



Technical University of Cluj - Napoca  
Computer Science Department

# Procesarea Imaginilor

## Curs 12

Modele de culoare. Procesarea și segmentarea imaginilor color.



# Achiziția imaginilor color

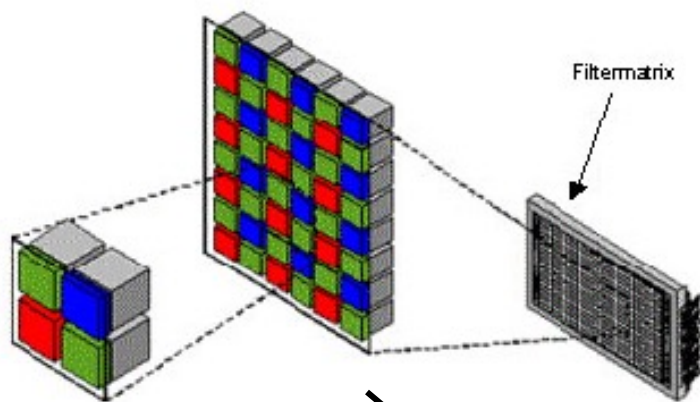
## Senzori color

<http://www.siliconimaging.com/RGB%20Bayer.htm>

[http://en.wikipedia.org/wiki/Three-CCD\\_camera](http://en.wikipedia.org/wiki/Three-CCD_camera)

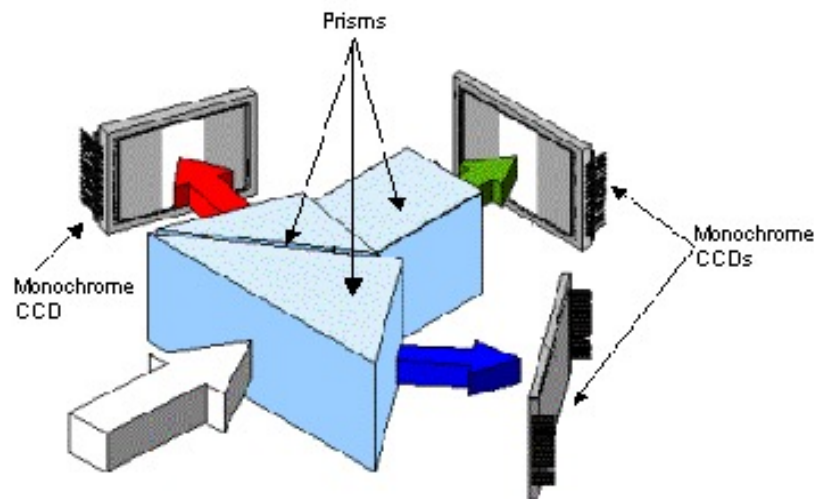
### a) Bayer mask

For color photos, the majority of commercial digital color cameras use pixels covered with special color filters in the three primary colors red, green and blue.



