

2. Spații de culoare

2.1 Introducere

Scopul acestui al doilea laborator este învățarea tehnicilor de bază în lucrul cu culorile imaginilor digitale în format bitmap.

2.2 Spațiul RGB

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**) (spațiu de culoare aditiv – Fig. 2.1 și 2.2).

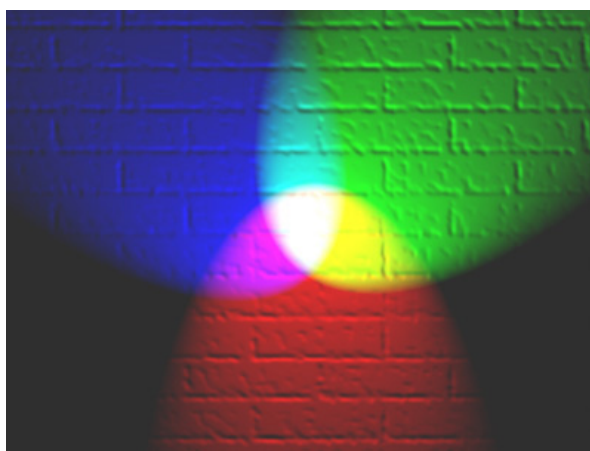


Fig. 2.1. Reprezentarea combinării aditive a culorilor. Acolo unde culorile primare se suprapun se observă apariția culorilor secundare. Acolo unde toate trei culorile se suprapun se observă apariția culorii albe[1].

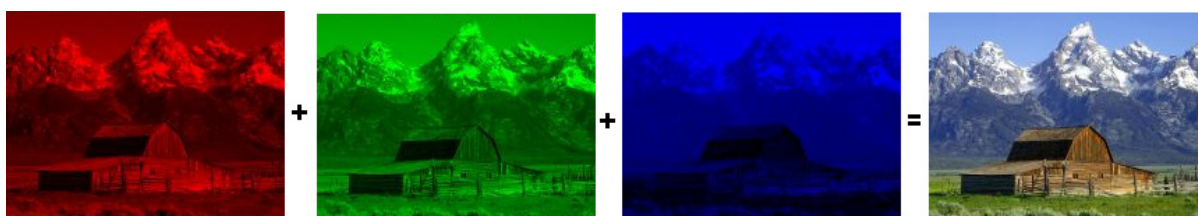


Fig. 2.2. Imaginea color se obține prin combinarea la nivel de pixel a celor trei culori primare (vezi cele trei canale).

Astfel, fiecare pixel din imagine va fi caracterizat prin câte o valoare pentru fiecare din cele trei componente de culoare primare. Culoarea sa reprezintă un punct în spațiul 3D al modelului de culoare RGB (Fig. 2.3). În acest cub al culorilor, originea axelor R, G și B corespunde *culorii negre* (0, 0, 0). Vârful opus al cubului corespunde *culorii albe* (255, 255, 255). Diagonala cubului, între negru și alb corespunde tonurilor de gri (grayscale) (R=G=B). Trei dintre vârfuri corespund culorilor primare **roșu**, **verde** și **albastru**. Celelalte 3 vârfuri corespund culorilor complementare: **turcoaz**, **mov** și **galben** (Cyan, Magenta and Yellow). Dacă translatăm originea sistemului de coordonate în punctul „alb” și redenumim cele 3 axe de coordonate ale sistemului în C, M, Y obținem spațiul de culoare complementar CMY, (folosit la dispozitive de imprimare color).

