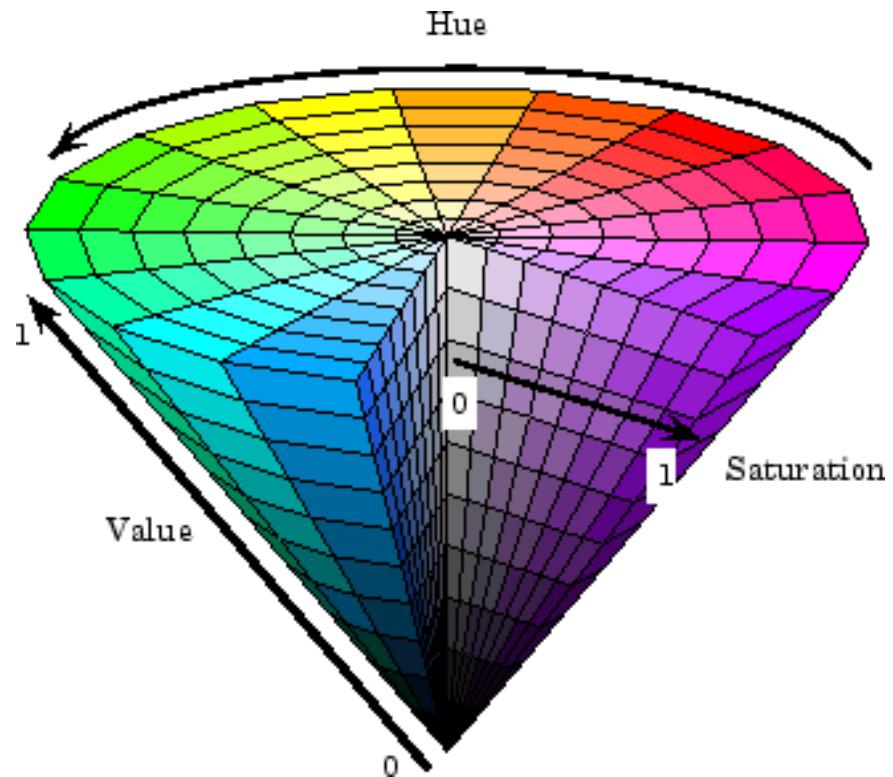
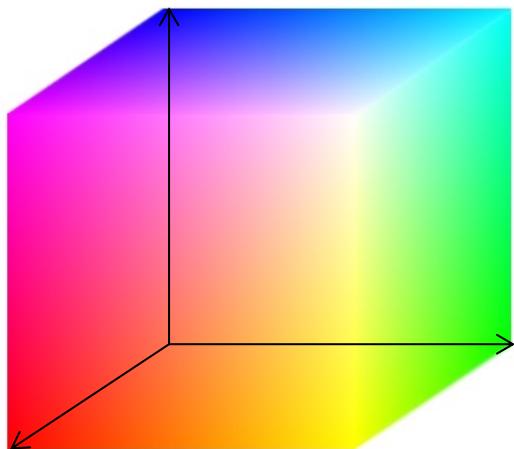
Detectare pietonilor = localizăm  
contururi în imagine asemănătoare cu  
contururi de pietoni dintr-o mulțime de  
template-uri



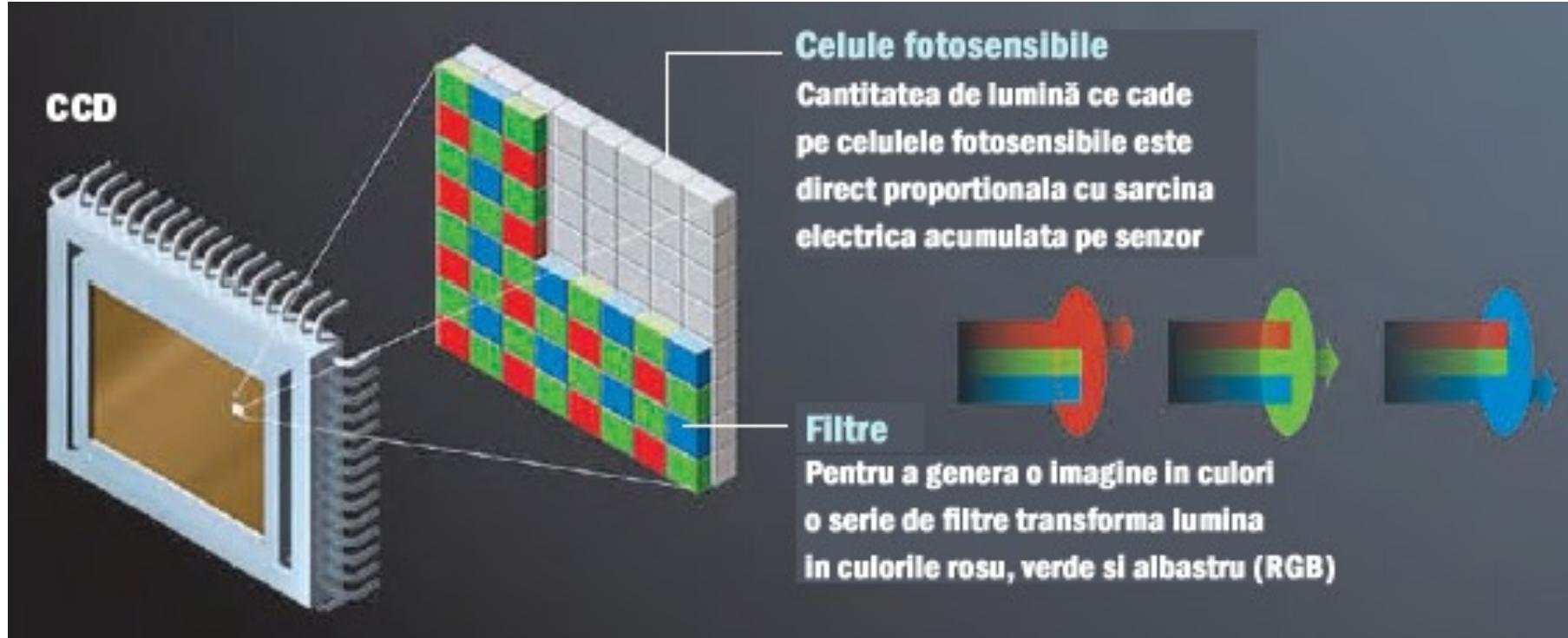
# Distanța Chamfer: proprietăți

- Avantaje
  - simplă de implementat
  - foarte rapidă
  - tolerantă la forme care diferă puțin de template
- Dezavantaje
  - sensitivă la mărime (scale) și rotație
  - număr mare de template-uri pentru forme diverse ca mărime și ca rotație
  - detectii false în regiuni cu multe muchii

# Spațiile culori RGB și HSV



# Imagini digitale color



Senzor de imagine

Filtru Bayer

# Tipuri de imagini digitale

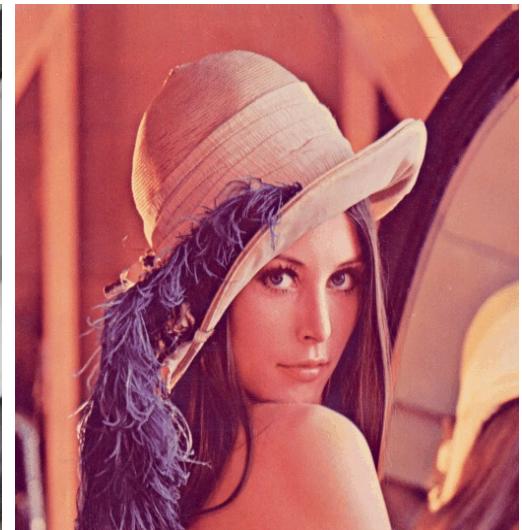
Binare



Grayscale  
(tonuri de gri)



Color



Luminozitate	negru, alb	tonuri de gri	R G B
Valori	{0,1}	{0, ..., 255}	{0, ..., 255} <sup>3</sup>
Culori	negru - 0, alb - 1	negru - 0, gri - 128, alb - 255	(255,0,0), (0,255,0), (0,0,255), (0,0,0), (255,255,255), (255,255,0), (255,125,0), (0,255,255), (255,0,255)
Memorie/pixel	1 bit/pixel	8 biți/pixel	24 biți/pixel

# RGB2GRAY

Color



Grayscale  
(tonuri de gri)



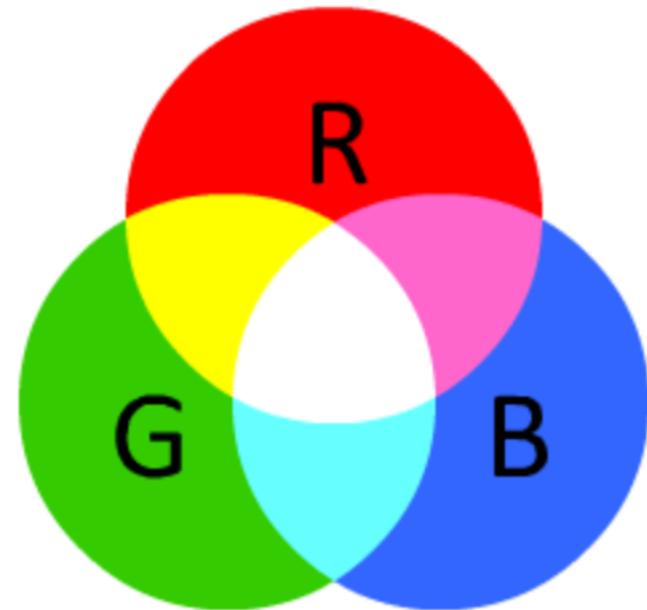
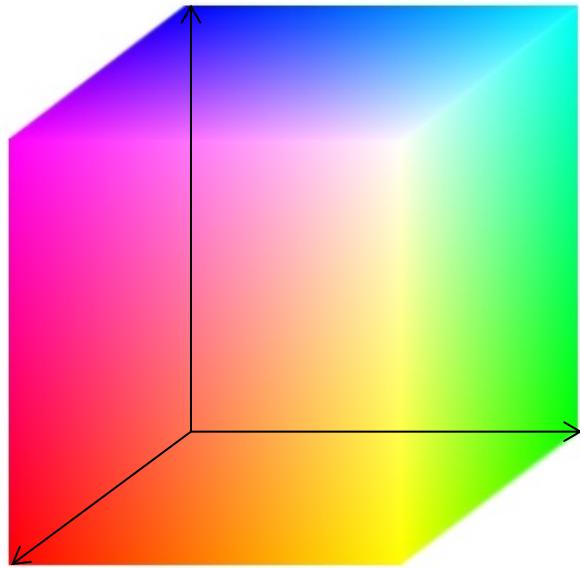
$$\text{gray} = 0.2989 * \text{R} + 0.5870 * \text{G} + 0.1140 * \text{B}$$

coeficienți determinați pe  
baza perceptiei vizuale umane

[https://en.wikipedia.org/wiki/Color\\_vision](https://en.wikipedia.org/wiki/Color_vision)



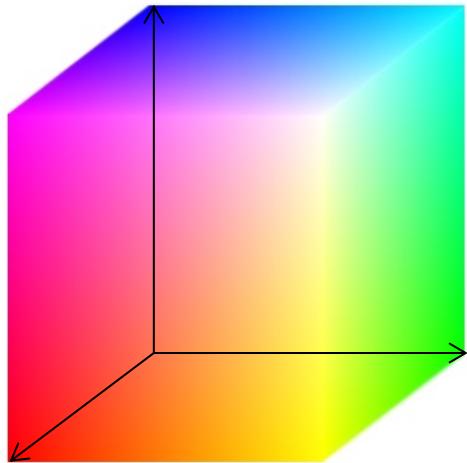
# RGB



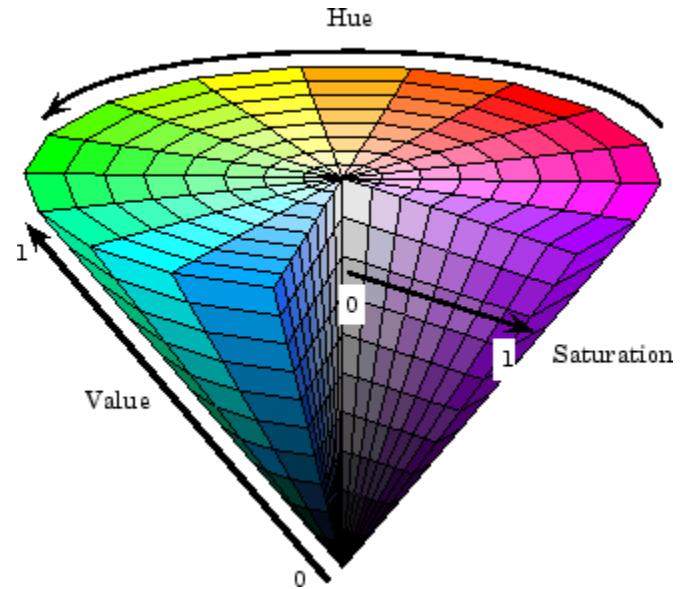
Spațiu de culori liniar – fiecare culoare e o combinație liniară de cele trei culori primare R, G ,B

Nu separă luminozitatea de culoare, nu e robust la schimbări de luminozitate ale scenei, umbre, etc.

# RGB vs HSV



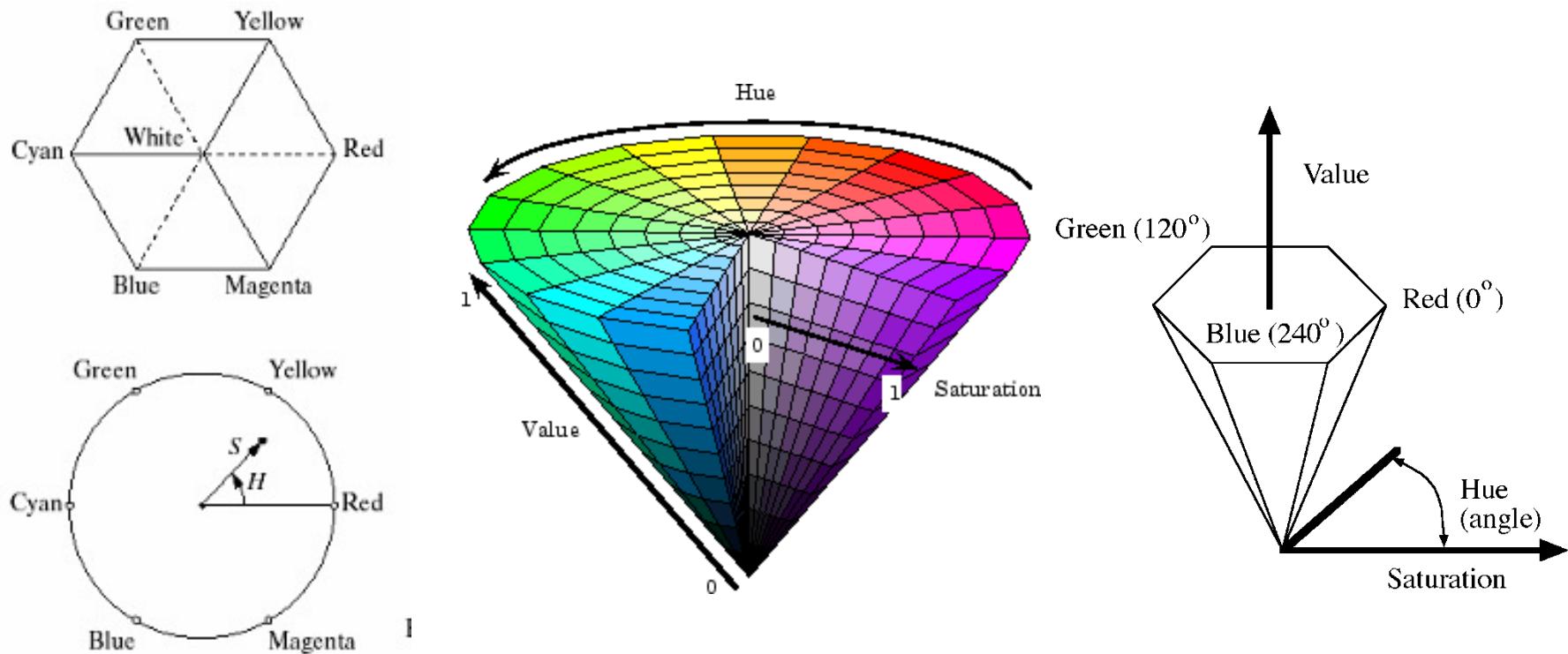
Nu separă luminozitatea de culoare, nu e robust la schimbări de luminozitate ale scenei, umbre, etc.



Separă luminozitatea de culoare, e robust la schimbări de luminozitate ale scenei, umbre, etc.

Demo: <https://colorizer.org/>

# Spațiu de culori neliniar HSV



- definit pe baza modului în care oamenii percep culorile
  - Hue = nuanță: ce culoare
  - Saturation = saturație: cât de multă culoare
  - Value = valoare: cât de multă luminozitate

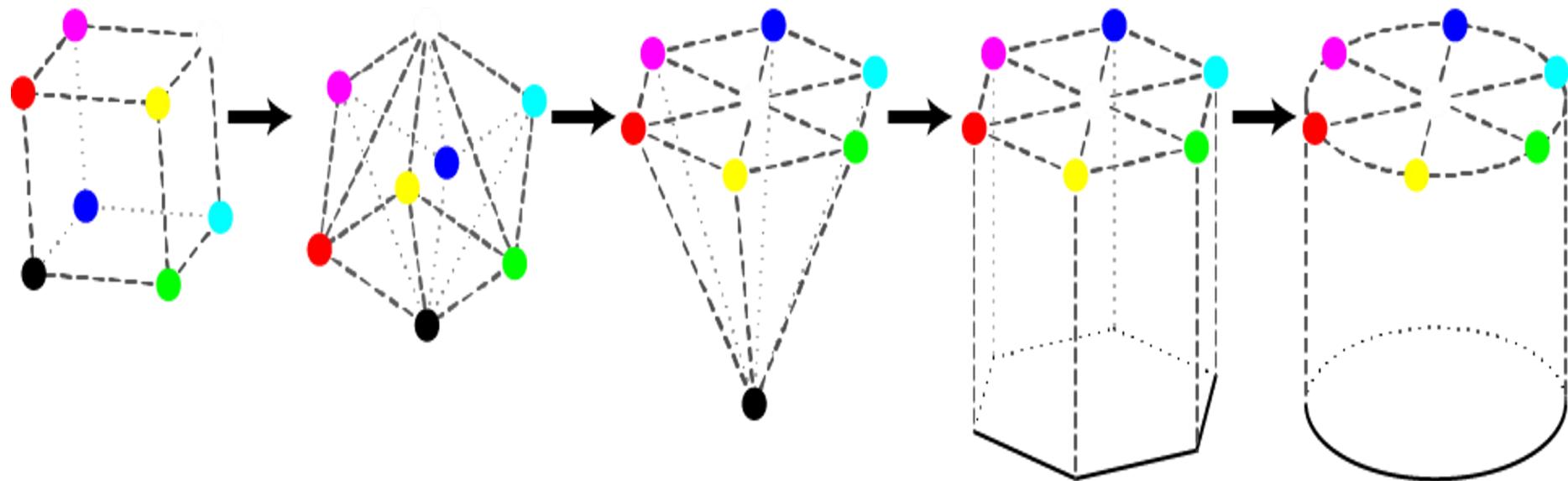
# RGB2HSV

tilt cube,  
add seems

squash colors  
to same plane

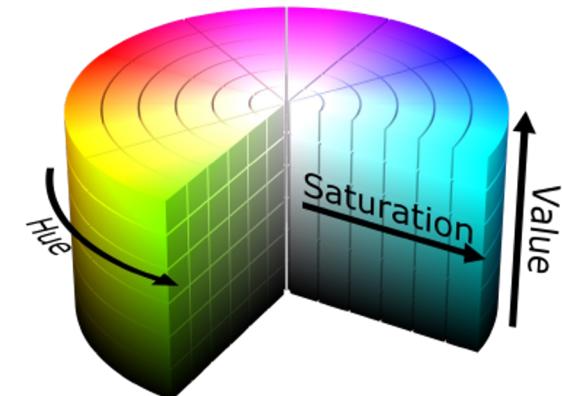
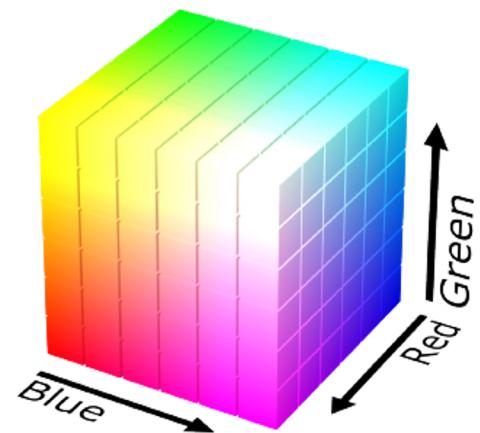
expand base