decodificarea  
pattern-ului Bayer

### c) 3-CCD camera



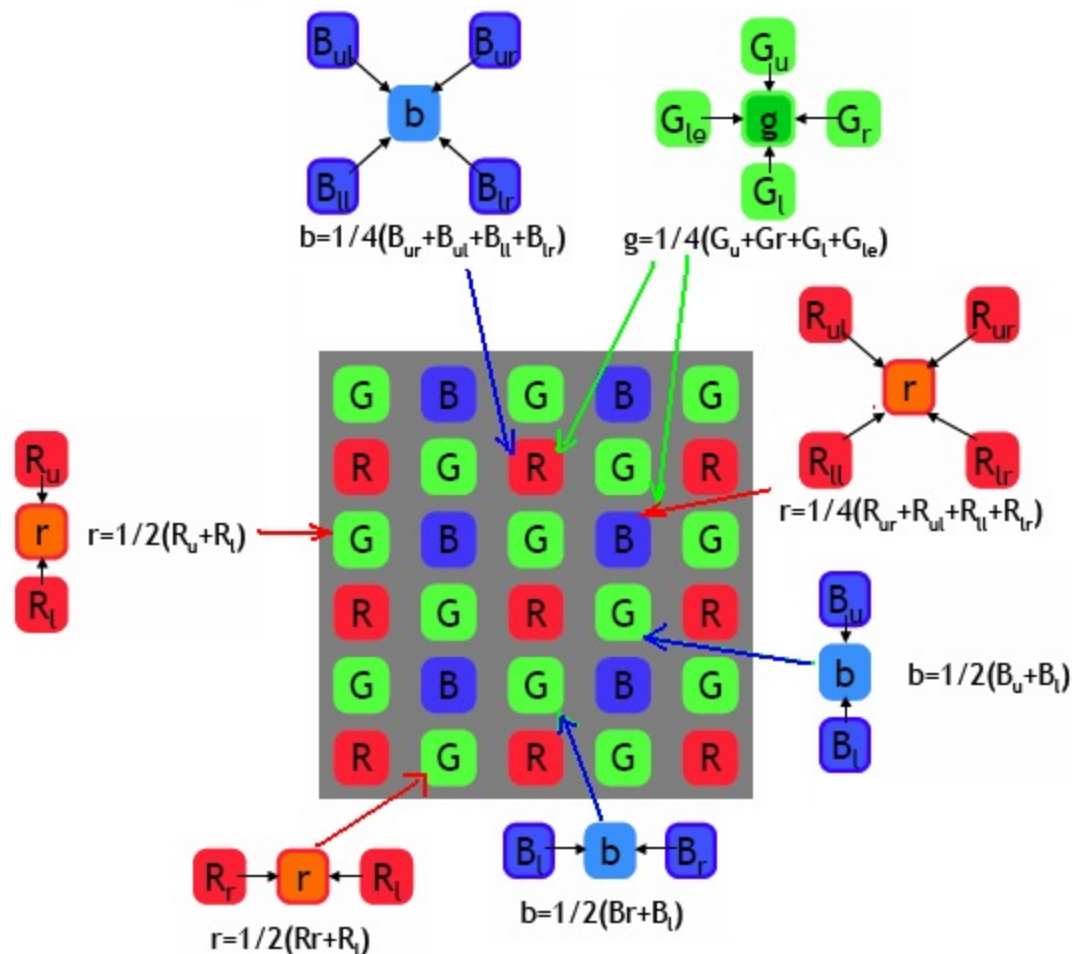
direct

Imagine RGB



# Achiziția imaginilor color

## Decodificarea pattern-ului Bayer



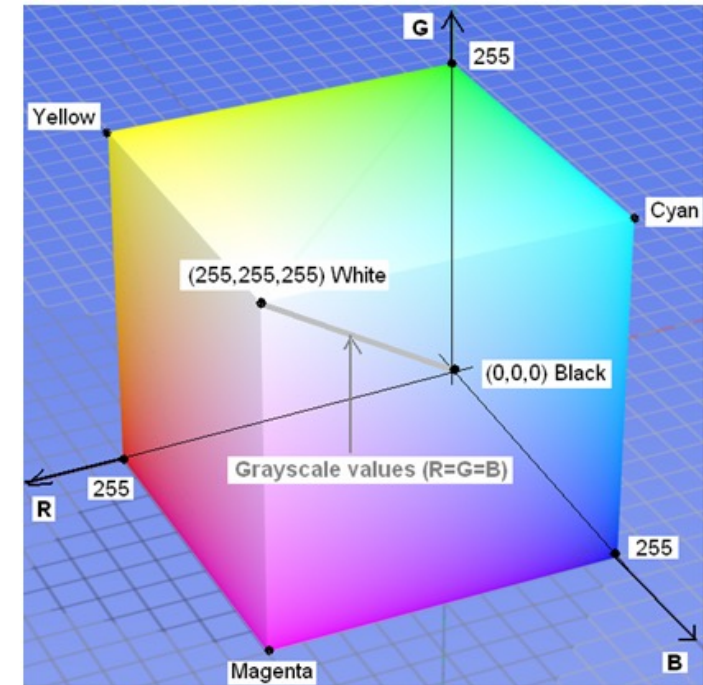
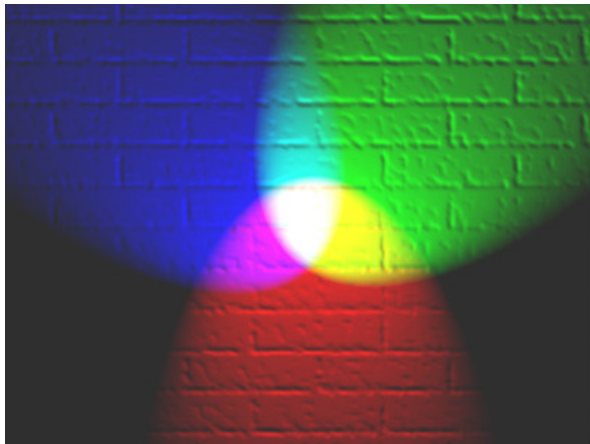
Calitatea imaginii (Bayer pattern vs. 3CCD) ???



# Spațiul de culoare RGB

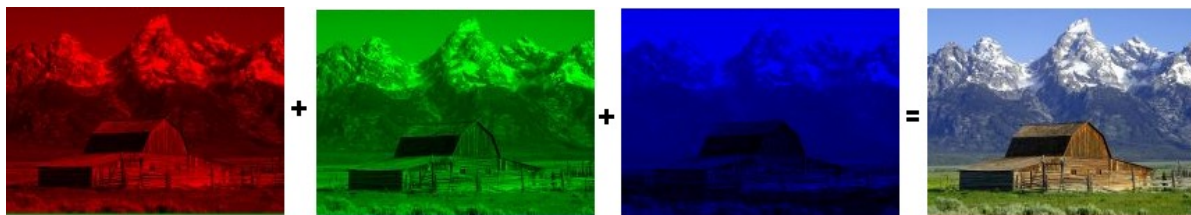
**RGB**  $\Rightarrow$  Culoarea fiecărui pixel (atât pentru echipamentele de achiziție – **camere** cât și pentru afișare - **TV, CRT, LCD**) se obține prin combinația a trei culori primare: **roșu**, **verde** și **albastru**. (**Red**, **Green** și **Blue**)

$\Rightarrow$  spațiu de culoare aditiv (**R+G+B**  $\Rightarrow$  Alb)



Modelul de culoare RGB mapat pe un cub.

În acest exemplu fiecare culoare este reprezentată pe câte 8 biți (256 de nivele) (imagini bitmap RGB24). Numărul total de culori este  $2^8 \times 2^8 \times 2^8 = 2^{24} = 16.777.216$ .







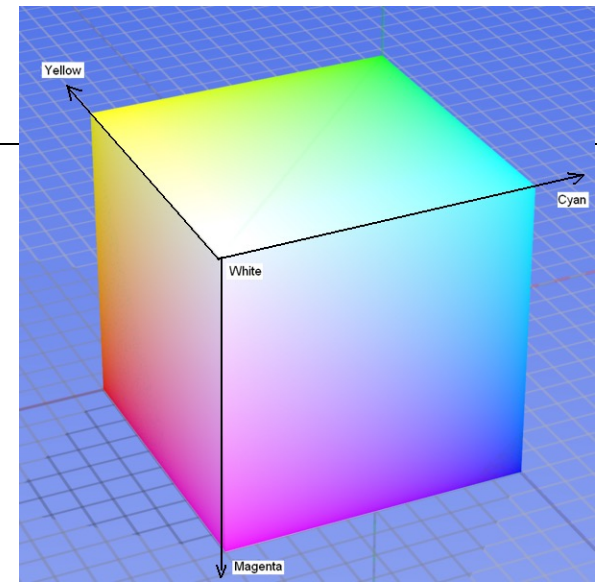
# Spațiul de culoare CMY

**CMY: spațiu de culoare complementar față de RGB, folosit la dispozitivele de imprimare color.**

**CMY: model substractiv**

**Alb = absenta componentelor de culoare**

**Negru = C + M + Y**



## CMYK





# Spațiul de culoare **RGB** normalizat

---

Reduce dependența de iluminare a culorii obiectului

Se poate aplica doar dacă variațiile de intensitate sunt uniforme de-a lungul spectrului RGB

$$r = \frac{R}{R + G + B}, \quad g = \frac{G}{R + G + B}, \quad b = \frac{B}{R + G + B}$$