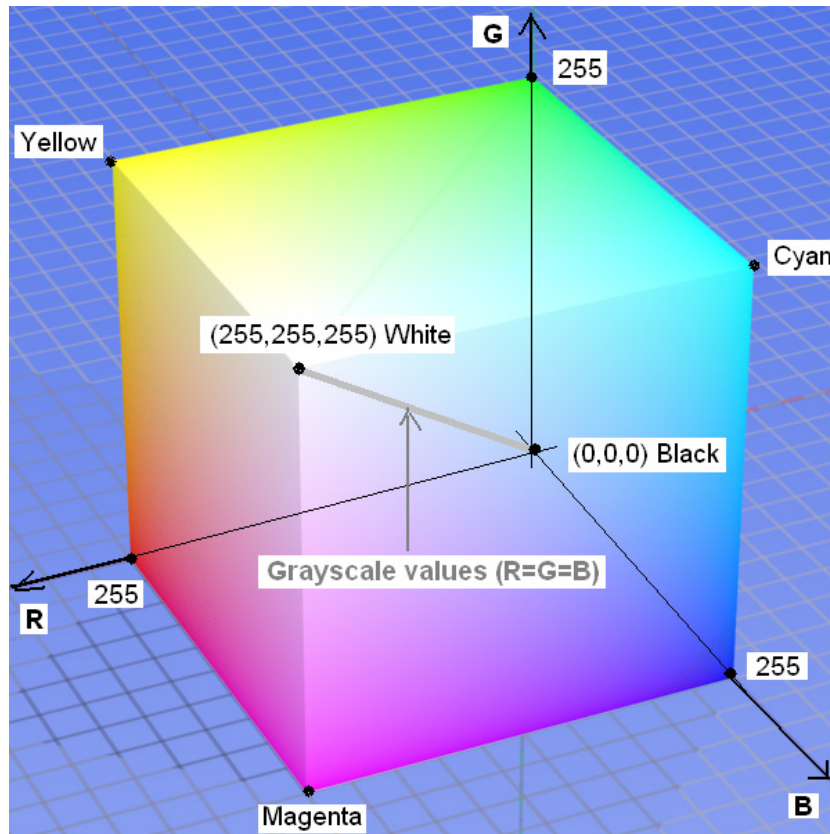


Fig. 2.3. Modelul de culoare RGB mapat pe un cub. În acest exemplu fiecare culoare este reprezentată pe câte 8 biți (256 de nivele) (imagini bitmap RGB24). Numărul total de culori este $2^8 \times 2^8 \times 2^8 = 2^{24} = 16.777.216$.

Pentru imagini RGB24 (24 biți/pixel) spațiul de culoare poate fi reprezentat complet. Într-o imagine indexată (cu paletă) poate fi reprezentat doar un anumit subspațiu al spațiului de culoare din figura 2.3. În acest context, numărul de biți/pixel (numărul de biți folosiți pentru codificarea unei culori) se numește „adâncime de culoare” (color depth). (Tabelul 2.1):

Tabel 2.1. Adâncimea și tipul imaginii

Adâncimea de culoare	Nr. Culori	Mod de culoare	Palette (LUT)
1 bit	2	Indexed Color	Yes
4 biți	16	Indexed Color	Yes
8 biți	256	Indexed Color	Yes
16 biți	65536	True Color	No
24 biți	16.777.216	True Color	No
32 biți	16.777.216	True Color	No

Există și alte modele de culoare[2] care nu vor fi discutate aici.

2.3 Conversia unei imagini color într-o imagine grayscale

Pentru a converti o imagine color într-o imagine grayscale, cele trei componente ale culorii fiecărui pixel trebuie egalizate. O metodă des folosită este medierea celor 3 componente:

$$R_{Dst} = G_{Dst} = B_{Dst} = \frac{R_{Src} + G_{Src} + B_{Src}}{3} \quad (2.1)$$

2.4 Conversia imaginilor grayscale în imagini binare (alb-negru)

O imagine binară (alb-negru) este o imagine care conține doar două culori: alb și negru. O imagine binară se obține dintr-o imagine grayscale printr-o operație simplă numită binarizare cu prag (thresholding). Binarizarea cu prag este cea mai simplă tehnică de segmentare a imaginilor, care permite separarea obiectelor de background. (Fig. 2.4).

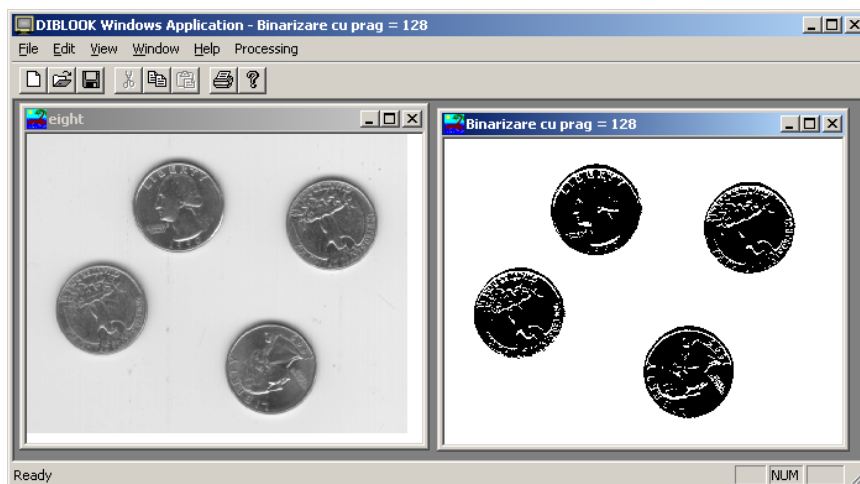


Fig. 2.4. Binarizarea cu prag.