smooth cylinder

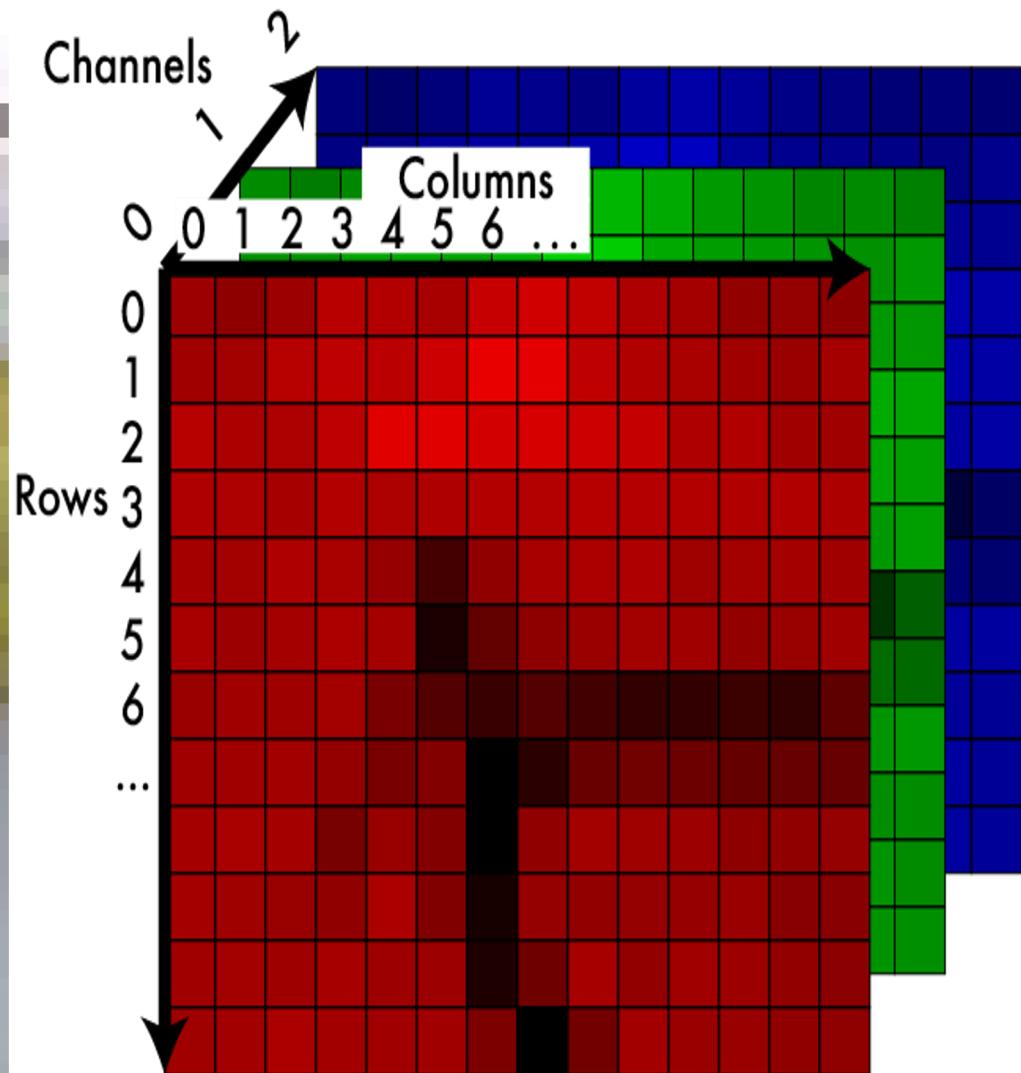
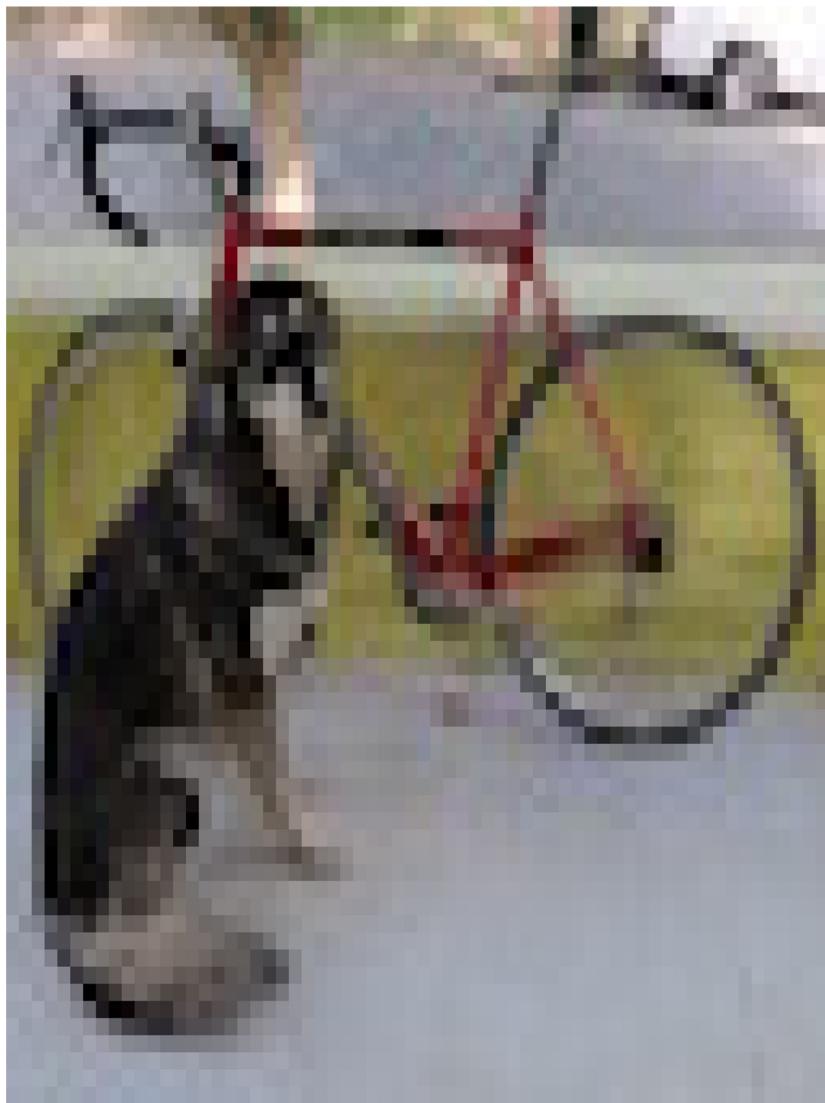


RGB

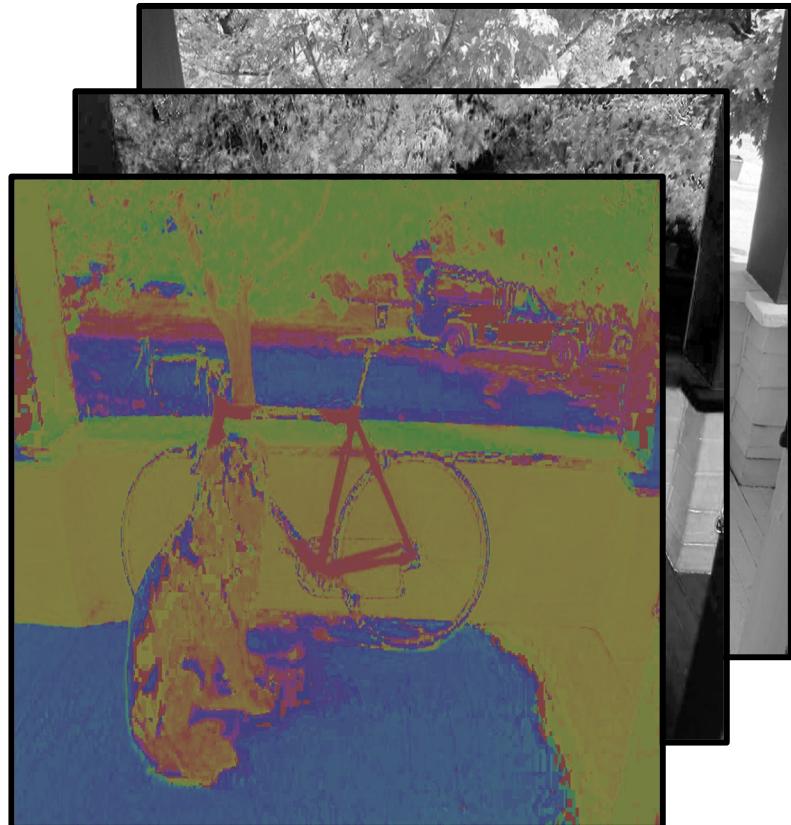
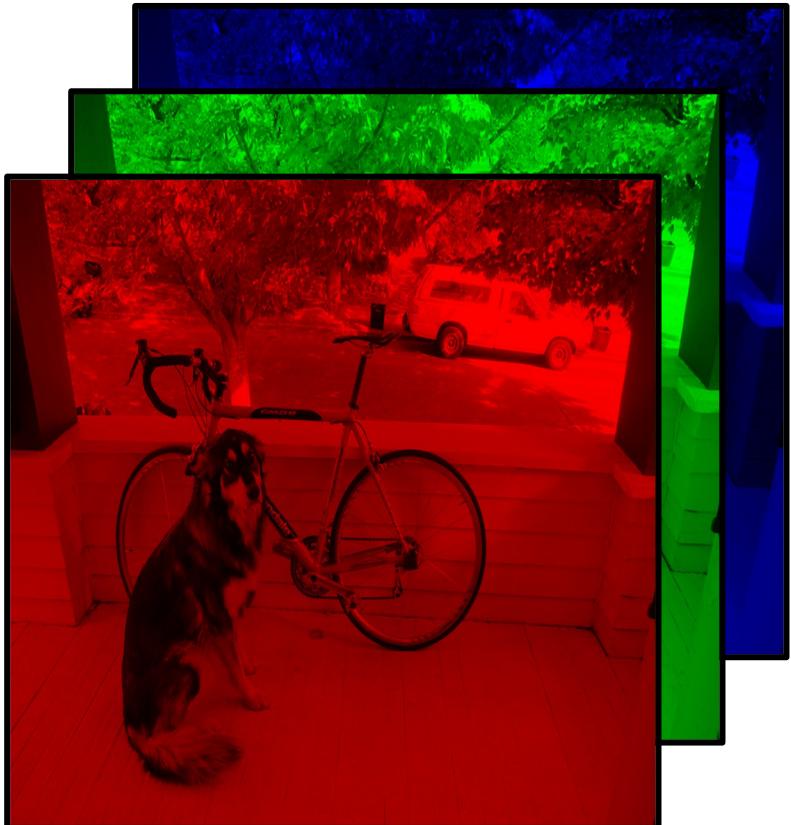
HSV



# Imagine în spațiul de culori RGB

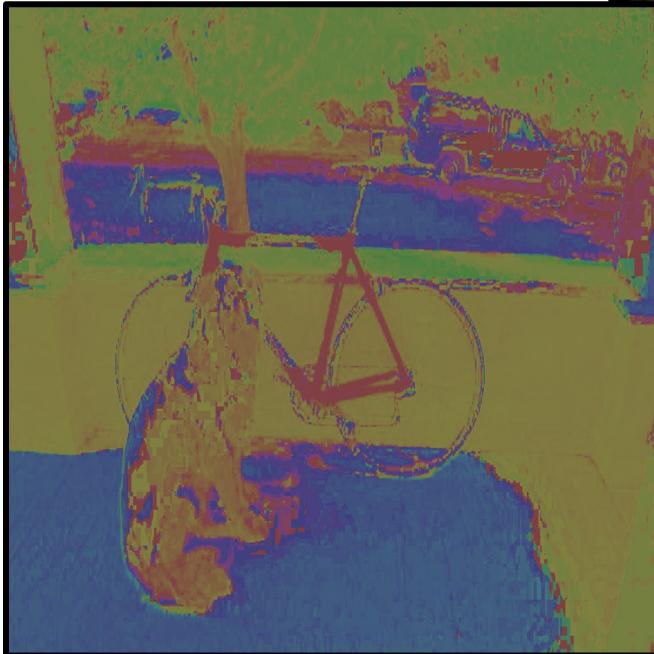


# RGB vs HSV



# Saturation

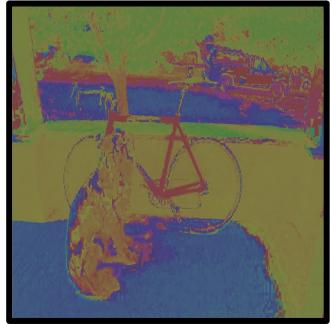
Hue



Value



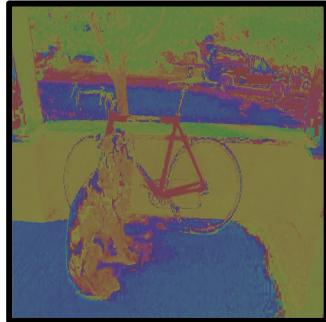
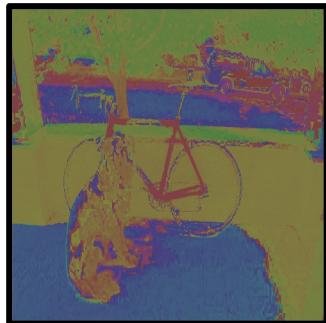
# Creștem saturația = culori mai intense



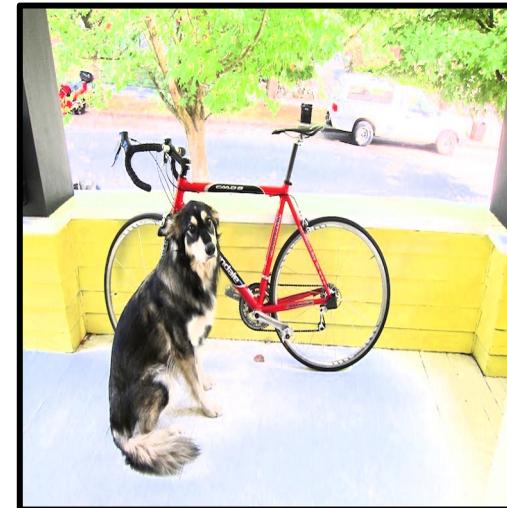
2x



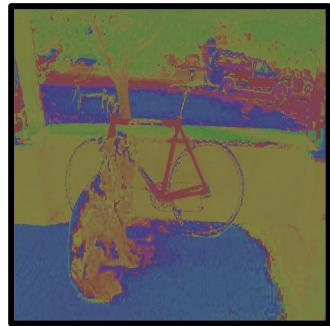
# Creștem valoarea = imagine cu luminozitate mai mare



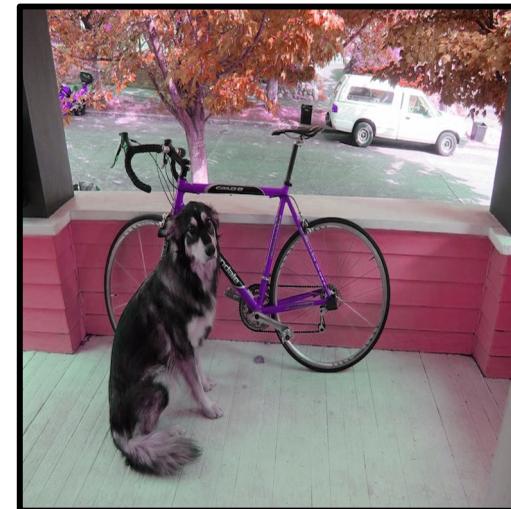
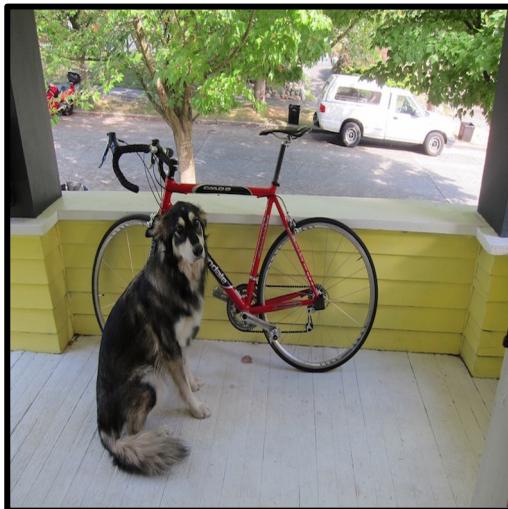
2x



# Modificăm nuanță = modificăm culorile

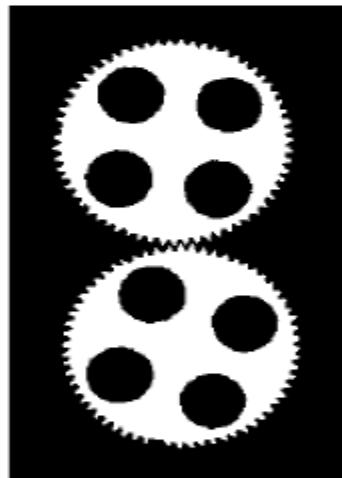
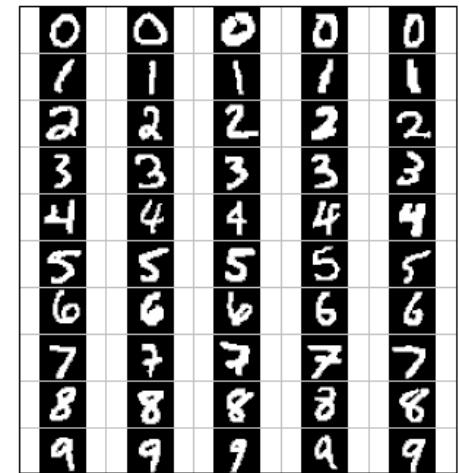
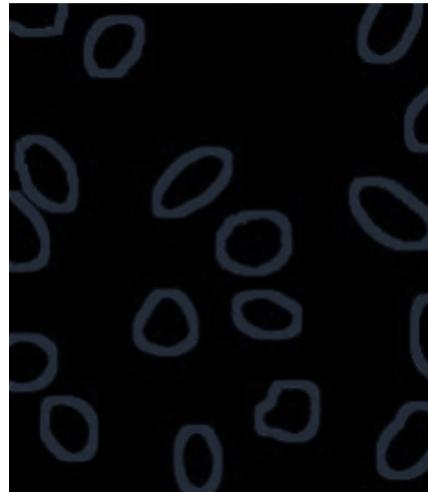
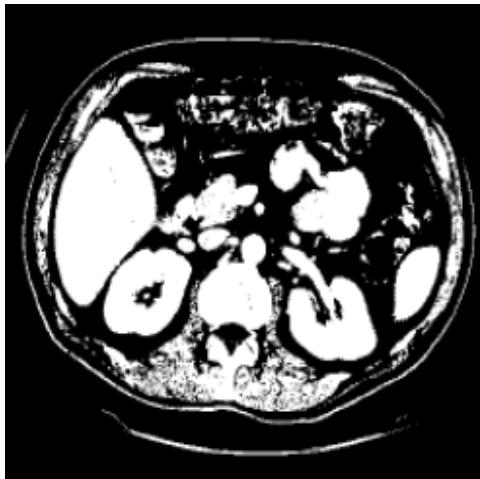


- .2



Imagini binare

# Imagini binare



A well-known model for the human  
bi-symmetry stage

Very young life

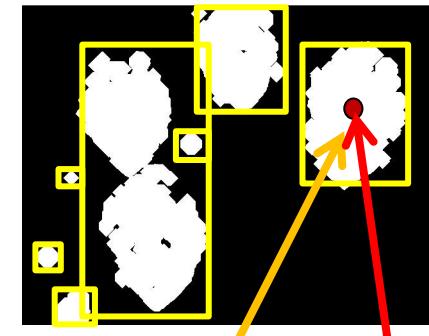
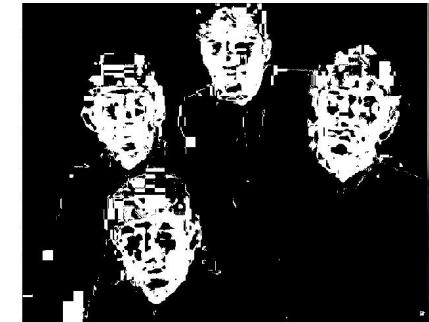
The human brain is organized to process both right and left visual information. This organization of the brain is called lateralization. In the first few months of life, the brain undergoes a process called "maturation". This maturation is characterized by the development of specific brain regions and connections. One of the most important stages of brain development is the "bi-symmetry stage", which occurs between 3 and 12 months of age. During this stage, the brain begins to show a preference for one side over the other, leading to the development of specialized functions for each hemisphere. This stage is also associated with the development of basic cognitive abilities, such as object recognition and memory.

Infancy

The first year of life is a period of rapid growth and development. The brain grows from approximately 350 grams at birth to about 1000 grams by the time the child is one year old. This growth is primarily due to the increase in the number of neurons and synapses in the brain. The brain also undergoes significant pruning, where unnecessary connections are removed to allow for more efficient processing. The brain also begins to develop specialized regions for different functions, such as language and movement.