$$r + g + b = 1$$



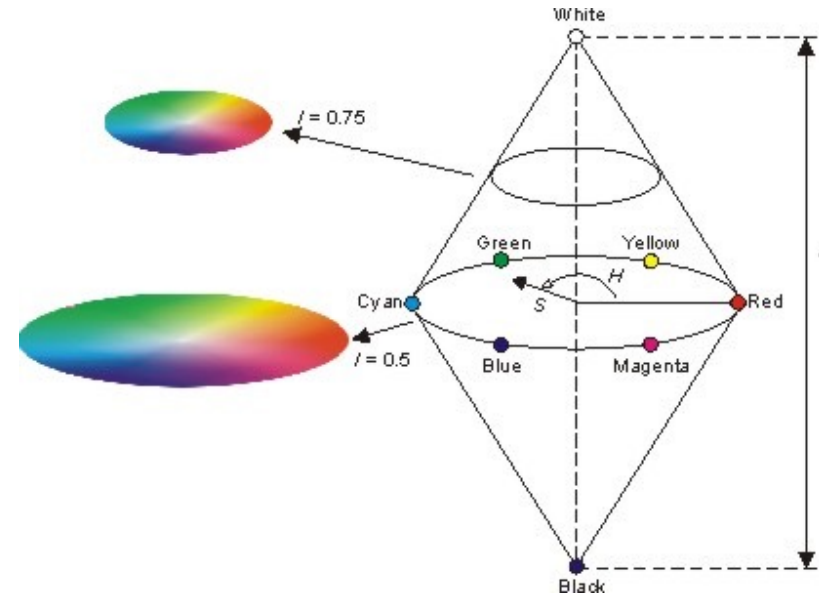
# Spațiul de culoare HSI (HSV, HSB, HSL)

HSI: (H, S, I),  $H=0 \dots 2\pi$ ,  $S=0 \dots 1$ ,  $I=0 \dots 255$

$$I = \frac{1}{3}(R + G + B)$$

$$S = 1 - \frac{3}{(R + G + B)}[\min(R, G, B)]$$


$$H = \begin{cases} \Phi & \text{if } G \geq B \\ 2\pi - \Phi & \text{if } G < B \end{cases}$$



$$\Phi = \cos^{-1} \left( \frac{\frac{1}{2}[(R - G) + (R - B)]}{\left[ (R - G)^2 + (R - B)(G - B) \right]^{\frac{1}{2}}} \right)$$



# Alte spații de culoare

**XYZ tristimulus** - transformare liniară asupra RGB:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.607 & 0.174 & 0.200 \\ 0.299 & 0.587 & 0.114 \\ 0.000 & 0.066 & 1.116 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

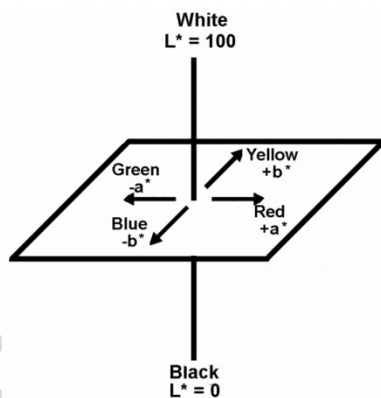
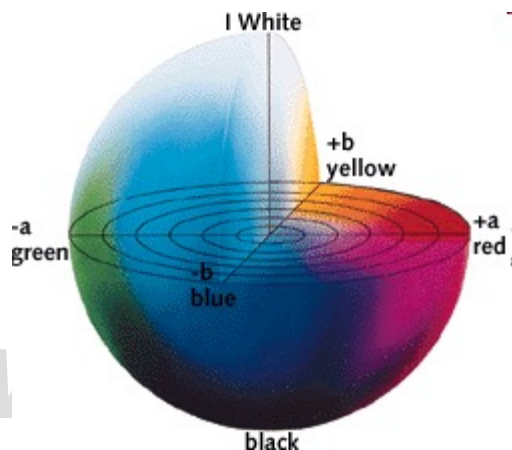
$$\begin{aligned} \text{sat} &= \sqrt{a^2 + b^2}, \\ \text{hue} &= \arctan'(b, a) \end{aligned}$$

## CIE(Lab) space

$$\begin{aligned} L &= 25(100Y/Y_0)^{1/3} - 16, \\ a &= 500 \left[ (X/X_0)^{1/3} - (Y/Y_0)^{1/3} \right] \\ b &= 200 \left[ (Y/Y_0)^{1/3} - (Z/Z_0)^{1/3} \right] \end{aligned}$$

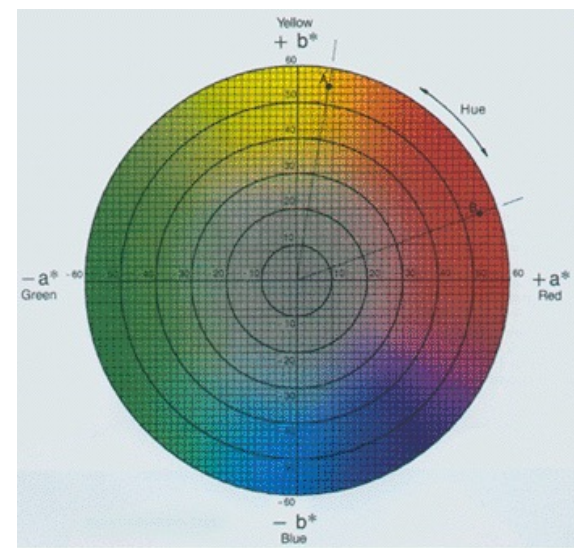
## CIE(Luv) space

$$\begin{aligned} u &= 13W(4X/(X + 15Y + 3Z) - 0.199) \\ v &= 13W(6Y/(X + 15Y + 3Z) - 0.308) \end{aligned}$$



L – componenta de intensitate

a, b - componentele de culoare cu variație liniară







# Proprietăți ale trăsăturilor cromatice

## Invarianța la translație și scalare

**Hue** – invariantă la scalarea uniformă a RGB:  $H(\alpha R, \alpha G, \alpha B) = H(R, G, B)$

**RGB-norm** – invariantă la scalarea uniformă RGB:

$$\begin{aligned} r(\alpha R, \alpha G, \alpha B) &= r(R, G, B) \\ g(\alpha R, \alpha G, \alpha B) &= g(R, G, B) \\ b(\alpha R, \alpha G, \alpha B) &= b(R, G, B) \end{aligned}$$