În acest laborator va fi discutată binarizarea cu prag fix (ales arbitrar) pentru imagini indexate (8 biți/pixel) de tip grayscale. Binarizarea poate fi aplicată prin parcurgerea valorilor pixelilor din imaginea sursă și înlocuirea lor în imaginea destinație cu valoarea dată de:

$$Dst(i, j) = \begin{cases} 0 & (\text{black}) \quad , \quad \text{if } Src(i, j) < \text{threshold} \\ 255 & (\text{white}) \quad , \quad \text{if } Src(i, j) \geq \text{threshold} \end{cases} \quad (2.2)$$

2.5 Spațiul de culoare HSV (Hue Saturation Value)

Este un spațiu / model de culoare invariant (componentele H (culoare) și S (saturație) sunt separate și cvasi-independente de iluminarea scenei caracterizată de V (intensitate)). Se reprezintă sub forma unei piramide cu baza hexagonală sau a unui con.

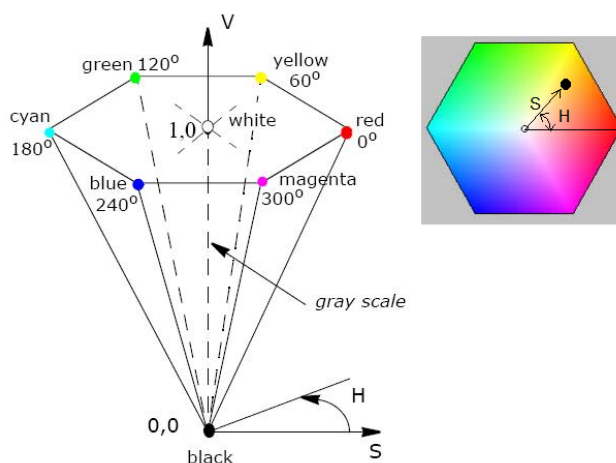


Fig. 2.5. Modelul (spațiul) de culoare HSV.

Unde:

H – reprezintă unghiul făcut de culoarea curentă cu raza corespunzătoare culorii Roșu

S – reprezintă distanța culorii curente față de centrul bazei piramidei/conului

V – reprezintă înălțimea culorii curente în piramida/con

2.6 Transformarea RGB → HSV

Ecuatiile de transformare din componentele RGB în HSV sunt [3]:

```

r = R/255; // r : componenta R normalizată
g = G/255; // g : componenta G normalizată
b = B/255; // b : componenta B normalizată
// Atenție declarați toate variabilele pe care le folosiți de tip float
// Dacă ați declarat R de tip uchar, trebuie să faceți cast: r = (float)R/255 !!!

M = max (r, g, b); // Atenție: în Visual C există predefinit un macro pentru min() și max()
m = min (r, g, b); // care primește 2 parametri. Cu 3 parametri nu dă eroare la compilare,
                  // dar rezultatul este incorect. Rescrieți în forma corectă cu 2 parametri.
C = M - m;

Value:
    V = M;

Saturation:
    If (V!=0)
        S = C / V;
    Else // negru
        S = 0;

Hue:
    If (C!=0) {
        if (M == r) H = 60 * (g - b) / C;
        if (M == g) H = 120 + 60 * (b - r) / C;
        if (M == b) H = 240 + 60 * (r - g) / C;
    }
    Else // grayscale
        H = 0;
    If (H < 0)
        H = H + 360;

```

Valorile pt. H, S și V calculate cu formulele de mai sus vor avea următoarele domenii de valori:

```

H = 0 .. 360
S = 0 .. 1
V = 0 .. 1

```

Aceste valori se normalizează (scalează) în intervalul 0 .. 255 pentru a reprezenta fiecare componentă de culoare ca și o imagine cu 8 biți/pixel (de tip CV_8UC1):