# Analiza imaginilor binare: pași de urmat

- Convertim imaginea în imagine binară
  - aplicarea unui prag
- Curățăm imaginea obținută
  - operatori morfologici (dilatare, eroziune)
- Extragem regiuni separate
  - componente conexe
- Caracterizăm regiunile cu anumite proprietăți



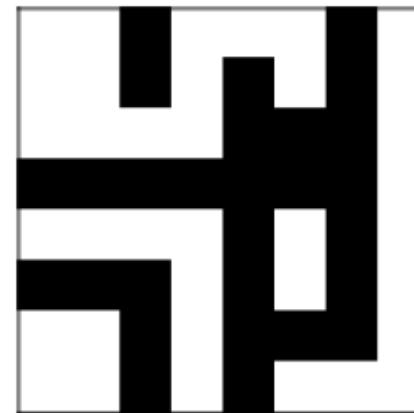
aria=370 pixeli

centroidul

# Imagini binare

- Pixelii iau două valori
  - foreground – valoare 1, pixeli albi, obiectul de interes
  - background – valoare 0, pixeli negri, mediul ambient

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



# Aplicarea unui prag (thresholding)

- Imagine de intensitate (tonuri de gri) → imagine binară
- Folositoare dacă ceea ce căutăm în imagine are o distribuție a intensităților diferită de background

$$F_T(i,j) = \begin{cases} 1, & \text{dacă } F(i,j) \geq T \\ 0, & \text{altfel} \end{cases}$$

$$F_{T_1, T_2}(i,j) = \begin{cases} 1, & \text{dacă } T_1 \leq F(i,j) \leq T_2 \\ 0, & \text{altfel} \end{cases}$$

# Aplicarea unui prag

- Imagine de intensitate (tonuri de gri) → imagine binară

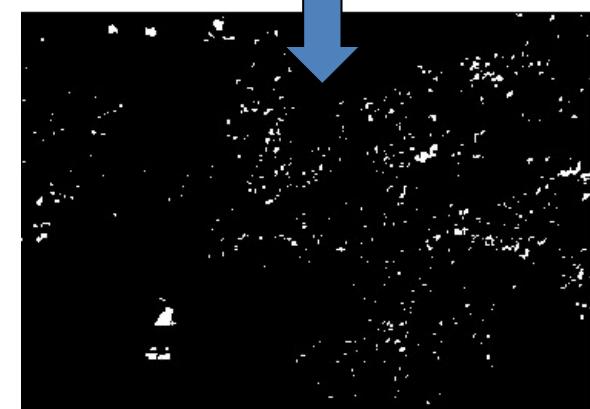
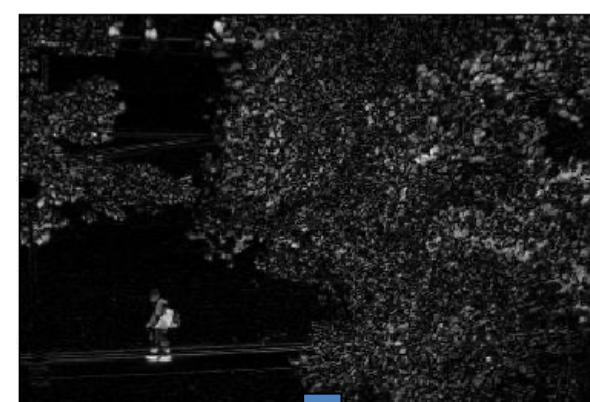
Exemplu: extragerea background-ului



-



=



Căutăm pixeli care diferă semnificativ de background

# Aplicarea unui prag

- Imagine de intensitate (tonuri de gri) → imagine binară

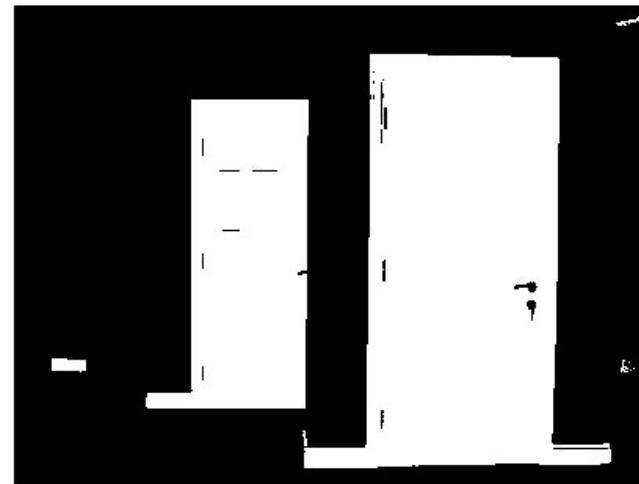
Exemplu: extragerea background-ului



# Aplicarea unui prag

- Imagine de intensitate (tonuri de gri) → imagine binară

Exemplu: detectarea pe bază de intensitate



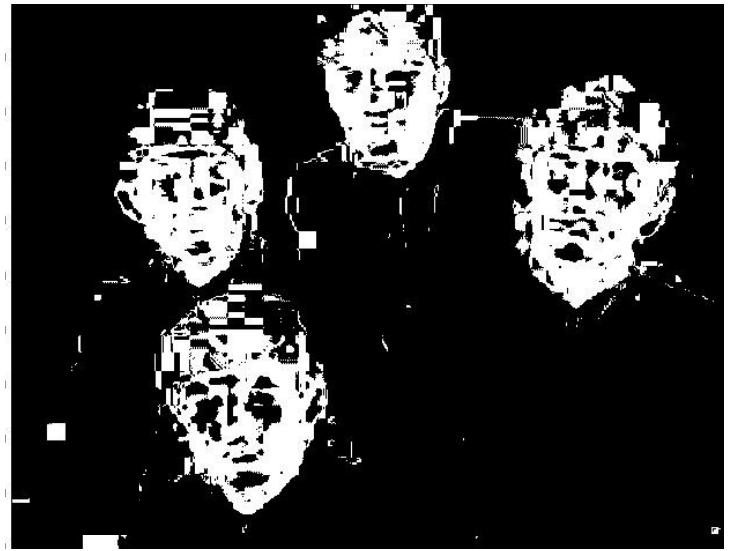
`fg_pix = find(im < 65);`

Căutăm pixelii cu intensitatea scăzută

# Aplicarea unui prag

- Imagine de intensitate (tonuri de gri) → imagine binară

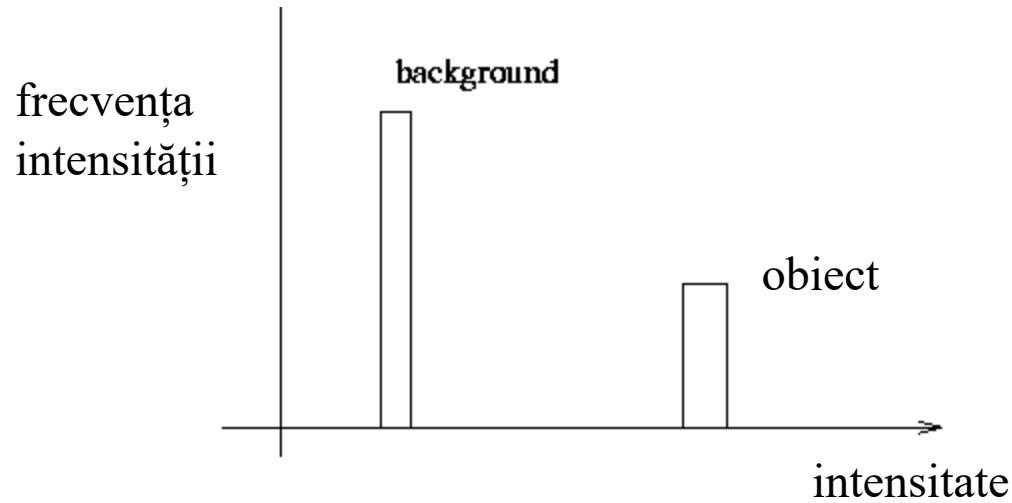
Exemplu: detectarea pe bază de culori



`fg_pix = find(hue > t1 & hue < t2);`

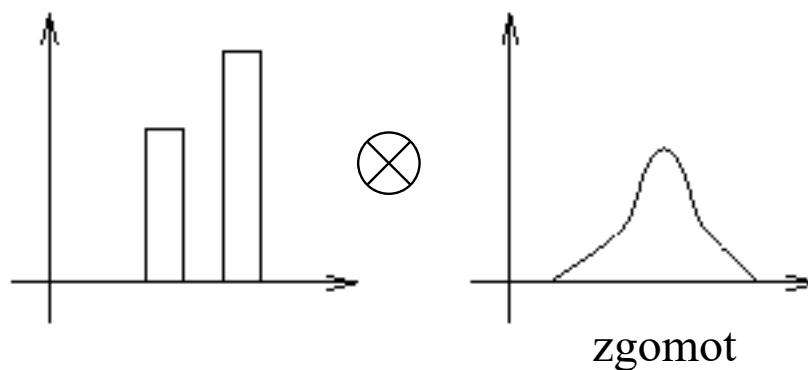
Căutăm pixeli cu o anumită nuanță a culorii (culoarea pielii).

# Caz ideal: histogramme de intensitate bimodale

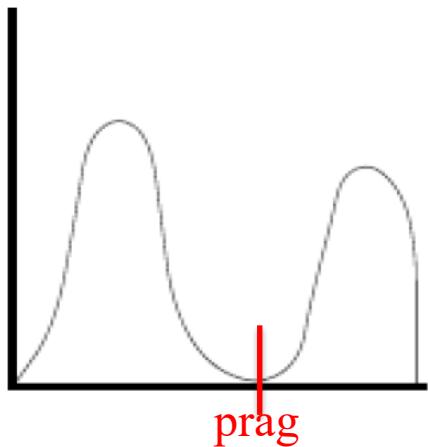


cazul ideal: histogramă bi-modală

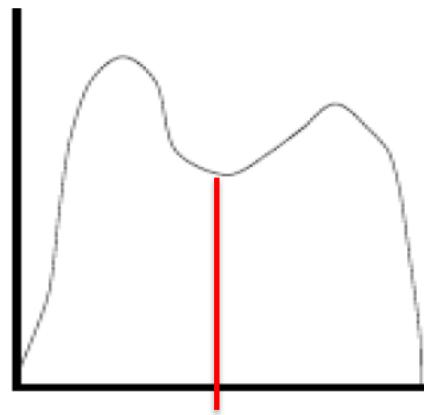
obiect luminos într-un background întunecat



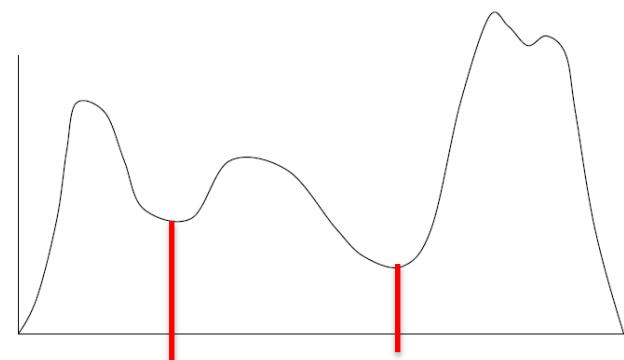
# Cazuri mai puțin ideale



două distribuții distincte

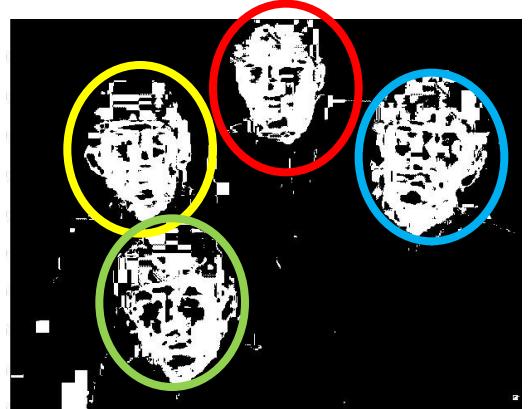
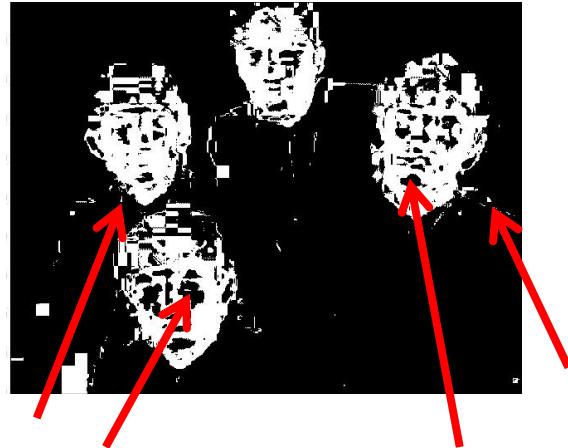


distribuții ce se suprapun



# Probleme

- Ce facem cu imaginile binare rezultate ca cea din dreapta?
  - găuri
  - fragmente foarte mici, izolate
- Cum delimităm regiunile de interes?
  - numărăm obiecte
  - calculăm diverse proprietăți pentru un obiect

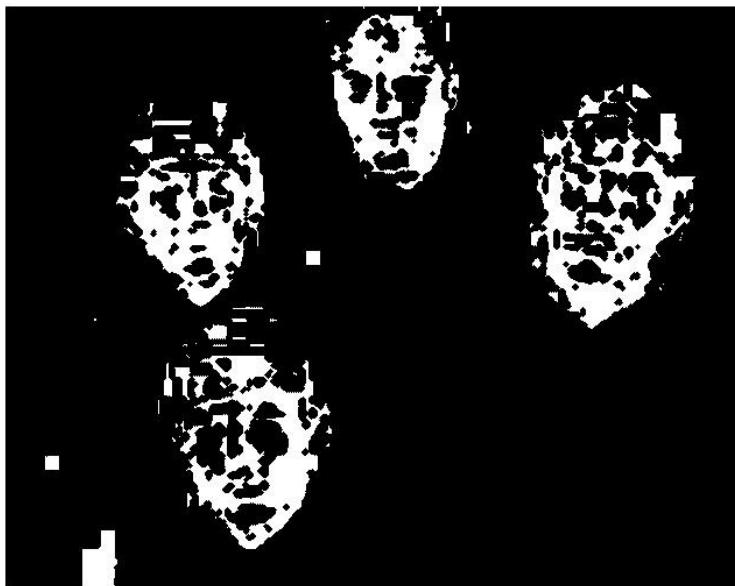


# Operatori morfologici

- Schimbă forma regiunilor de foreground folosind operații de intersecție/uniune dintre un element structural și o imagine binară.
- Eliminăm efectele nedorite rezultate în urma aplicării unui prag
- Operatorii de bază:
  - Dilatare
  - Eroziune

# Dilatare

- Mărește componentele conexe
- Adaugă pixeli albi
- Umple găurile



Înainte de dilatare



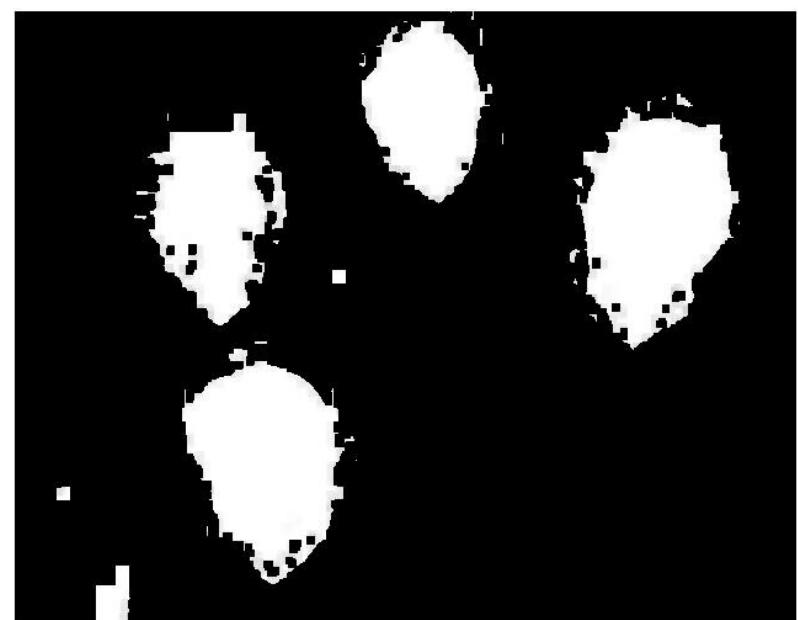
După dilatare

# Eroziune

- Micșorează componentele conexe
- Elimină pixeli albi
- Elimină fragmentele mici, zgomot



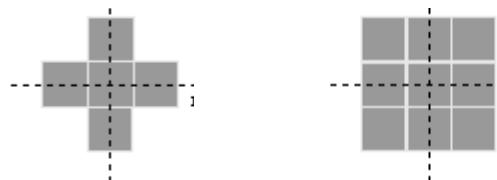
Înainte de eroziune



După eroziune

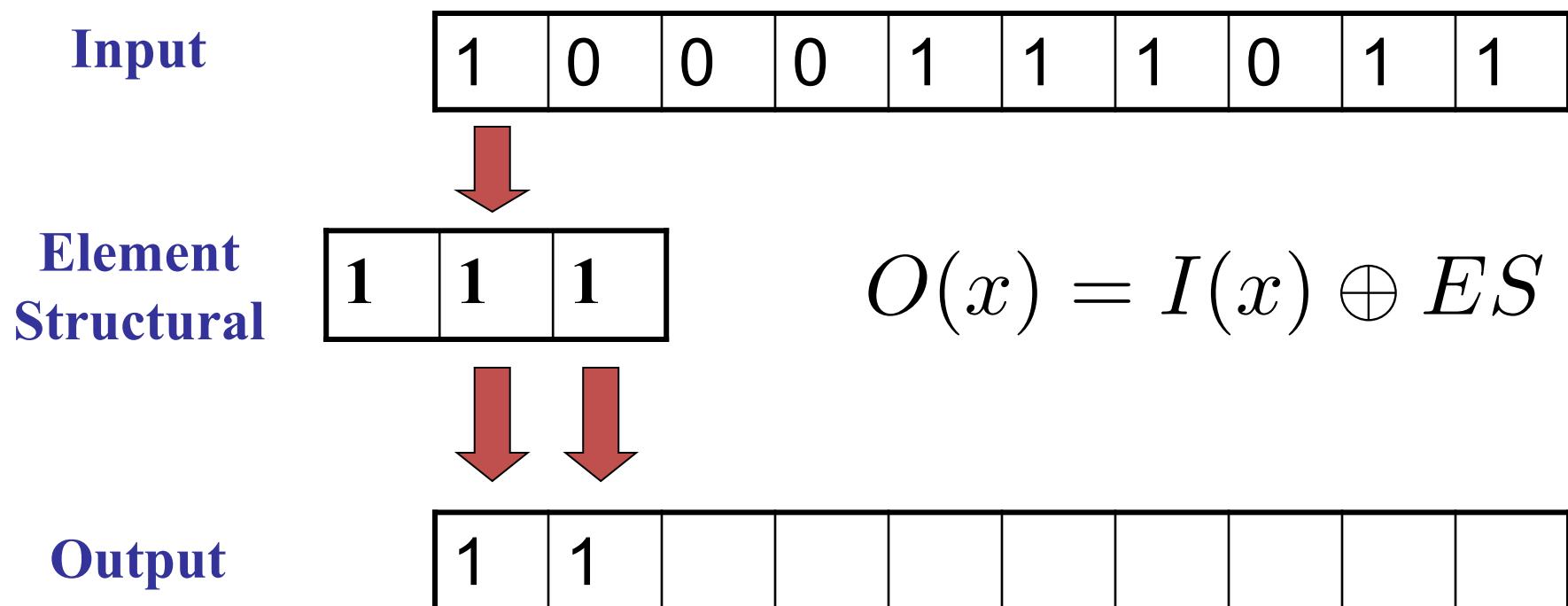
# Elemente structurale

- **Măști** de diverse forme și dimensiuni folosite de operatori morfologici:



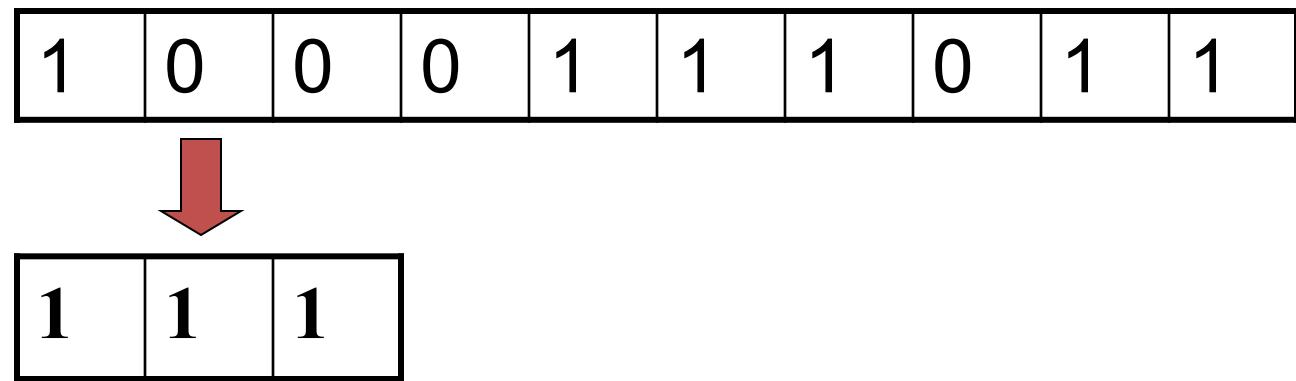
- Suprapunem masca peste imaginea binară (un fel de corelație)