**Hue** – invariantă la translația uniformă RGB:  $H(R + \beta, G + \beta, B + \beta) = H(R, G, B)$

**RGB-norm** – **nu prezintă invarianță** la scalarea uniformă RGB:

$$\begin{aligned} r(R + \beta, G + \beta, B + \beta) &\neq r(R, G, B) \\ g(R + \beta, G + \beta, B + \beta) &\neq g(R, G, B) \\ b(R + \beta, G + \beta, B + \beta) &\neq b(R, G, B) \end{aligned}$$



# Proprietăți ale trăsăturilor cromatice

---

## Singularitate Hue pentru $RGB \approx 0$

$R=G=B=0 \Rightarrow H$  nedefinit

Exemple:

$$H(1,0,0) = 0, H(0,1,0) = 2\pi/3$$

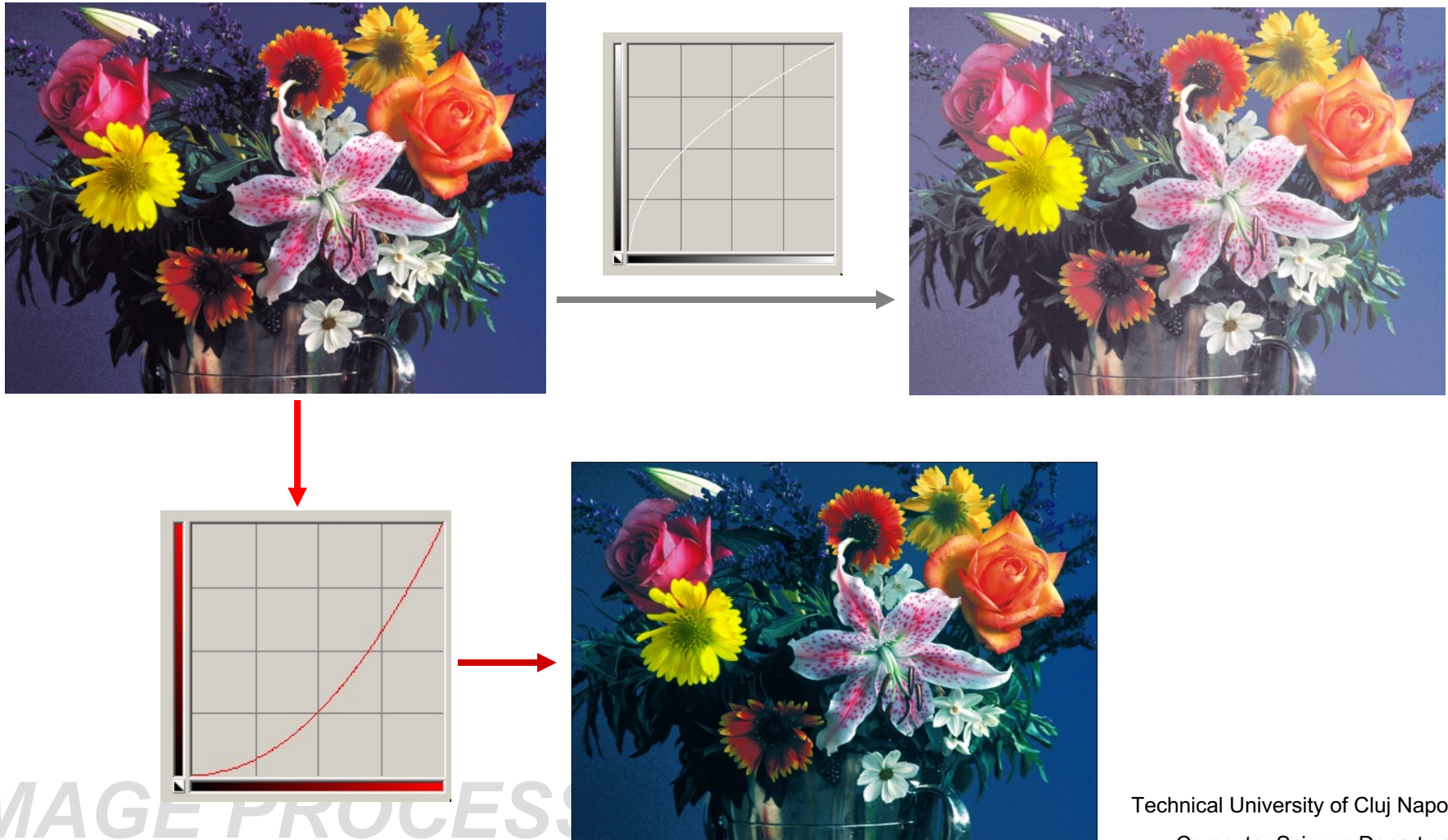
$$H(1,1,0) \Rightarrow H(2,1,0) : \delta H = \pi/6$$

Concluzie: calcularea  $H$  în zone cu intensitatea mică  $\Rightarrow$  erori numerice



# Procesări pe imagini color

Se pot aplica transformări pe fiecare componentă de culoare în parte





# Procesări pe imagini color

## Transformări folosind spațiul de culoare HSI

1. Transformare RGB  $\rightarrow$  HSI
2. Aplicare transformări pe fiecare componentă:
  - Pe I: modificare luminozitate, contrast, etc
  - Pe S: modificare a intensității culorilor
  - Pe H: modificare a nuanței (ex: deplasare spre roșu/spre albastru)
3. Transformare HSI  $\rightarrow$  RGB





# Metoda Canny pentru imagini color

---

[2] A. Koschan, M. Abidi, Digital Color Image Processing, Wiley & Sons, 2008.