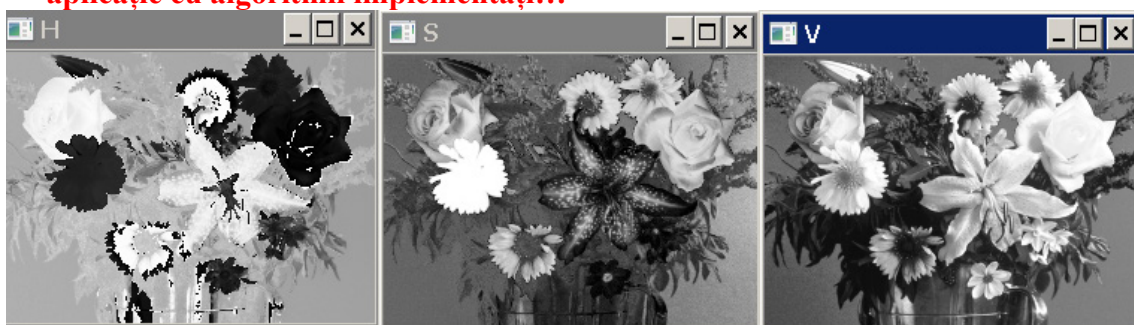
```

H_norm = H*255/360
S_norm = S*255
V_norm = V*255

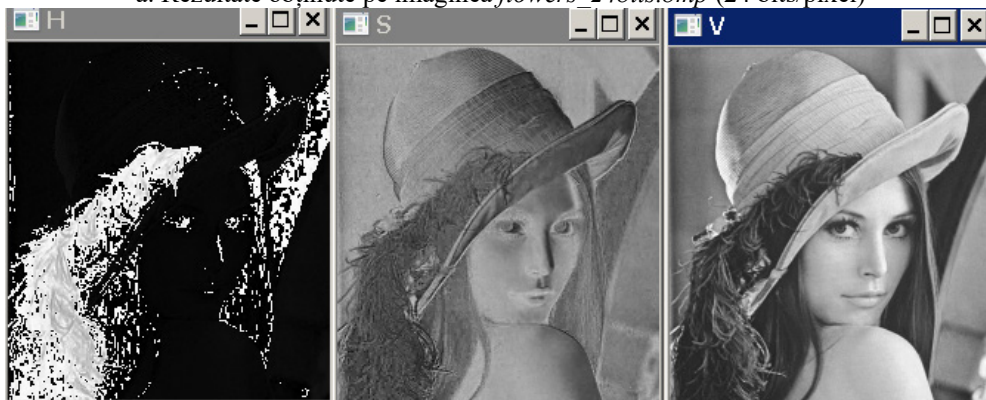
```

2.7 Activități practice

1. Adăugați la framework o funcție care copiază canalele R, G, B ale unei imagini RGB24 (tip CV_8UC3) în trei matrice de tip CV_8UC1. Afișați aceste matrice în 3 ferestre diferite.
2. Adăugați la framework o funcție de conversie de la o imagine color RGB24 (tip CV_8UC3) la o imagine grayscale de tip (CV_8UC1) și afișați imaginea rezultat într-o fereastră destinație.
3. Adăugați la framework o funcție de procesare pentru conversia de la *grayscale* la *alb-negru* pentru imagini grayscale (CV_8UC1), folosind (2.2). Citiți valoarea pragului de la linia de comandă. Testați operația de binarizare folosind diverse imagini și diverse praguri.
4. Adăugați la framework o funcție care convertește canalele R,G,B ale unei imagini RGB24 (tip CV_8UC3) în componente H, S, V folosind ecuațiile din 2.6. Memorați componente H, S, V în câte o matrice de tip CV_8UC1 corespunzătoare canalelor H, S, V. Afișați aceste matrice în 3 ferestre diferite. Verificați corectitudinea implementării prin comparație vizuală cu rezultatele de mai jos.
5. Implementați o funcție *isInside(img, i, j)* care verifică dacă poziția indicată de perechea (i,j) (rând, coloană) este înăuntrul imaginii *img*.
6. **Salvați-vă ceea ce ați lucrat. Utilizați aceeași aplicație în laboratoarele viitoare. La sfârșitul laboratorului de procesare a imaginilor va trebui să prezentați propria aplicație cu algoritmi implementați!!!**



a. Rezultate obținute pe imaginea *flowers_24bits.bmp* (24 bits/pixel)



b. Rezultate obținute pe imaginea *Lena_24bits.bmp* (24 bits/pixel)

Fig. 2.6. Exemple ale conversiei imaginilor din spațiul de culoare RGB în HSV.

2.8 Referințe

- [1] http://en.wikipedia.org/wiki/RGB_color_model
- [2] http://en.wikipedia.org/wiki/Color_models
- [3] Open Computer vision Library, Reference guide, *cvtColor()* function, https://docs.opencv.org/4.5.1/d8/d01/group_imgproc_color_conversions.html#ga397ae87e1288a81d2363b61574eb8cab