# Dilatare – exemplu (1D)

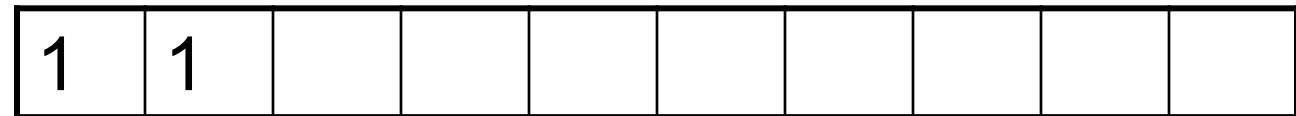


# Dilatare – exemplu

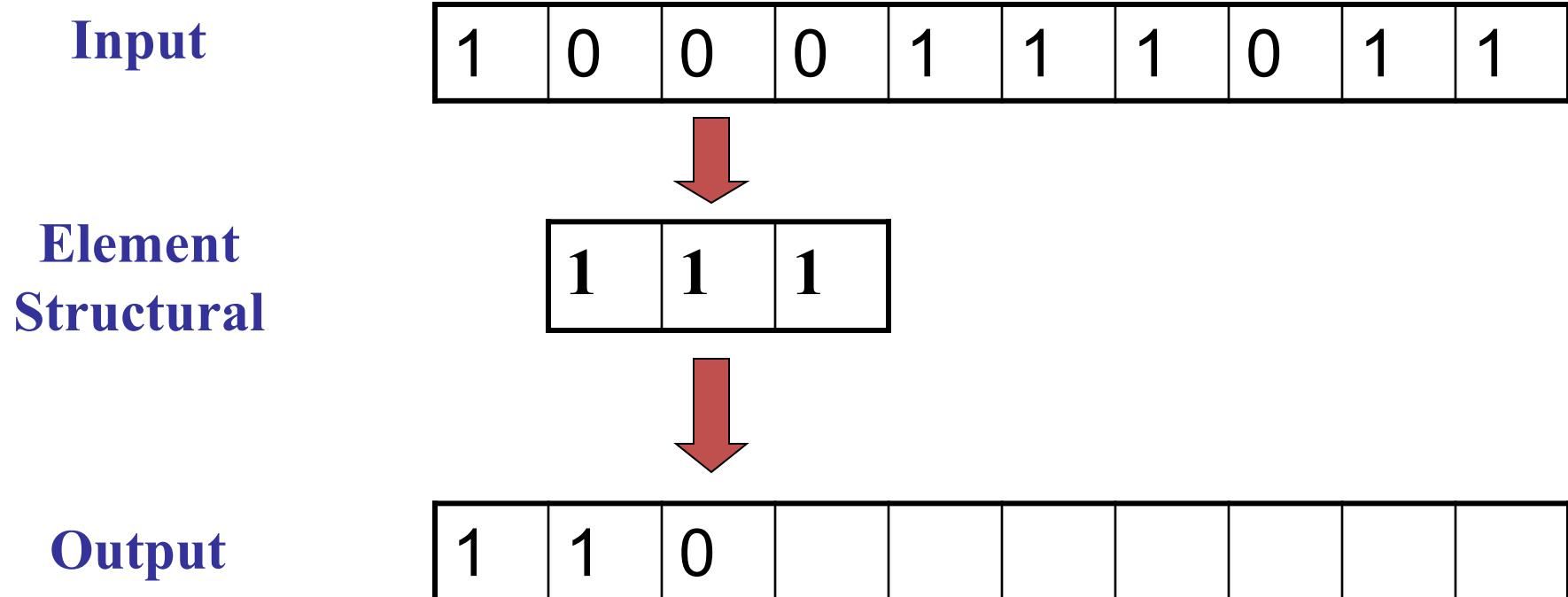
**Input**  
**Element Structural**



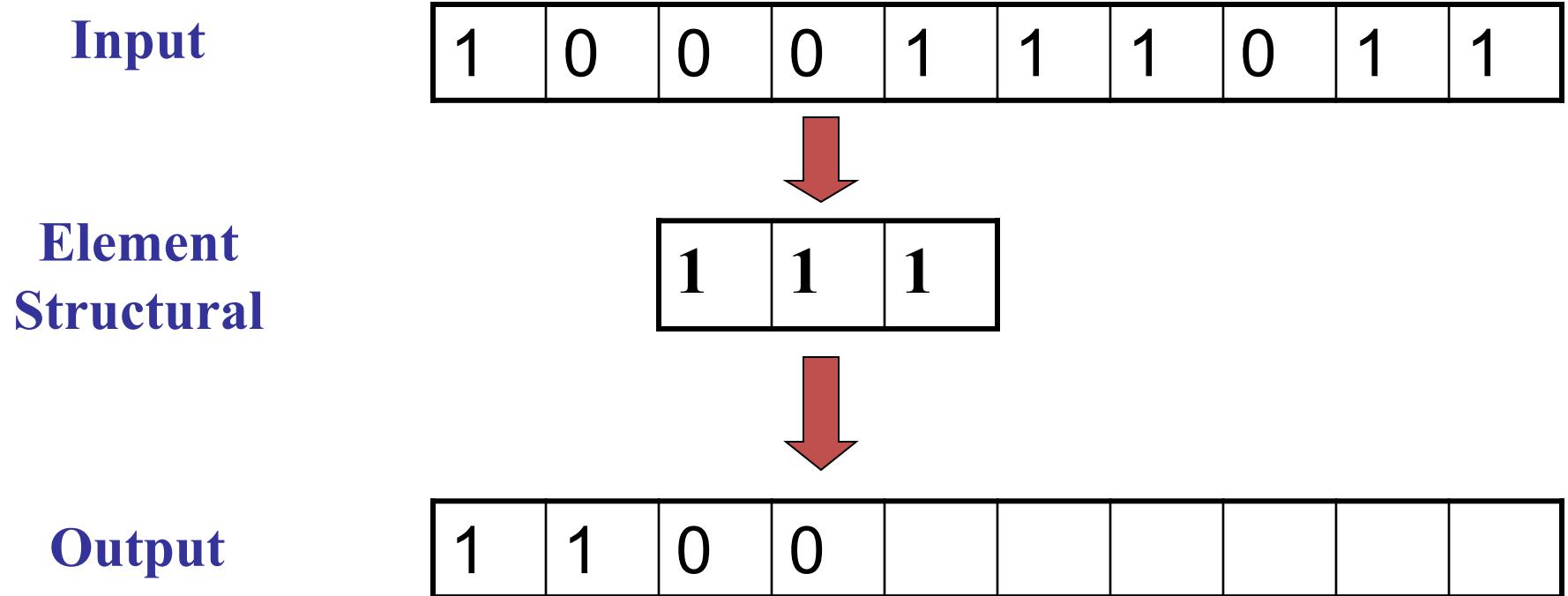
**Output**



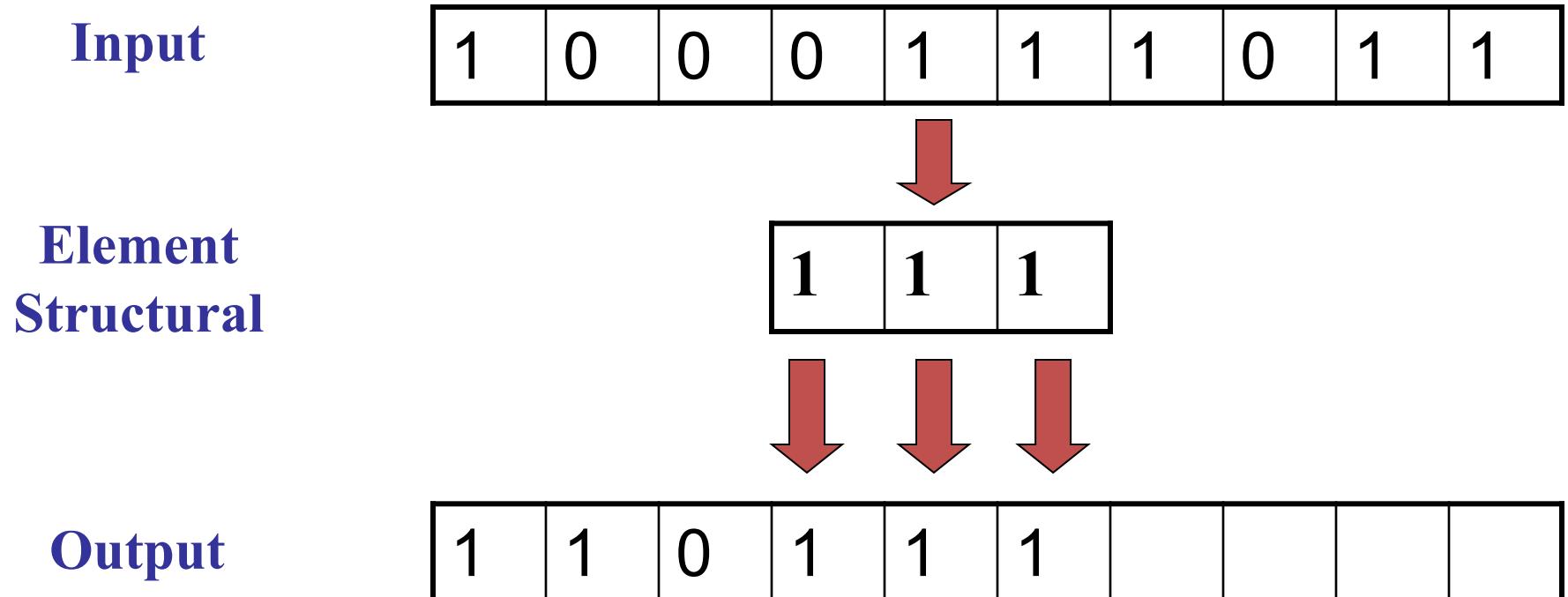
# Dilatare – exemplu



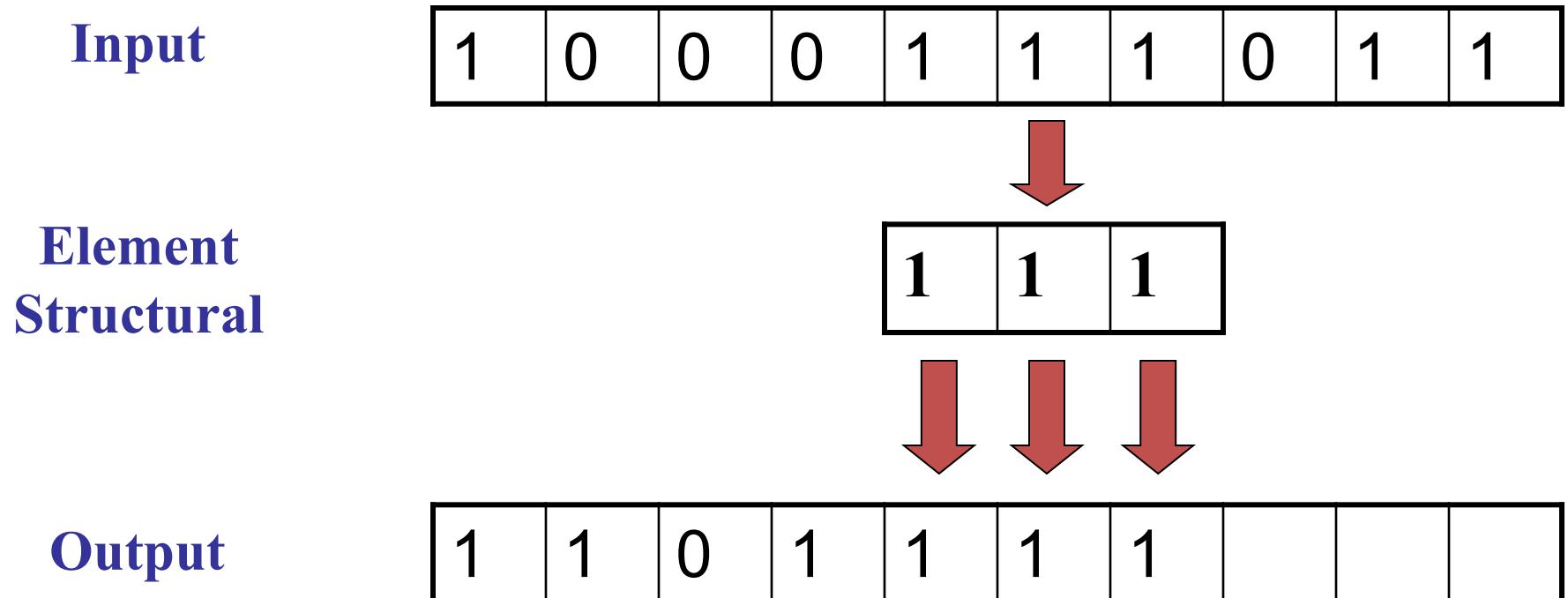
# Dilatare – exemplu



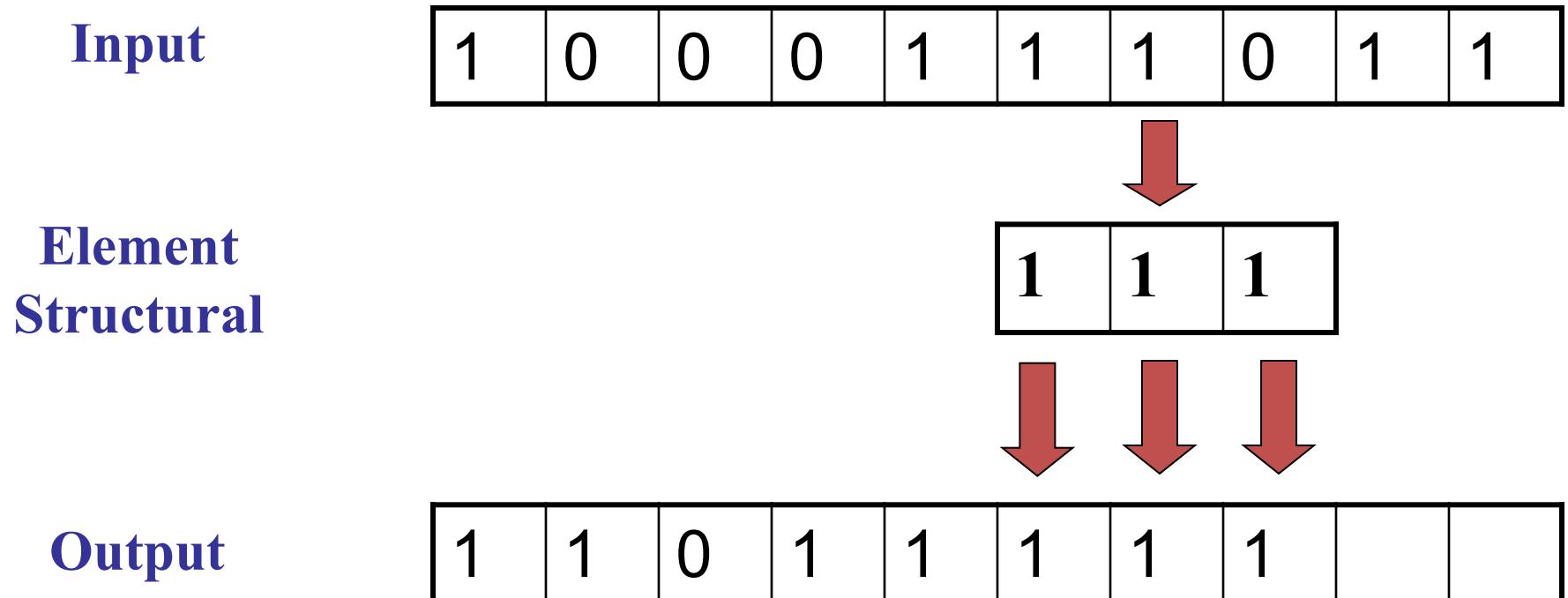
# Dilatare – exemplu



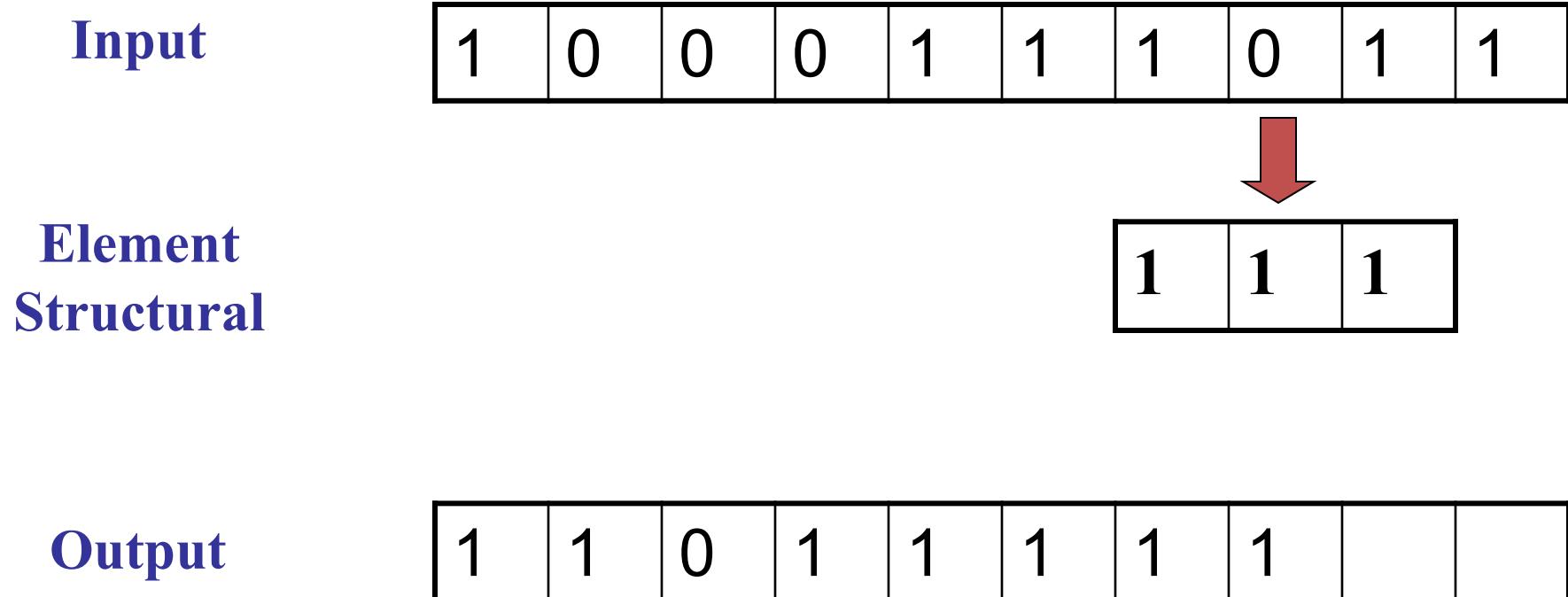
# Dilatare – exemplu



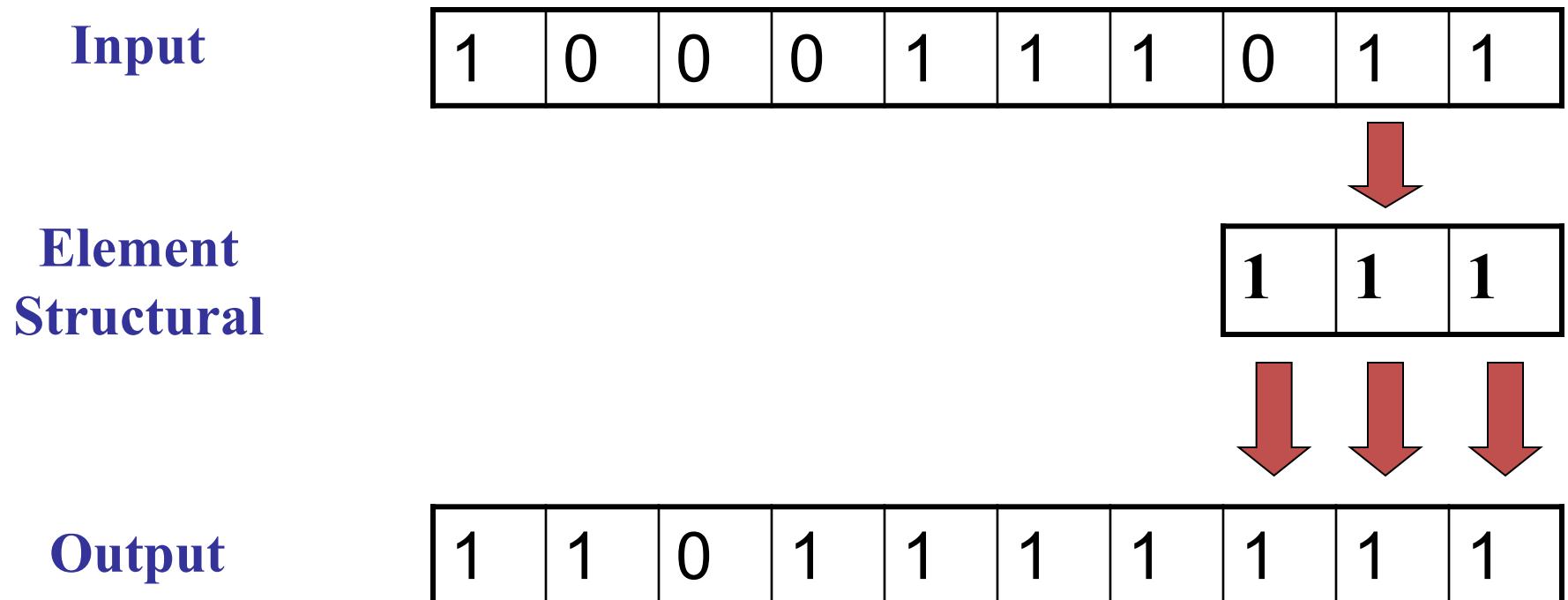
# Dilatare – exemplu



# Dilatare – exemplu



# Dilatare – exemplu



Foreground-ul devine mai mare, găurile sunt umplute.

# Dilatare vs. Eroziune

Pentru fiecare pixel:

- **Dilatare:** dacă originea elementului structural se suprapune peste un pixel alb/foreground din imagine, atunci toți pixelii acoperiți de valori 1 din elementul structural devin albi/foreground.