## Algoritm

1. Filtrare zgomot cu un filtru trece jos ([2], cap. 5.3, pp102-117)

**2. Calculul modulului și a direcției gradientului** ([2], cap 6.1.1, pag 126-128)

3. Supresia non-maximelor

4. Binarizare cu histereză





# Metoda Canny pentru imagini color

Pas2 :

Pixel (x,y)  $\Rightarrow$  culoarea :  $C(x,y)=(R,G,B)$

Gradient  $\Rightarrow$  **Jacobian** (matricea derivatelor parțiale ale vectorului C):

$$\mathbf{J} = \begin{pmatrix} R_x & R_y \\ G_x & G_y \\ B_x & B_y \end{pmatrix} = (\mathbf{C}_x, \mathbf{C}_y).$$

$$R_x = \frac{\partial R}{\partial x} \text{ and } R_y = \frac{\partial R}{\partial y}$$



# Metoda Canny pentru imagini color

**Direcția** - vectorul propriu al  $J^T J$  corespunzător celei mai mici valori proprii:

$$\tan(2\theta) = \frac{2 \cdot \mathbf{C}_x \cdot \mathbf{C}_y}{\|\mathbf{C}_x\|^2 - \|\mathbf{C}_y\|^2}$$

$$\mathbf{C}_x = (R_x, G_x, B_x)$$

**Magnitudinea:**

$$m^2 = \|\mathbf{C}_x\|^2 \cos^2(\theta) + 2 \cdot \mathbf{C}_x \cdot \mathbf{C}_y \cdot \sin(\theta) \cos(\theta) + \|\mathbf{C}_y\|^2 \sin^2(\theta).$$



# Rezultate Canny color



Sursa imaginii: <http://lear.inrialpes.fr/people/vandeweijs/research1>



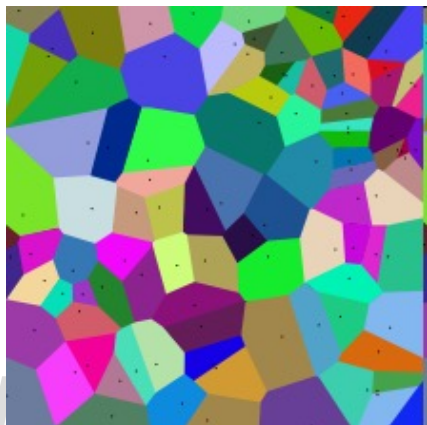
# Segmentarea imaginilor color

**Segmentare** := identificarea zonelor omogene din imagine

**Segmentare imagini color** := identificare regiuni (componente conexe) care satisfac anumite criterii de omogenitate, bazate pe trăsături derivate din componentele de culoare, definite în spațiul de culoare considerat.

**(1) Regiune (definiție bazată pe proprietățile pixelului)** := submulțime de pixeli specificată printr-o funcție de apartenență la o clasă definită în spațiul de culoare considerat. De exemplu:

- (a) Culoarea pixelului este într-un semispațiu definit de un plan;
- (b) Culoarea pixelului se încadrează într-un poliedru;
- (c) Culoarea pixelului se încadrează într-o celulă Voronoi dată de o mulțime de puncte reprezentative;



**Decompoziție (spațiu) Voronoi** := În cazul cel mai simplu (2D) se dă un set de puncte  $S$  în plan (centrele Voronoi). Fiecare centru  $s$  are asociată o celulă Voronoi  $V(s)$  conținând toate punctele mai apropiate de  $s$  decât de toate celelalte centre.

Muchiile diagramei Voronoi sunt mulțimi de puncte (segmente de dreaptă) care sunt egal depărtate de două centre.

Nodurile Voronoi sunt puncte echidistante față de 3 sau mai multe centre Voronoi



# Segmentarea imaginilor color

---

**(2) Regiune** (definiție bazată pe relația dintre pixeli) := setul maximal de pixeli pentru care este satisfăcută o condiție de uniformitate (predicat de omogenitate):

- (a) Regiuni uniforme obținute prin creșterea unui bloc (seed) prin unirea altor pixeli sau blocuri de pixeli
- (b) Regiuni uniforme obținute prin împărțirea unor regiuni mai mari care nu sunt omogene

**(3) Regiune** (definiție bazată pe noțiunea de muchie) := set de pixeli delimitați de pixeli de muchie (contur) - (condiția de muchie este un predicat de ne-omogenitate):





# 1. Segmentare la nivel de pixel

---

**Segmentarea se face în spațiul trăsăturilor (culturilor)**

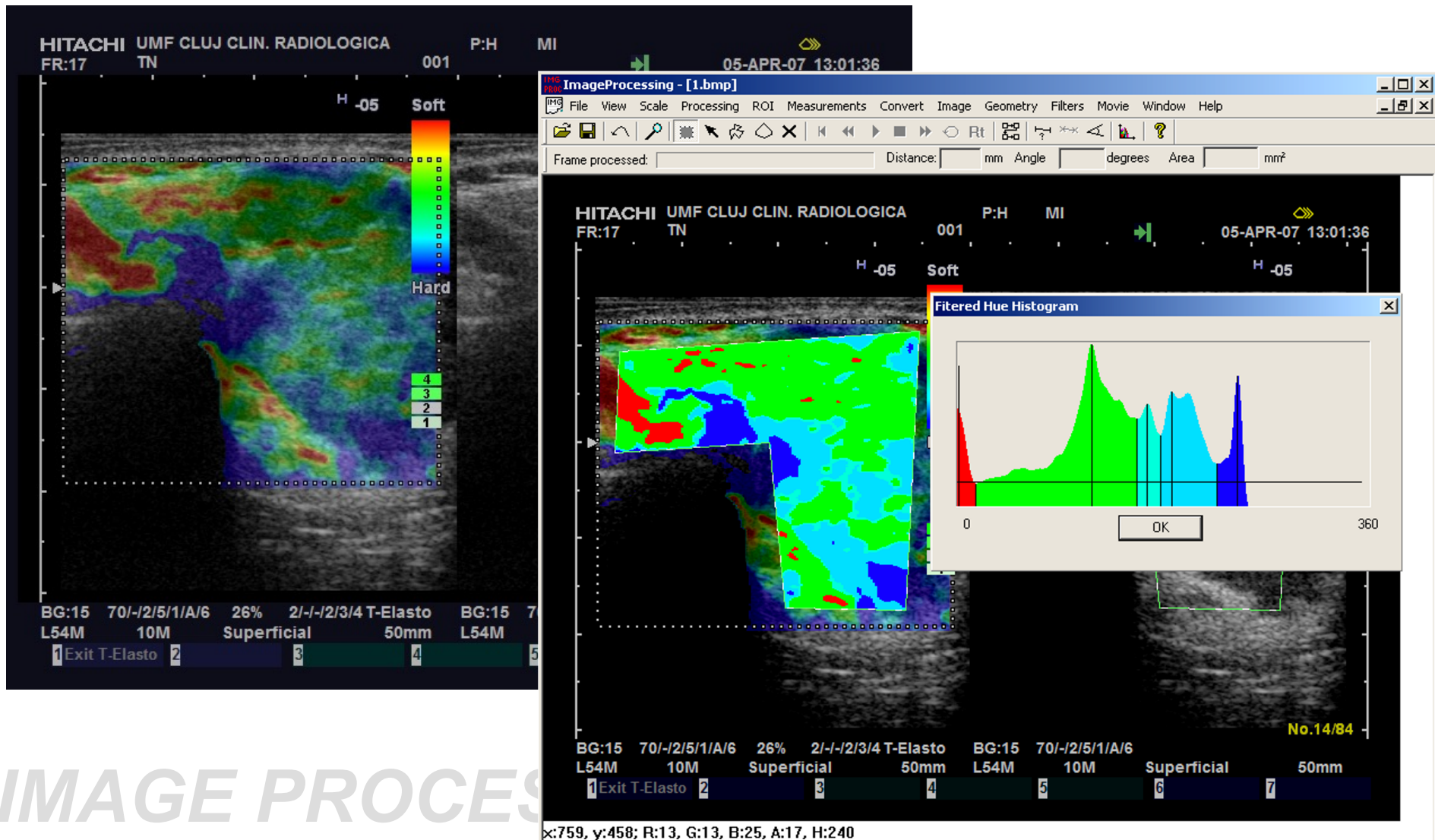
**Abordări:**

1. Bazate pe histogramă: detecția maximelor și gruparea culorilor (clustering) în jurul maximelor + clasificarea pixelilor în aceste clustere
2. Clustering în spațiul de culoare – punctele din spațiul de culoare sunt grupate în jurul unor centre reprezentative + clasificarea pixelilor în aceste clustere
3. Clustering fuzzy – pe baza funcțiilor de apartenență fuzzy



## 1.1. Segmentare bazată pe împărțirea histogramei

Detecția maximelor histogramei Hue (filtrate) și împărțirea spațiului Hue în clustere având ca centre aceste vârfuri, clasificare pixeli pe baza vârfului corespunzător.





## 1.2 Clustering în spațiul de culoare

Gruparea culorilor (din spațiul culorilor) și asignarea unei culori reprezentative fiecărui grup

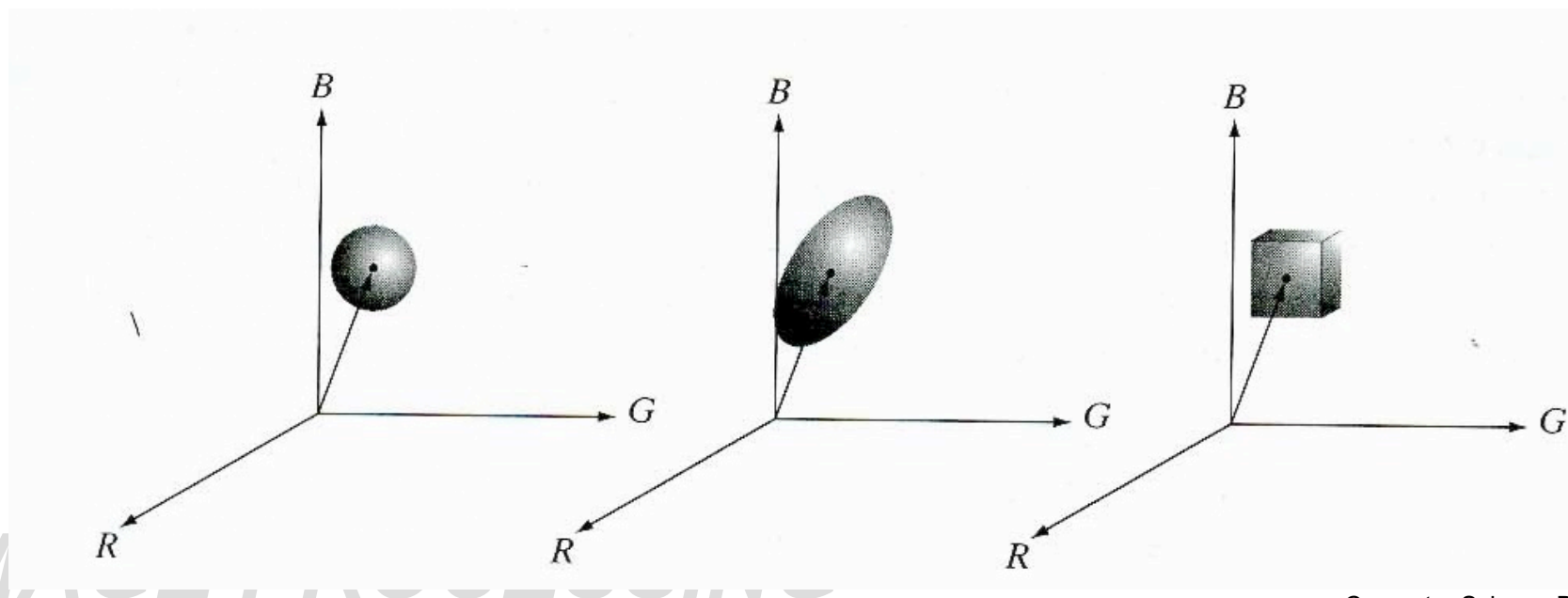
- **cuantizare/posterizare** (împărțirea spațiului trăsăturilor în subspații de dimensiuni fixe);

- **clustering**:

- supervizat – informații apriori: se știe numărul clusterelor și/sau pozițiile centrelor (de exemplu varfurile histogramelor)