# Dilatare – exemplu (2D)

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

Imagine binară **B**

1	1	1
1	1	1
1	1	1

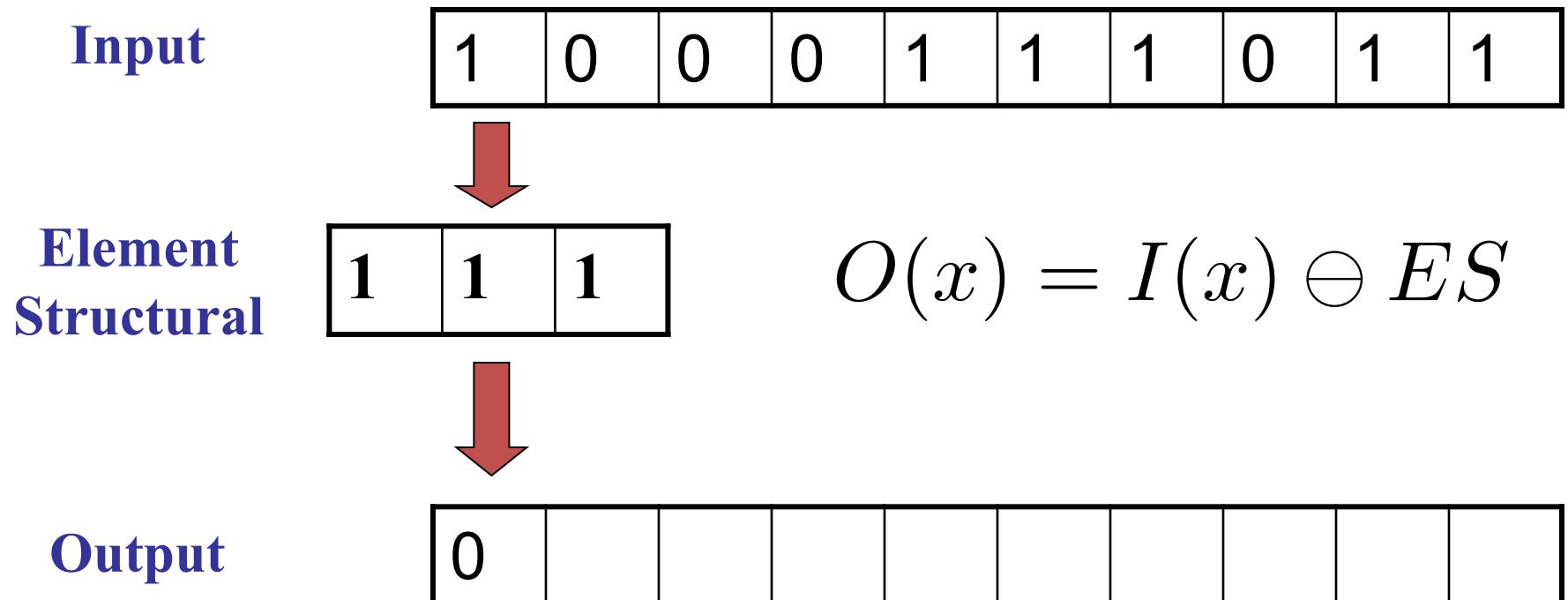
Element structural **S**

# Dilatare vs. Eroziune

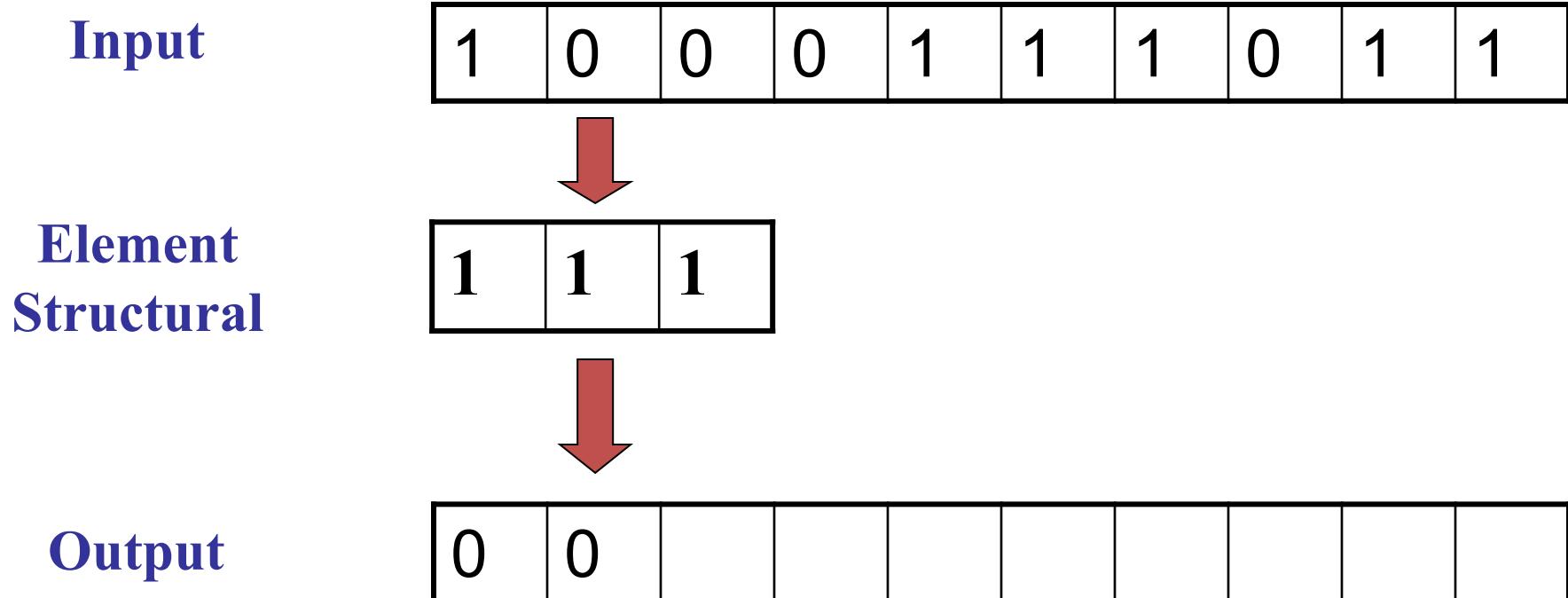
Pentru fiecare pixel:

- **Dilatare:** dacă originea elementului structural coincide cu un pixel alb/foreground din imagine , atunci toți pixelii acoperiți de valori 1 din elementul structural devin albi/foreground
- **Eroziune:** dacă originea elementului structural se suprapune peste un pixel inițial alb/foreground din imagine, și există un pixel alb/foreground din elementul structural care se suprapune peste un pixel negru/background din imagine atunci pixelul inițial devine negru/background

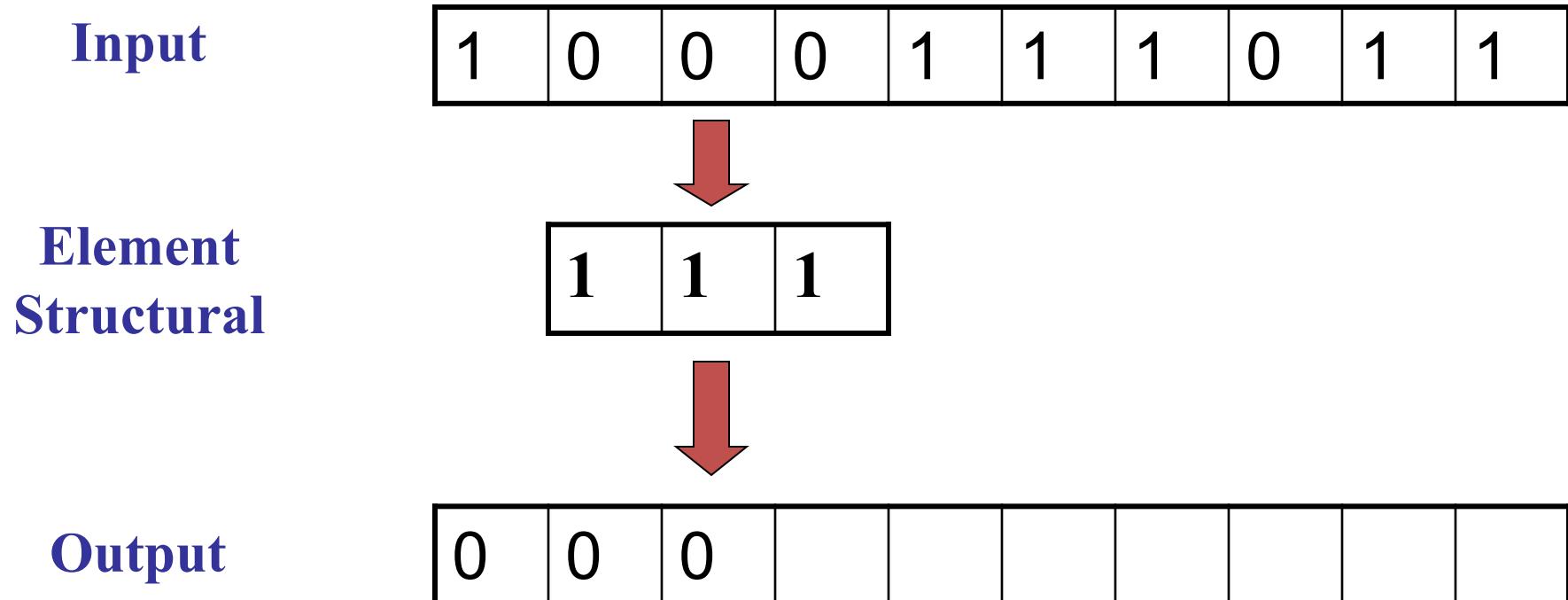
# Eroziune – exemplu (1D)



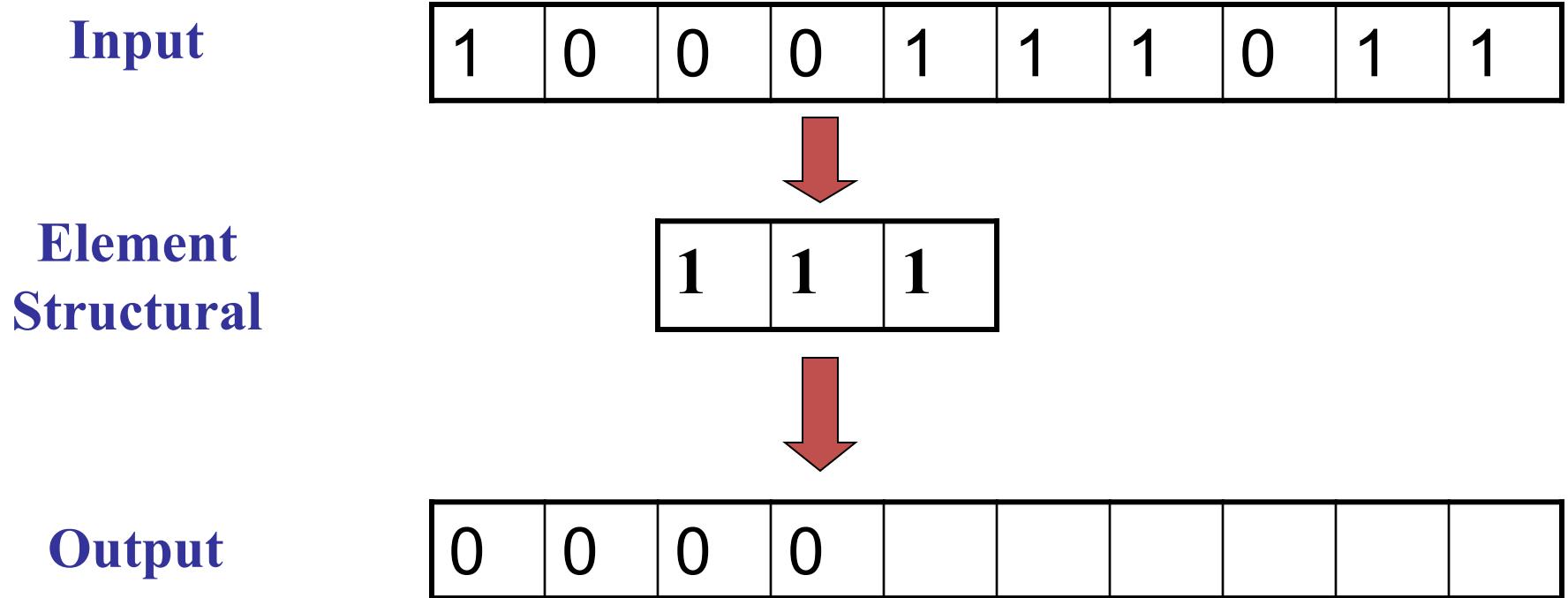
# Eroziune – exemplu



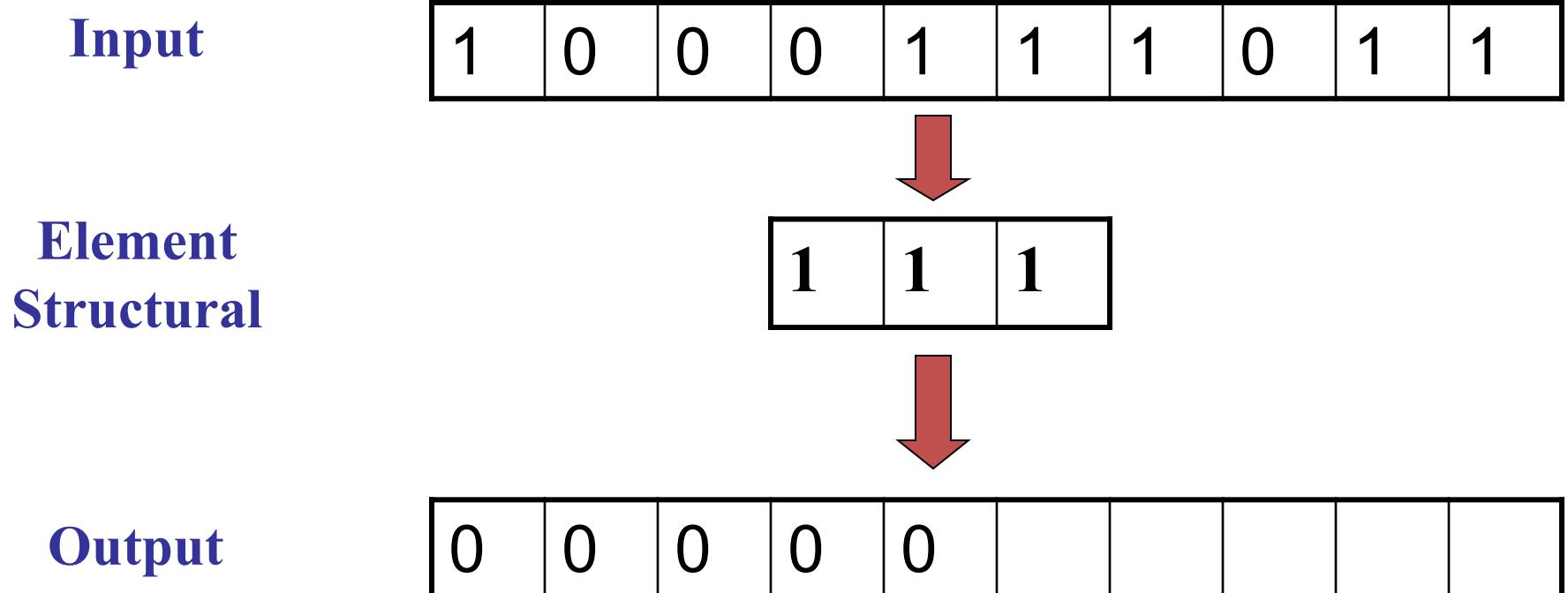
# Eroziune – exemplu



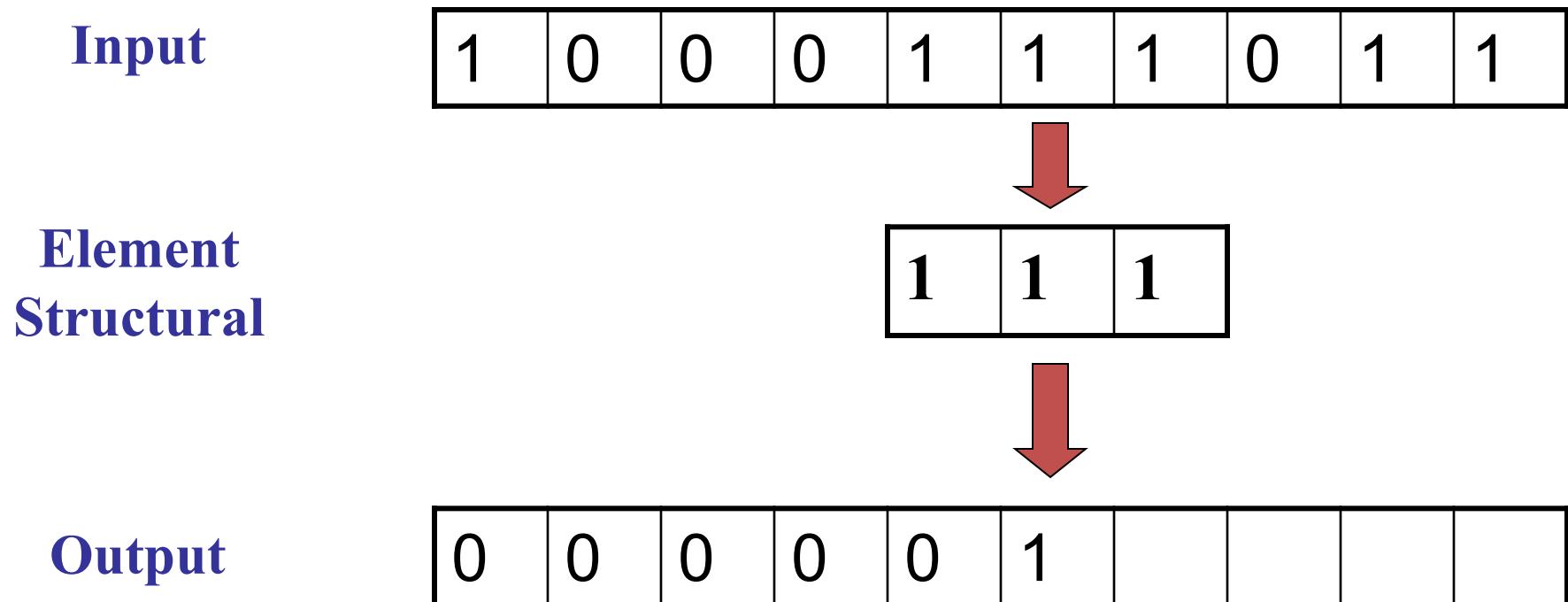
# Eroziune – exemplu



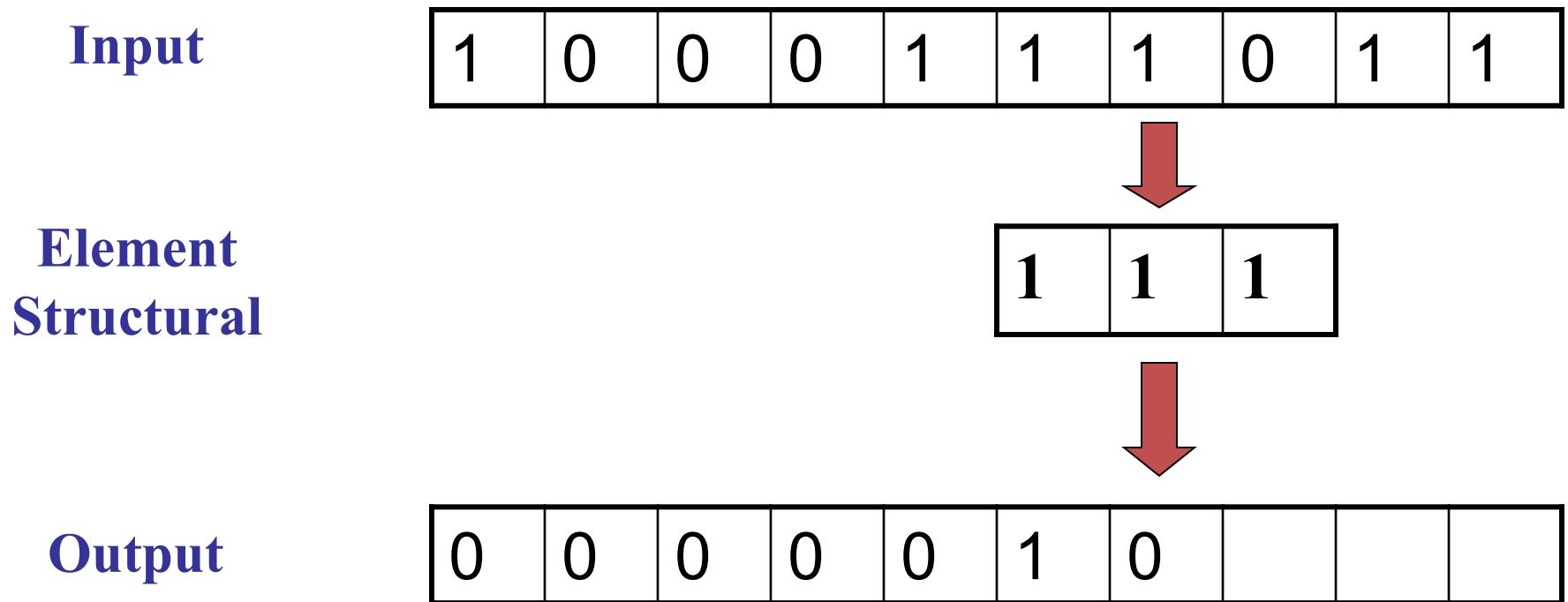
# Eroziune – exemplu



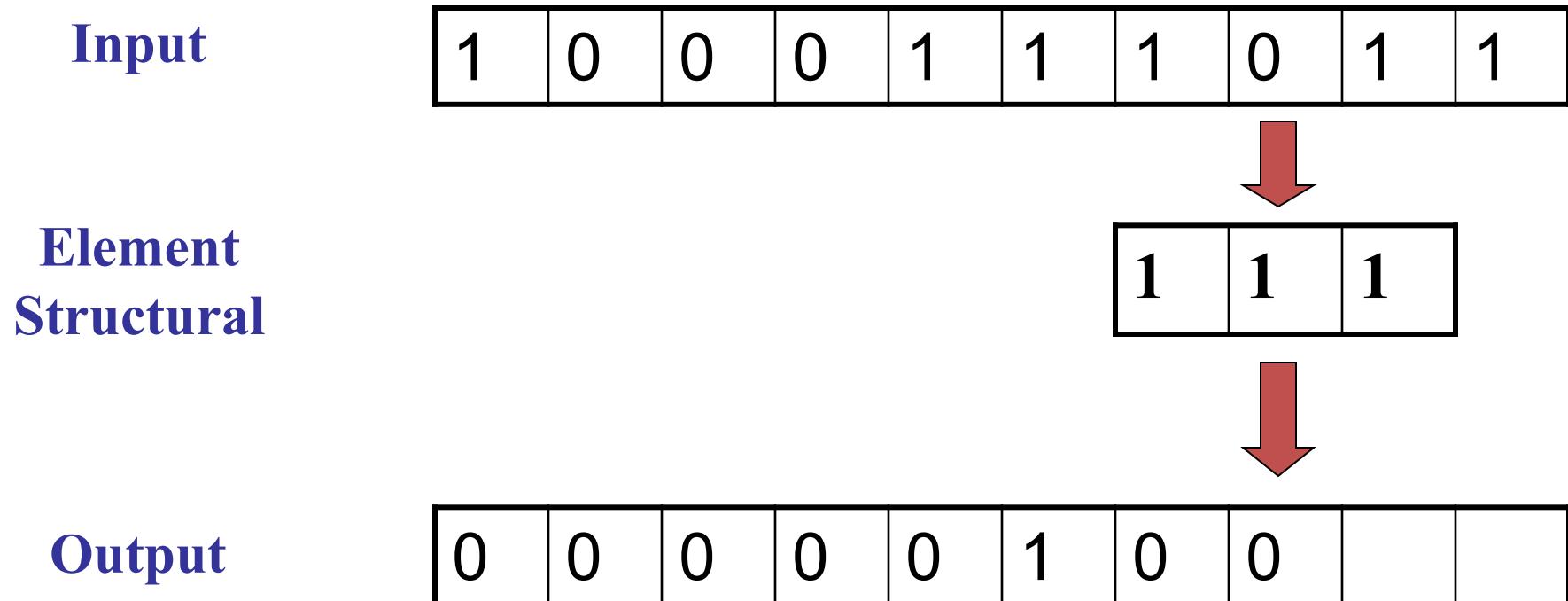
# Eroziune – exemplu



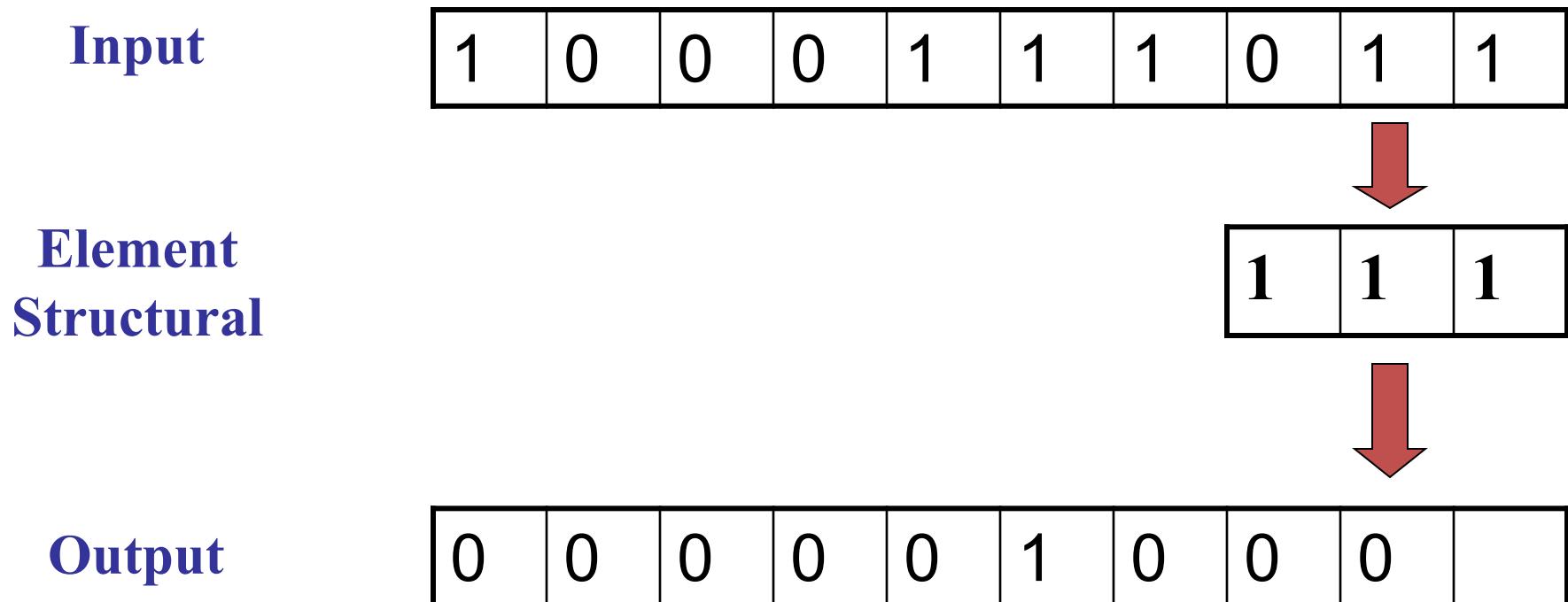
# Eroziune – exemplu



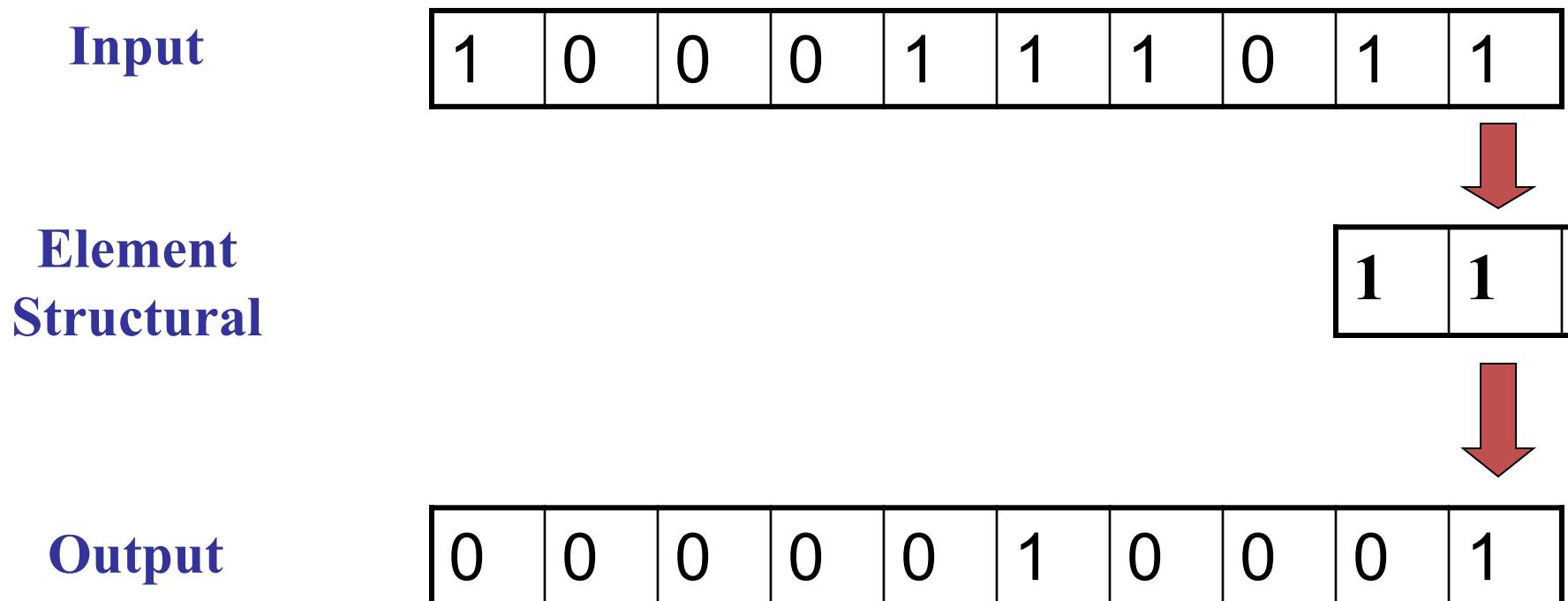
# Eroziune – exemplu



# Eroziune – exemplu



# Eroziune – exemplu



Foreground-ul se micșorează.

# Eroziune – exemplu (2D)

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

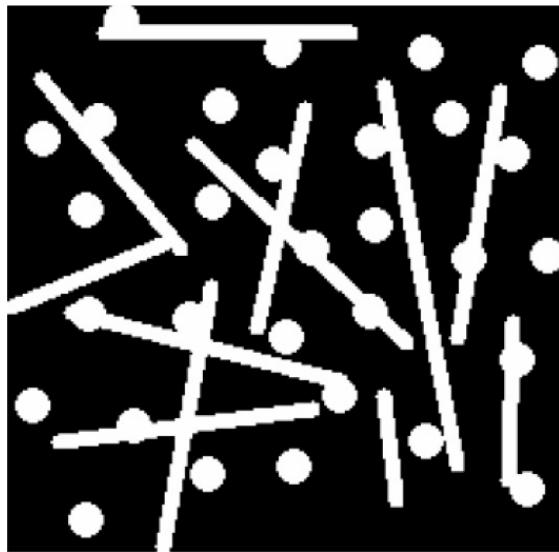
Imagine binară **B**

1	1	1
1	1	1
1	1	1

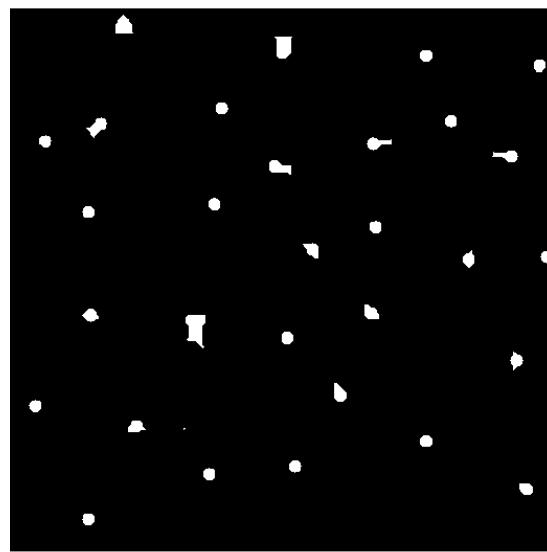
Element structural **S**

# Deschidere = eroziune + dilatare

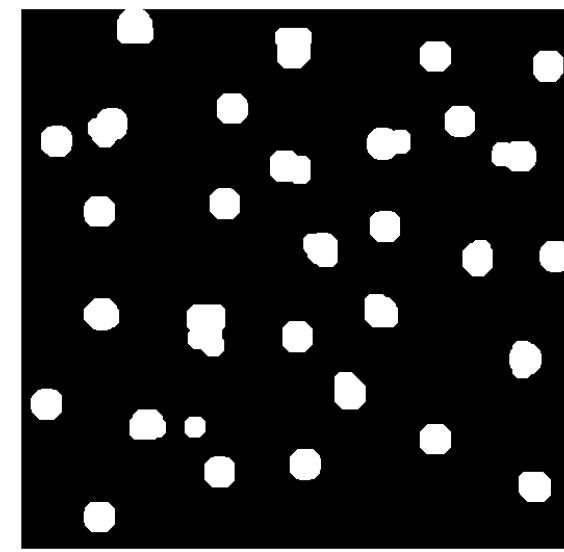
- Eroziune, apoi dilatare
- Elimină obiecte mici, păstrează forma originală



Imagine inițială



Eroziune



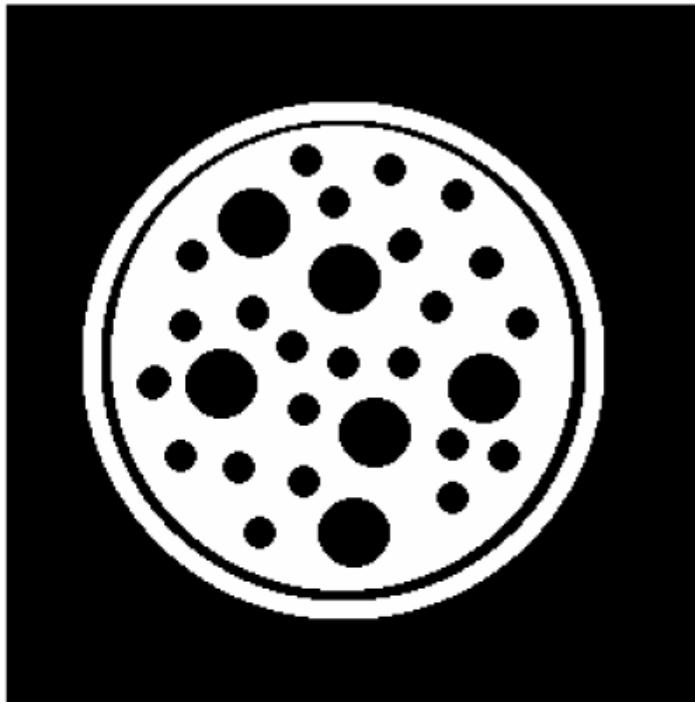
Eroziune + Dilatare

Element de structură



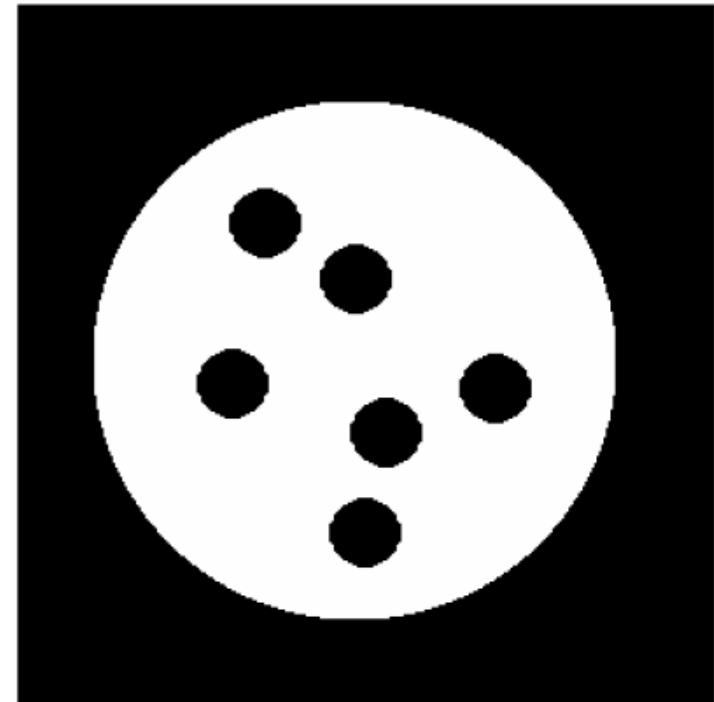
# $\hat{\text{I}}$ nchidere = dilatare + eroziune

- Dilatare, apoi eroziune
- Umple găurile, păstrează forma originală



**Înainte de Închidere**

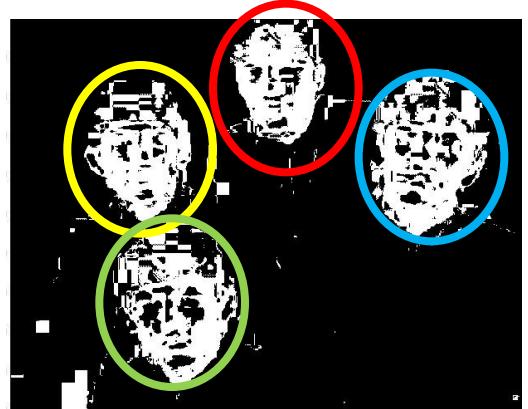
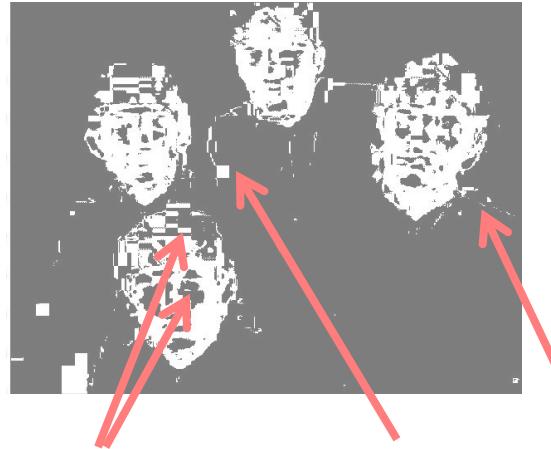
Element de structură



**După Închidere**

# Probleme

- Ce facem cu imaginile binare rezultate ca cea din dreapta?
  - găuri
  - fragmente foarte mici, izolate
- Cum delimităm regiunile de interes?
  - numărăm obiecte
  - calculăm diverse proprietăți pentru un obiect



# Componente conexe

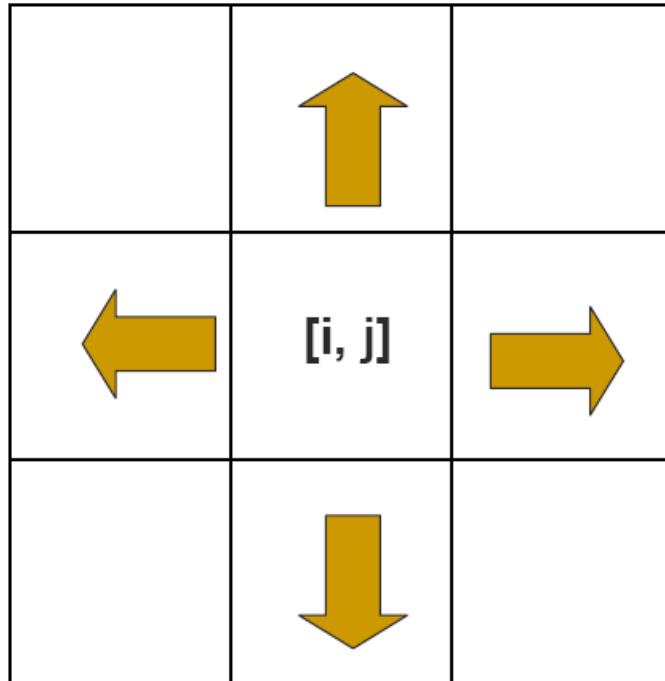
- Identifică regiuni distințe de pixeli “conectați”

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

Imagine binară

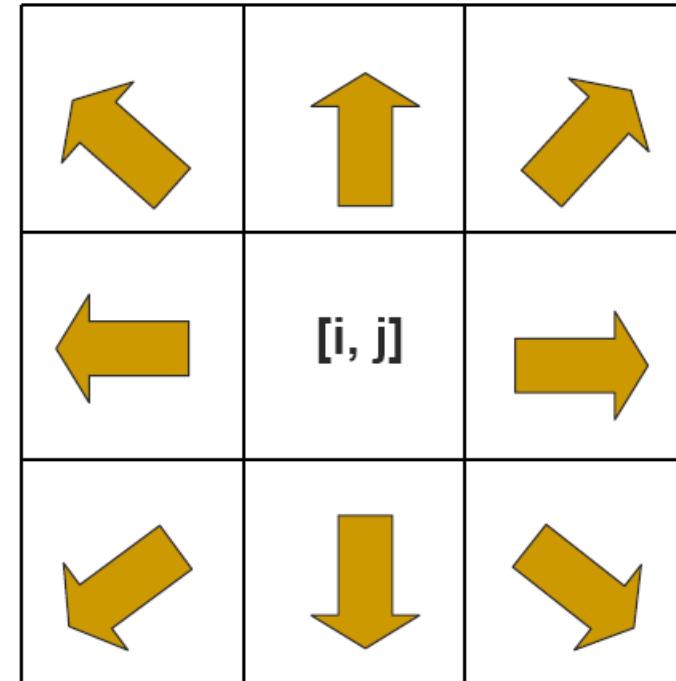
# Conecțivitate

- Definește ce pixeli sunt considerați vecini (“conectați”)



**conectivitate – 4**

(4 vecini)

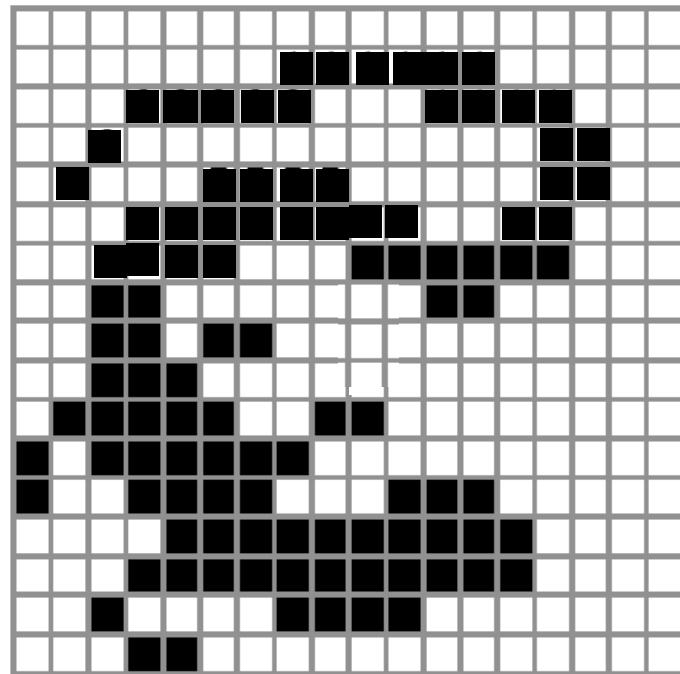


**conectivitate – 8**

(8 vecini)

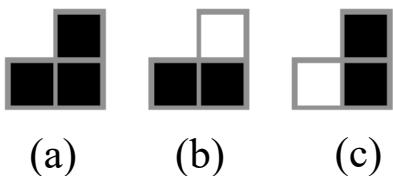
# Algoritm pentru determinarea componentelor conexe

- Considerăm un algoritm secvențial liniar în numărul de pixeli ai imaginii (2 treceri).
- **Input:** imagine binară
- **Output:** imagine “etichetată”, fiecare pixel are asignat un număr al componentei conexe
- Foreground-ul este reprezentat în acest exemplu de pixeli negri
- Conectivitate - 4

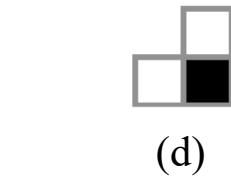


# Algoritm pentru determinarea componentelor conexe

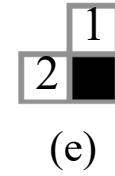
- Etichetăm un pixel foreground pe baza valorilor vecinilor săi
- Vecinii - dați de conectivitate (aici – 4)
- Procesăm pixelii din imagine secvențial de la stânga la dreapta, de sus în jos
- Comparăm pixelul curent cu vecinul din stânga și cel de sus



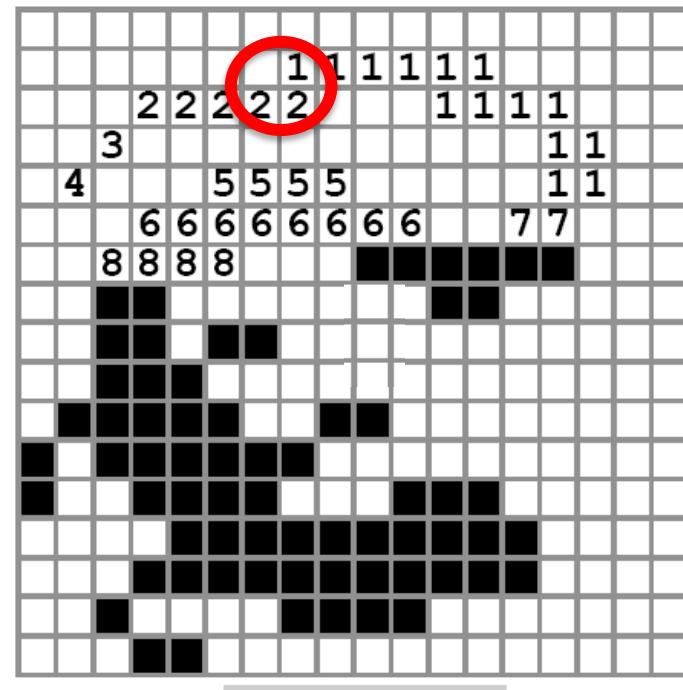
aceeași componentă  
conexă – preia eticheta



altă componentă  
conexă – adaugă etichetă



aceeași componentă  
conexă – etichete diferite  
Cum procedăm?