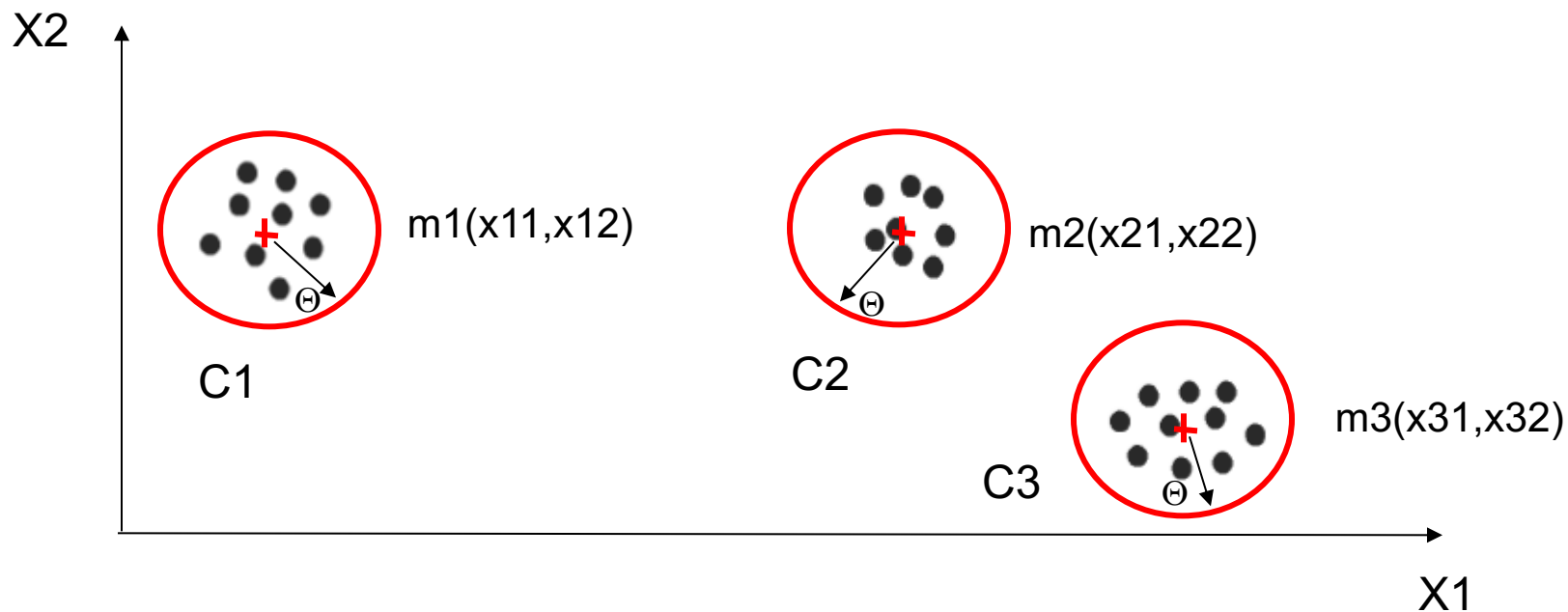
- nesupervizat – fără informații apriori

Nu se aplică de obicei pe RGB ci pe **HS** sau pe **ab** sau **uv (liniar)**





## Exemplu: clustering in $R^2$



$X_1$ ,  $X_2$  – pot fi cele două axe de coordonate corespunzătoare componentelor de culoare (H,S), (a,b) sau (u,v)



## K-means

Partiționează un set de  $n$  observații (ex. culorile pixelilor în spațiul considerat) în  $k$  cluster, în care fiecare observație va aparține de clusterul cu cel mai apropiat centru (medie).

Setul de observații:  $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ ,  $\mathbf{x}_i$  – vector  $d$ -dimensional (ex:  $\mathbf{x}_i = (a_i, b_i)$ )

Scop  $\Rightarrow k$  mulțimi ( $k \leq n$ )  $\mathbf{S} = \{S_1, S_2, \dots, S_k\}$  astfel încât să minimizăm suma pătratelor distanțelor de la fiecare punct din cluster la centrul (media) clusterului

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

$\boldsymbol{\mu}_i$  – centroidul (media clasei  $S_i$ )





# K-means clustering

## Algoritmul standard (Lloyd)

**Inițializare:** se dă un set inițial de centroide (medii/means):

$$\mathbf{m}_1(1), \dots, \mathbf{m}_k(1)$$

**Pas 1: atribuire** – se atribuie fiecare observație la clusterul cu centroidul (media) cea mai apropiată (se partiționează observațiile după diagrama Voronoi dată de medii).

$$S_i^{(t)} = \left\{ \mathbf{x}_j : \|\mathbf{x}_j - \mathbf{m}_i^{(t)}\| \leq \|\mathbf{x}_j - \mathbf{m}_{i^*}^{(t)}\| \text{ for all } i^* = 1, \dots, k \right\}$$

**Pas 2: actualizare** – se actualizează mediile (centrele) fiecărui cluster pe baza observațiilor încorporate

$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

Se repetă pașii 2 și 3. Se oprește când se atinge convergența (nu mai sunt schimbări sau se atinge un anumit număr de pași).

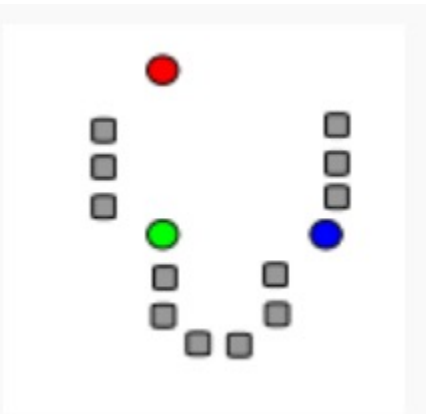


# K-means clustering

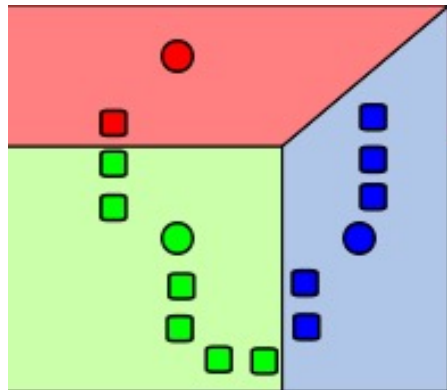
## Metode de inițializare

- Forgy** – se aleg aleator  $k$  observații ca medii initiale
- Random partition** - se asignează aleator un cluster la fiecare observație
- **Pe baza histogramelor:** pentru clustering în spațiul de culoare (HS) sau (ab) sau (uv) se pot lua cele mai pronunțate  $K$  vârfuri ale histogramei bidimensionale