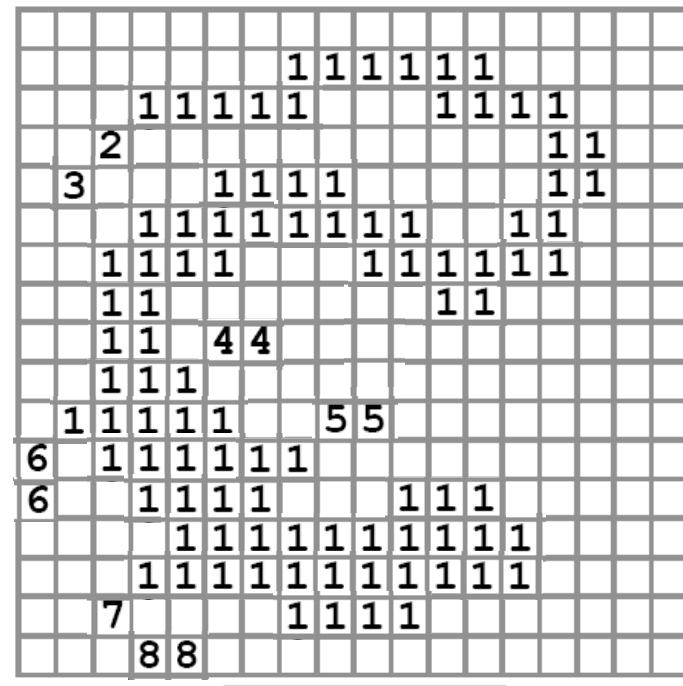


Tabel de echivalență  
 $\{1, 2\}, 7\}$   
 $\{3\}$   
 $\{4\}$   
 $\{5, 6, 8\}$

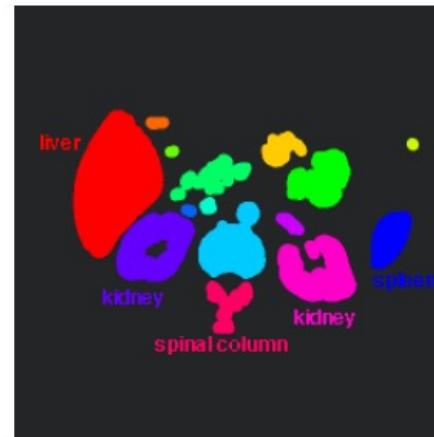
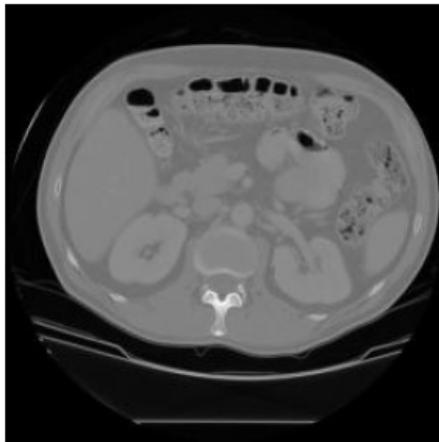
Slide adaptat după J. Neira

# Algoritm pentru determinarea componentelor conexe

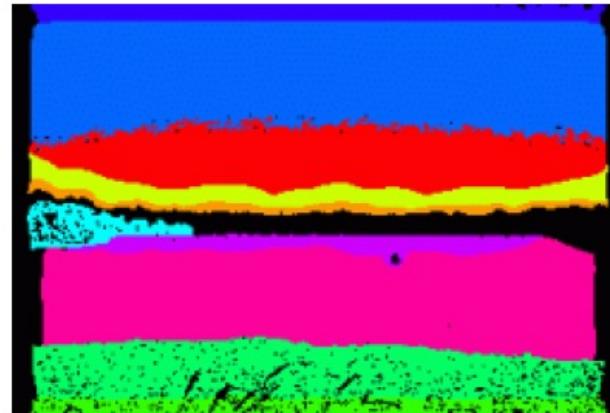
- Etichetăm un pixel foreground pe baza valorilor vecinilor săi
- Vecinii - dați de conectivitate (aici – 4)
- Procesăm pixelii din imagine secvențial de la stânga la dreapta, de sus în jos
- Comparăm pixelul curent cu vecinul din stânga și cel de sus
- Pe baza tabelului de echivalență renumerotăm componentele



# Componente conexe - exemple



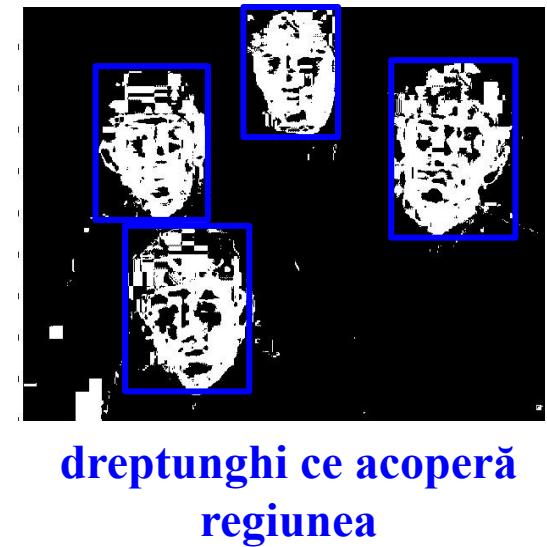
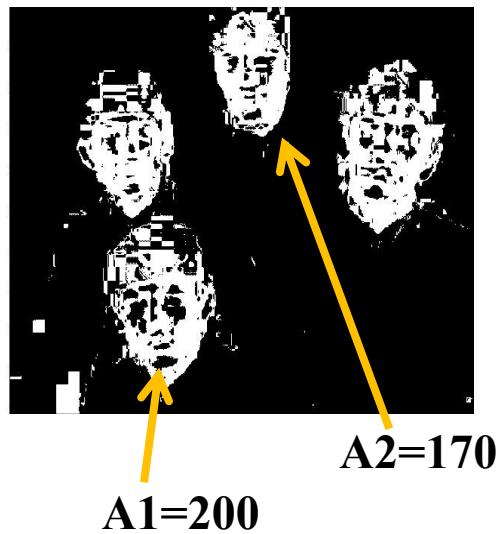
componente  
conexe obținute  
în urma aplicării  
unei prag



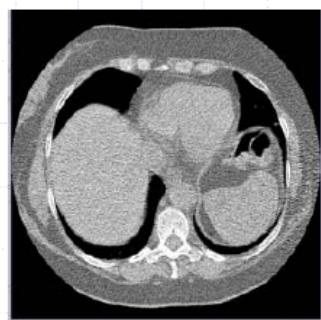
componente  
conexe obținute  
în urma grupării  
pixelilor în  
clustere

# Proprietăți ale regiunilor

- Fiind date componente conexe, putem calcula proprietăți pentru caracterizarea lor:
  - aria (numărul de pixeli în regiune)
  - centroidul (media coordonatele x and y pentru pixelii din regiune)
  - fereastra dreptunghiulară ce acoperă regiunea (pe baza coordonatelor minime si maxime x,y)



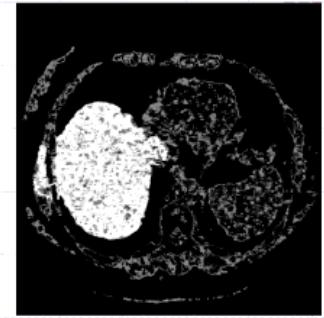
# Analiza imaginilor binare - exemplu segmentare



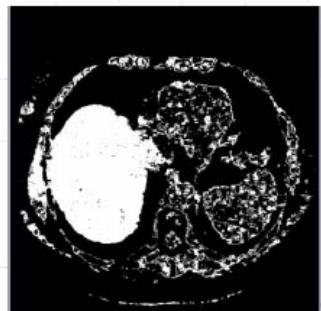
*Aplicare prag*



*Extrage regiunea cea mai mare*



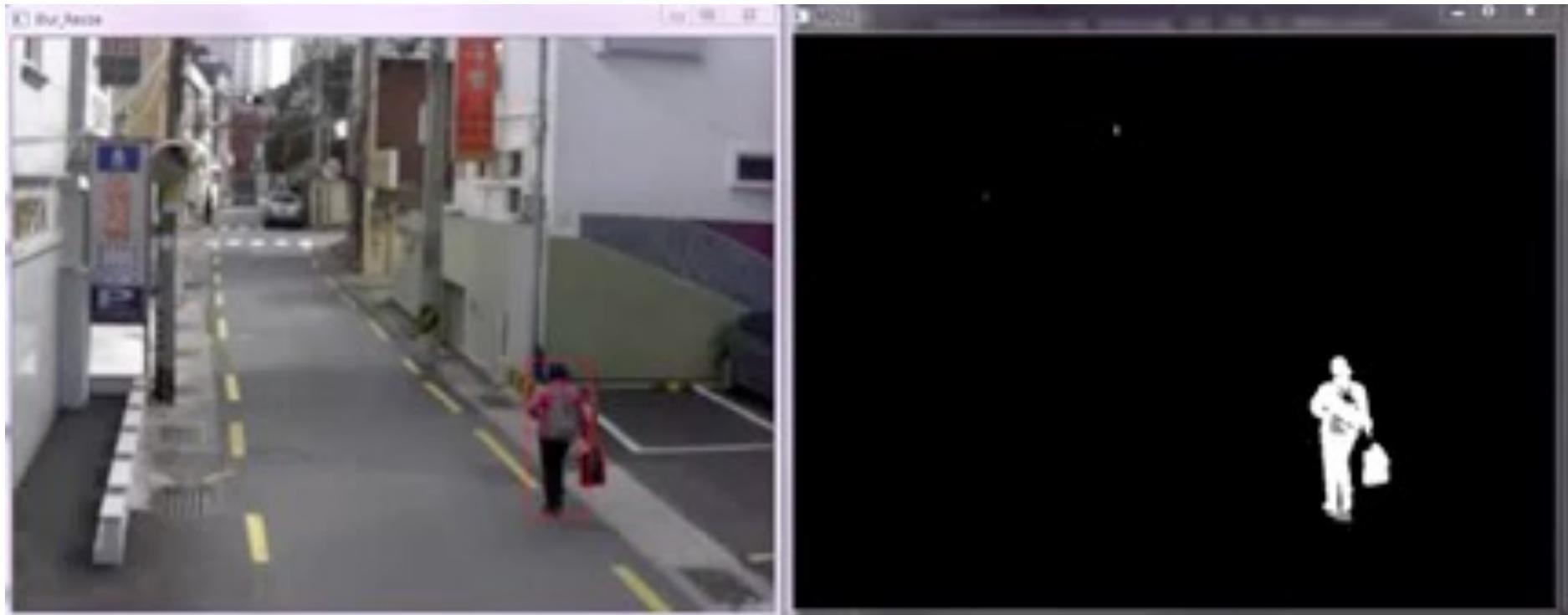
*umple golurile*



*Extrage regiunea cea mai densă*



# Analiza imaginilor binare - exemplu extragerea background-ului, detectare regiuni

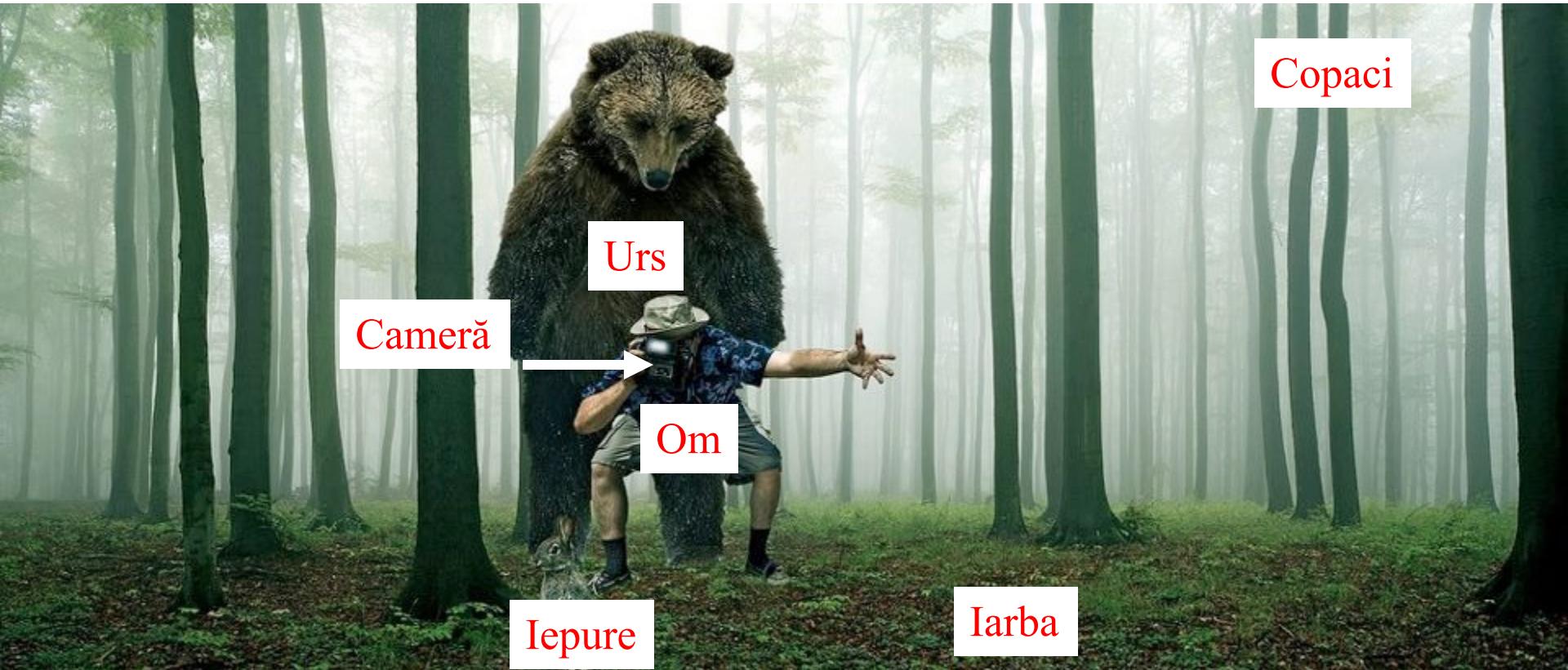


# Imagini binare

- Avantaje
  - foarte rapid de procesat
  - necesită memorie mică
  - putem obține descriptori compacti pentru forme
- Dezavantaje
  - greu să obținem contururi perfecte (siluete)
  - zgomot ce apare în imagini reale
  - pierdem prea multă informație prin binarizare

# Recunoașterea claselor de obiecte

# Ce vedeți în această imagine?



Pădure



Este **periculos**?

Este **viu**?

Cât de **repede** aleargă?

Are **coadă**?

# Recunoașterea vizuală a claselor de obiecte



**Clasificare:** este vreun felinar în imagine?

**Da/Nu**



**Detectare:** unde este felinarul?

**Localizare**

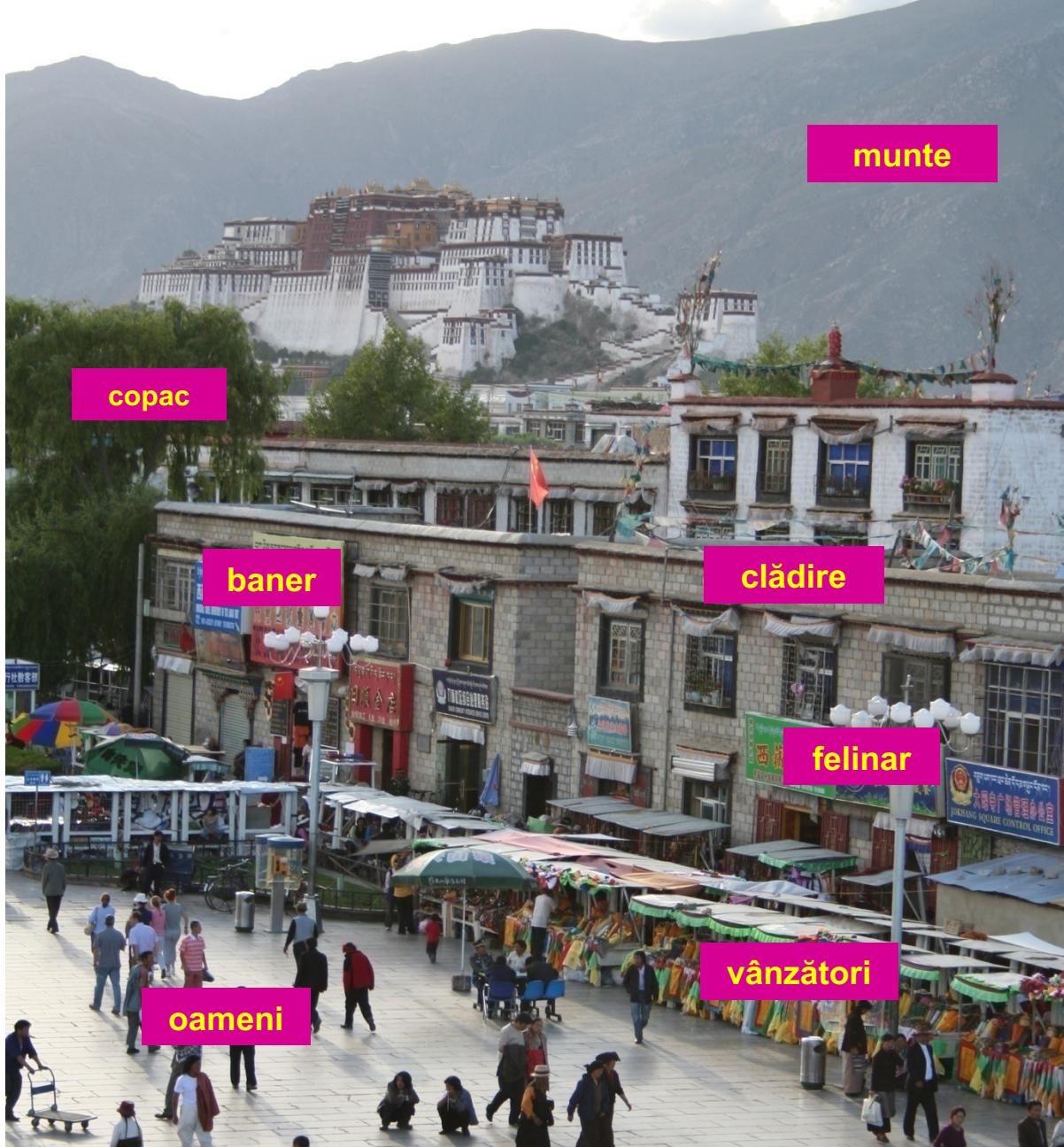


# Recunoaștere: care palat?

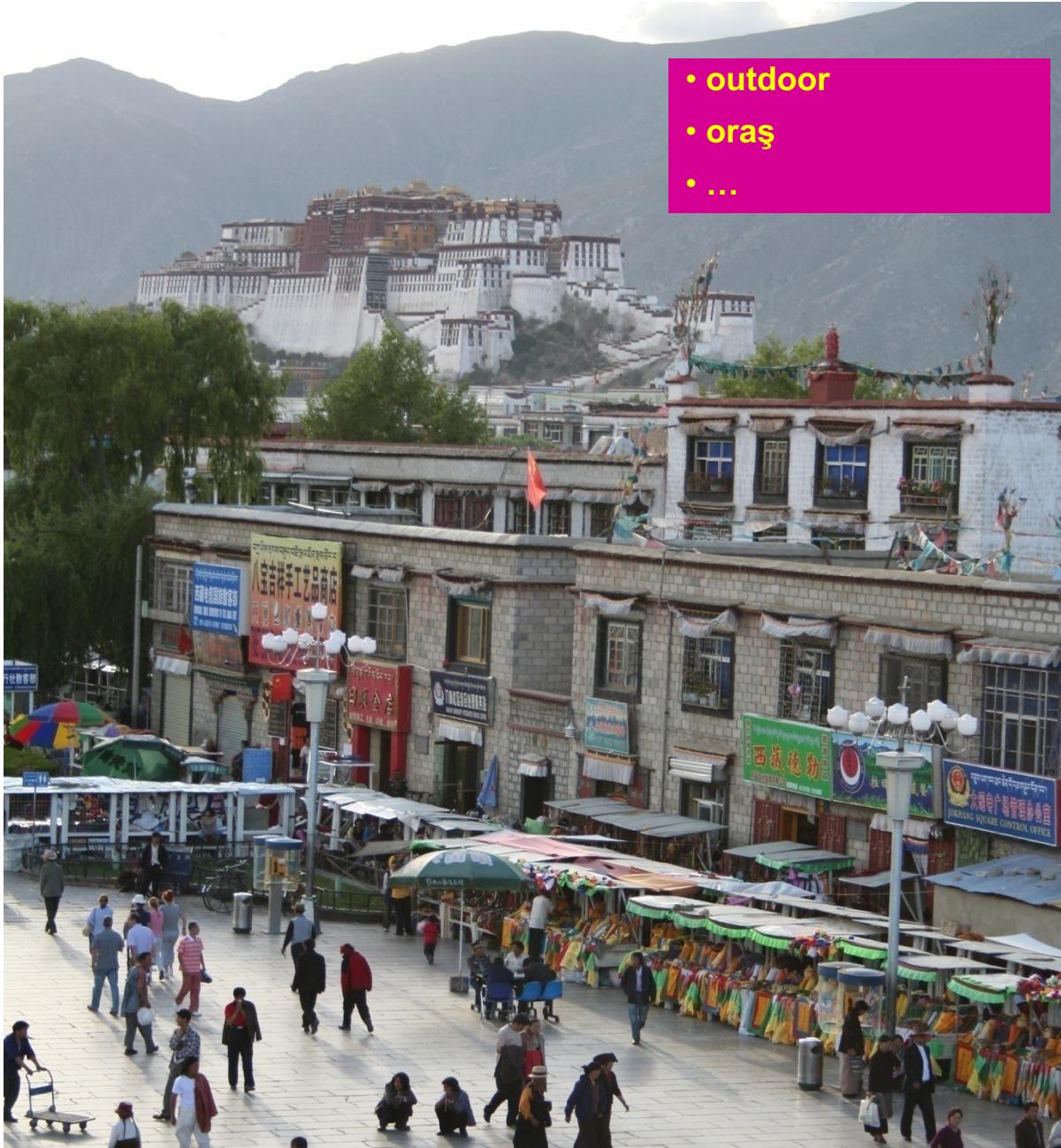
Instantieri



# Clasificarea regiunilor din imagine



# Clasificarea întregii scene



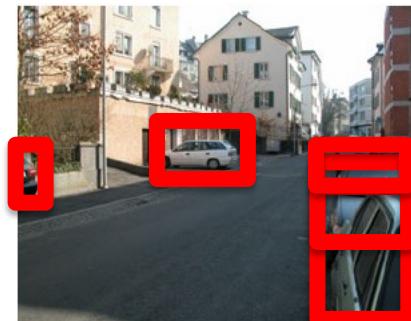
- outdoor
- oraș
- ...

# Recunoașterea claselor de obiecte la nivel de instanță



mașina lui Mihai

# Recunoașterea claselor de obiecte la nivel generic



Vrem să recunoștem toate instanțierile unei clase de obiecte

# Recunoașterea claselor de obiecte

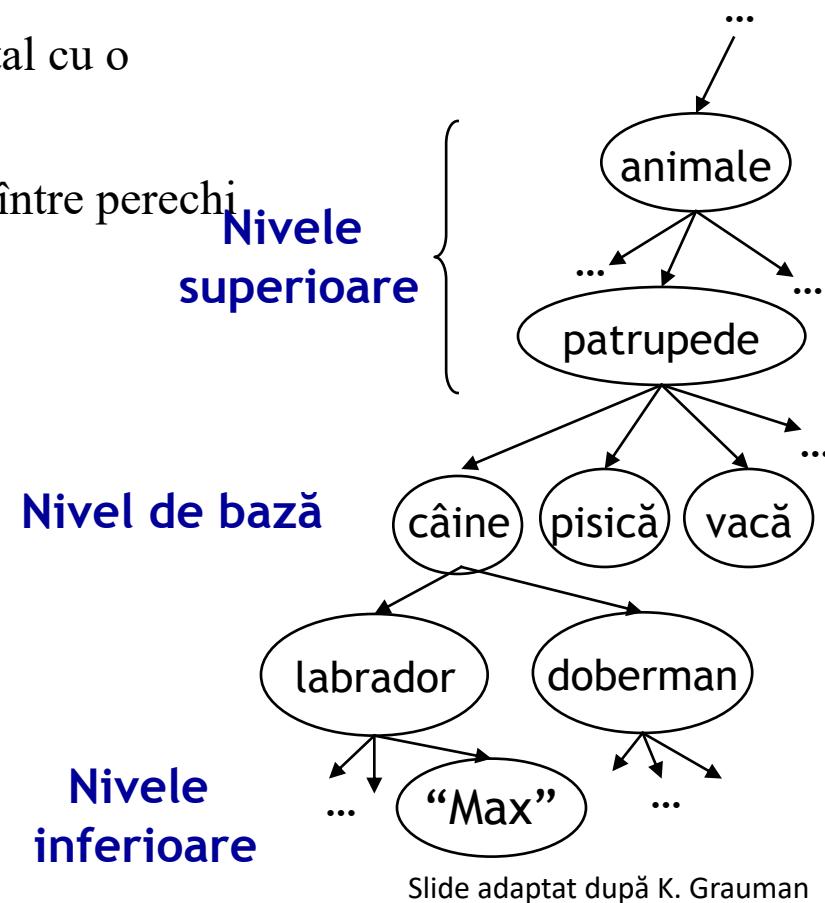
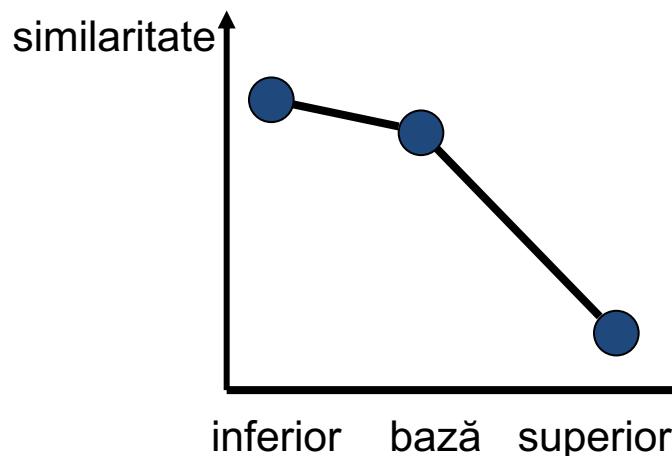
- Descrierea problemei:
  - “Fiind dat un număr mic de imagini de antrenare ale unei clase de obiecte, recunoașteți instanțieri ale clasei de obiecte asignând eticheta corespunzătoare clasei de obiecte.”
- Pentru care clasă de obiecte puteți asocia repede o imagine?



# Organizarea ierarhică a claselor de obiecte (Rosch, 1976)

Nivelul de bază:

- clasele de obiecte la nivel de bază sunt clasele de nivel cel mai înalt pentru care membrii lor au forme percepute similare.
- nivelul cel mai înalt pentru care reprezentăm mental cu o singură imagine întreaga clasă de obiecte
- există un număr seminificativ de atrbute comune între perechi de membri
- primul nivel numit și înțeles de copii

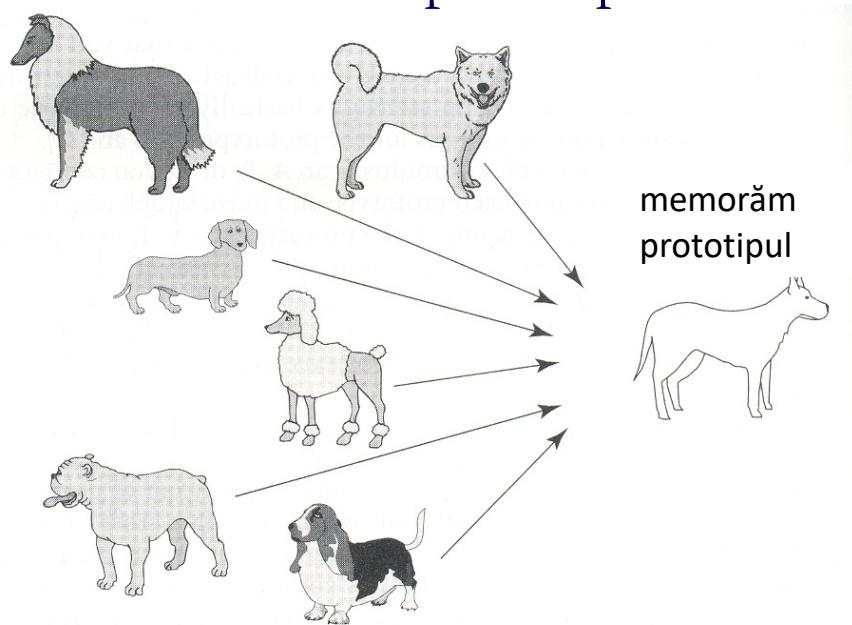


# Organizarea ierarhică a claselor de obiecte (Rosch, 1976)

- Rosch a găsit următoarele:
  - oamenii clasifică clasele de obiecte mult mai repede la nivel de bază
  - tendință de a clasifica după nivelul de bază ('câine') înainte de nivelul superior ('animal') sau nivelul inferior ("golden retriever")
  - mai întâi clasificare la nivel de bază iar apoi recunoaștere
  - acord la nivel de bază (de cele mai multe ori)
    - desktop, laptop, tablete, pc, mac – nivel de bază
    - calculator – nivel de bază
      - desktop, laptop, tablete, pc, mac – nivel inferior
      - ține de nivelul de expertiză

# Recunoașterea claselor de obiecte

## Modelul prototip



Realizăm clasificarea într-o clasă de obiecte prin compararea unui nou exemplu cu prototipul

# Clase de obiecte - exemple



# Cât de multe clase de obiecte există?



# Recunoașterea claselor de obiecte în Vederea Artificială - ce funcționează bine

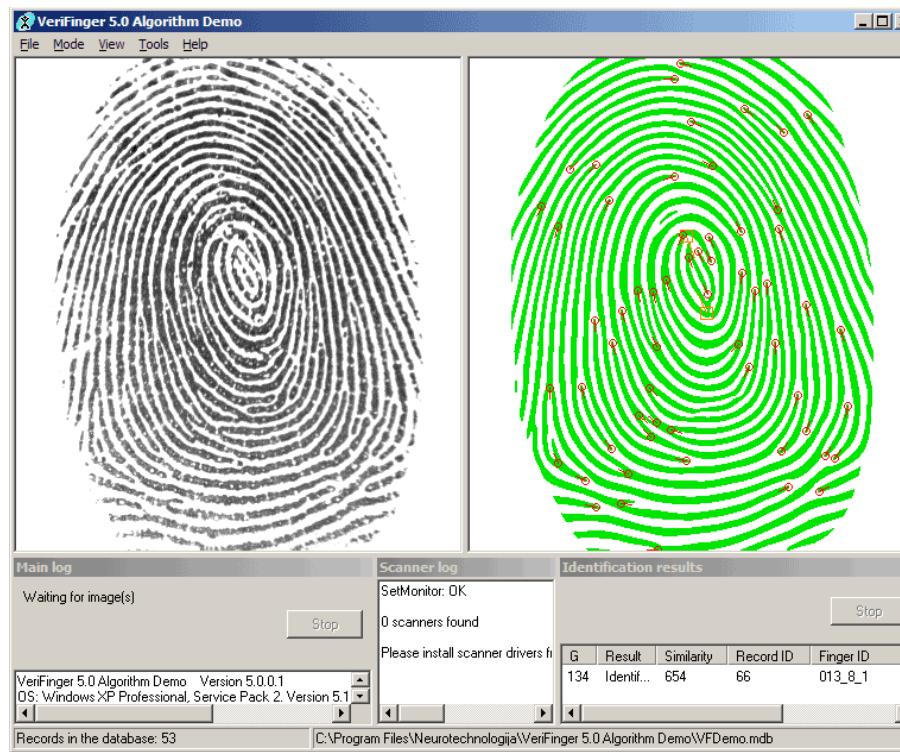
- OCR – plăcuțe de înmatriculare, coduri poștale, etc.

A grid of handwritten digits arranged in 10 rows and 9 columns. The digits are written in a cursive style. Some digits are bolded or underlined to serve as reference points for character recognition.

3	6	8	/	7	9	6	6	9	1
6	7	5	7	8	6	3	4	8	5
2	1	7	9	7	1	2	8	4	6
4	8	1	9	0	1	8	8	9	4
7	6	1	8	6	4	1	5	6	0
7	5	9	2	6	5	8	1	9	7
2	2	2	2	2	3	4	4	8	0
0	2	3	8	0	7	3	8	5	7
0	1	4	6	4	6	0	2	4	3
7	1	2	8	1	6	9	8	6	1

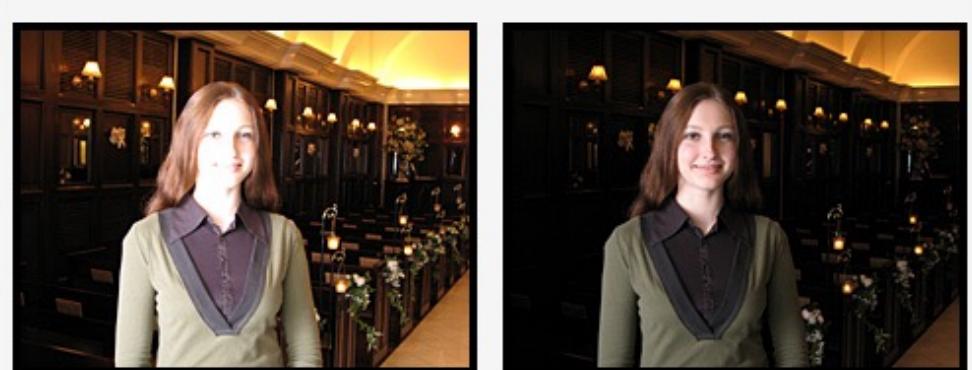
# Recunoașterea claselor de obiecte în Vederea Artificială - ce funcționează bine

- OCR – plăcuțe de înmatriculare, coduri poștale, etc.
- Recunoașterea amprentei



# Recunoașterea claselor de obiecte în Vederea Artificială - ce funcționează bine

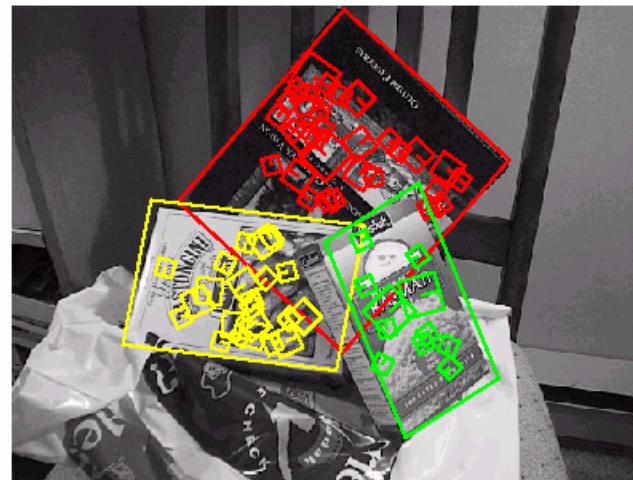
- OCR – plăcuțe de înmatriculare, coduri poștale, etc.
- Recunoașterea amprentei
- Detectarea facială



[Face priority AE] When a bright part of the face is too bright

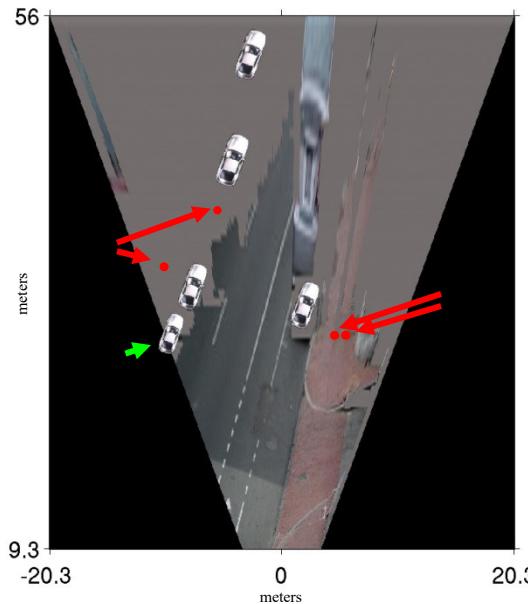
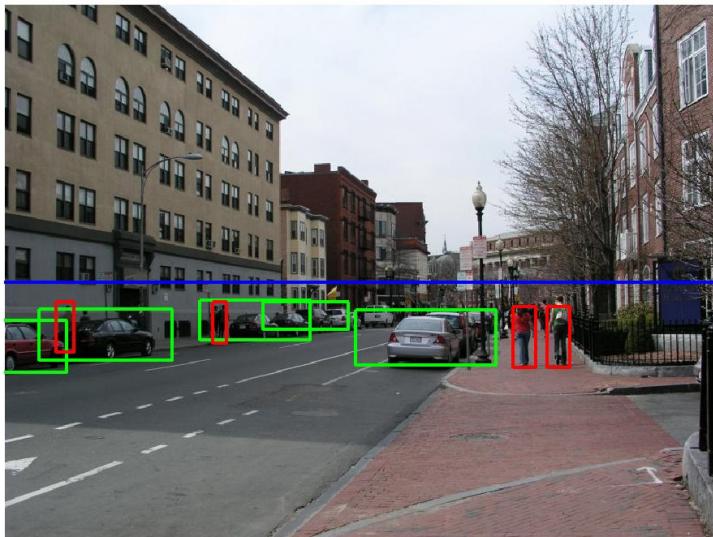
# Recunoașterea claselor de obiecte în Vederea Artificială - ce funcționează bine

- OCR – plăcuțe de înmatriculare, coduri poștale, etc.
- Recunoașterea amprentei
- Detectarea facială
- Recunoașterea obiectelor cu textură (coperți CD)

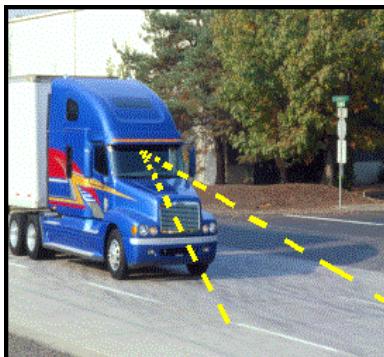


# Alte aplicații: conducere asistată

Detectarea mașinilor și a pietonilor



Detectarea liniilor de marcaj



# Găsirea obiectelor vizual similare

like visual shopping alpha

My Like List | NewsLetter | Blog

ALL SHOES BAGS WOMEN'S APPAREL MEN'S APPAREL KIDS ACCESSORIES JEWELRY & WATCHES HOLIDAY FOR THE HOME

IN Women's Shoes Search

Refine by Style Refine by Color Refine by Brand

Pumps Sandals Flats Par crimson taupe scarlet Clarks Sofft

Why is Like.com Different?  
Like is a visual shopping engine that lets you find items by color, shape and pattern.  
Click on Likeness Search to get started

All Products > Shoes > Women's Shoes > Cole Haan > Cole Haan - Carma OT Air Pump

Results 1 - 20 of 140,207

Sort By Likeness™ Price Change Your View: 1 2 3 4 5 6 7 NEXT >

**Natural Comfort - LV58**  
\$99.95  
  
Shop at Zappos.com  
Free Shipping Available  
Shop for more items like this:  
Likeness Search

**Cole Haan 'Carma Air' Patent Leather Open Toe Pump**  
\$275.00  
  
Shop at NORDSTROM.com  
Shop for more items like this:  
Likeness Search

**rsvp - Caitlyn**  
\$89.95  
  
Shop at Zappos.com  
Free Shipping Available  
Shop for more items like this:  
Likeness Search

Your Search Item  
Which part of the image do you like?  
Draw a box on the item to focus your search on that area.

Cole Haan - Carma OT Air Pump  
\$278.95  
More Details + Save to LikeList  
Shop at Zappos.com

Slide adaptat după K. Grauman

# Recunoaștere în Vederea Artificială

Dificultăți:

- Variabilitatea intra-clasă
- Clase de obiecte ce arată similar (mașină, camion, dubă)
- Variabilitate în mărime a obiectelor
- Confuzia cu backgroundul