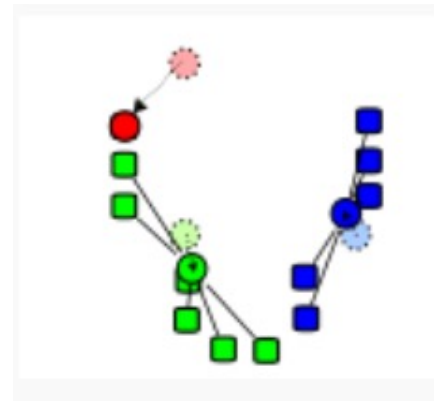
## Exemplu:



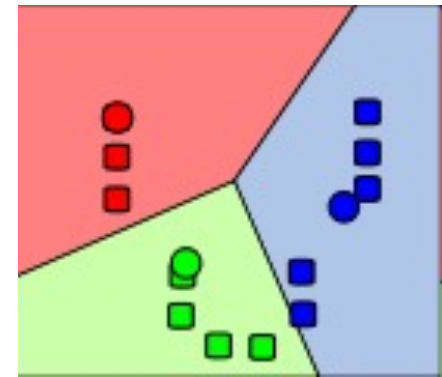
Inițializare:  
 $k=3$ , centre  
alese aleator



Pas 2:  
Partiționare  
Voronoi



Pas 3:  
Recalculare  
centroide



Repetă pașii 2 și 3  
până la  
convergență

Technical University of Cluj Napoca

Computer Science Department



# Metrice de distanță

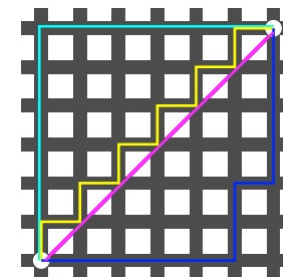
## Exemple pentru cazul 2D:

- două puncte,  $P_1 = (x_1, y_1)$  și  $P_2 = (x_2, y_2)$

**Distanța Euclidiană** – distanța geometrică între două puncte în spațiul bidimensional este definită ca lungimea segmentului de dreaptă care le unește:

$$d_{Euclidian}(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} = \|P_1 - P_2\|$$

**City block sau Manhattan** – distanța este definită ca lungimea celei mai scurte căi de între cele două puncte, dacă se pot face deplasări doar pe verticală și orizontală:



$$d_{CityBlock}(P_1, P_2) = |x_2 - x_1| + |y_2 - y_1|$$

**Chessboard sau Chebyshev** (mișcarea regelui pe tabla de șah) - parcurgerea se poate realiza în cele opt direcții spațiale:

$$d_{Chessboard}(P_1, P_2) = \max(|x_2 - x_1|, |y_2 - y_1|)$$



## 2. Segmentare la nivel de regiune

---

- În spațiul imagine
- Se bazează pe criterii de uniformitate / neuniformitate
- Tipuri:
  1. Region Growing
  2. Region Splitting
  3. Split & Merge

### **Segmentarea imaginii prin Region Splitting (împărțire)**

- (1) Împarte imaginea în blocuri B de dimensiune  $N \times N$ ;  $N = 2^n$ , n - rangul blocului.
- (2) Pentru fiecare bloc:
  - (3) Dacă  $NEUNIFORMITATEA(B) > T$  și  $k = rang(B) > 0$ , atunci
    - divide blocul B în 4 blocuri egale B1, B2, B3, B4;
    - repetă pasul (3) pentru B1, B2, B3, B4;
  - Altfel raportează B ca un bloc final (regiune).



# Region Growing (Creșterea regiunilor)

Metoda **region growing** are la bază un proces iterativ prin care regiuni ale imaginii sunt fuzionate începând de la regiuni primare (care pot fi pixeli sau alte regiuni mici – celule de bază). Iterațiile de creștere se opresc atunci când nu mai sunt pixeli de procesat.

## Algoritm:

1. Se împarte imaginea în celule de bază (dimensiune  $\geq 1$  pixel).
2. Fiecare celulă este comparată cu vecinii ei folosind o măsură de similaritate. În caz de similaritate (valoarea metricii de similaritate  $<$  prag) celulele sunt fuzionate într-un fragment mai mare, și se actualizează trăsăturile regiunii folosite la măsura similarității (de obicei prin mediere ponderată).
3. Se continuă procesul de creștere a fragmentului prin examinarea tuturor vecinilor până când nu se mai pot realiza fuziuni.
4. Se trece la următoarea celulă rămasă nemarcată și se repetă pașii 2-3. Algoritmul se oprește atunci când nu au mai rămas celule nemarcate.





## Exemplu de implementare

---

O implementare eficientă folosește o coadă (similar cu pasul 4 de la Canny). Algoritmul rulează astfel:

1. Parcurge imaginea de la stânga la dreapta și de sus în jos și găsește primul seed point (celula de baza), pune coordonatele sale în coadă, și stabilește o etichetă unică pentru acea regiune.
2. Cât timp coada nu este vidă, repetă:
  - Extrage primul punct din coadă
  - Găsește toți vecinii acestui punct care satisfac condiția de similaritate
  - Marchează în imaginea destinație vecinii acestui punct cu eticheta punctului inițial
  - Pune coordonatele acestor puncte în coadă
3. Continuă de la pasul 1 cu următorul seed point (neparcurs încă).



# Region Growing - exemple

RG pornind de la un punct definit de utilizator, folosind spațiul (H,S)



RG cu multiple regiuni, în spațiul de culoare RGB





## Referințe

---

- [1] W. Skarbek, A. Koschan, Colour Image Segmentation: A Survey, Technical report 94-32, Technische Universitat Berlin, Fachbereich 13 Informatik Franklinstrasse 28/29, 10587 Berlin, Germany
- [2] A. Koschan, M. Abidi, Digital Color Image Processing, Wiley & Sons, 2008.
- [3] [http://en.wikipedia.org/wiki/K-means\\_clustering](http://en.wikipedia.org/wiki/K-means_clustering)