1. Segmentare imagini color (1): conversii intre modele de culoare si construirea histogramelor de culoare

Scop: transformare componentelor de culoare ale unei imagini color din modelul RGB24 (Red Green Blue) in modelul HSI (Hue Saturation Intensity), model invariant la variatii de luminozitate; calculul si afișarea histogramelor de culoare ale componentelor H, S si V, tratarea evenimentelor de mouse pentru afisarea de informatii din imaginile color.

1.1. Transformarea RGB ⇒ HSV

Modelul de culoare RGB nu este adecvat pentru a fi folosit in operatiile de segmentare a imaginii deoarce nu este invariant la variatiile de iluminare ale scenei (canalele RGB includ atat componenta cromatica cat si componenta de intensitate. In cazul modelului HSV componenta de intensitate (V – Value) este separata de componenta cromatica (H - Hue) si de componenta de saturatie (S – Saturation)

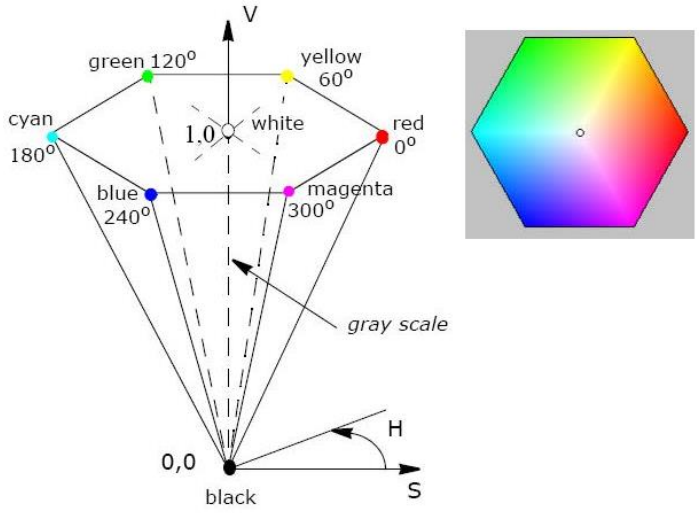

Fig. 1.1. Reprezentarea modelul (spaţiului de culoare) HSV.

Ecuaţiile de transformare din componentele RGB în HSV sunt [2]:

r = R/255; // r : componenta R normalizata
g = G/255; // g : componenta G normalizata
b = B/255; // b : componenta B normalizata
// Atentie declarati toate variabilele pe care le folositi de tip float
// Daca ati declarat *R* de tip *uchar*, trebuie sa faceti cast: *r = (float)R/255* !!!

M = max (r, g, b);
m = min (r, g, b);
C = M - m;

Value:
V = M;

Saturation:
If (C)

S = C / V;
Else // grayscale
S = 0;
Hue:
If (C) {
*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 pentru H, S si 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 pt. a reprezenta fiecare componenta 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

1.2. Transformarea unei imagini color din modelul RGB in HSV cu ajutorul functiei cvtColor si afisare componentelor H,S,V

Conversia intre cele 2 modele de culoare se poate face in OpenCV cu ajutorul functiei `cvtColor(src, dest, tip_concersie)` [2]. Imaginea rezultata (modelul HSV) este o matrice cu elemente cu 24 biti/pixel (3 canale, fiecare cu 8 biti/pixel). Separarrea imaginii cu 3 canale in 3 imagini / matrici distincte, fiecare cu cate 8 biti/pixel s epoate face cu ajutorul functiei split:

```
void myBGR2HSV()
{
	char fname[MAX_PATH];
	while (openFileDlg(fname))
	{
		Mat rgb = imread(fname);

		Mat hsv;
		Mat channels[3];
		cvtColor(rgb, hsv, COLOR_BGR2HSV);

		split(hsv, channels);

		// Componentele de culoare ale modelului HSV
		Mat H = channels[0]*255/180;
		Mat S = channels[1];
		Mat V = channels[2];

		imshow("input rgb image", rgb);
```

```
            imshow("input hsv image", hsv); // vizualizarea matricii hsv (ca un mat cu
      3 canale) nu are semnificatie vizuala utila / relevanta

            imshow("H", H);
            imshow("S", S);
            imshow("V", V);

            waitKey();
      }
}
```

Observatie: in OpenCV valorile celor trei canale ale unei imagini color RGB cu 24 biti/pixel sunt stocate in ordineea Blue, Green, Red (BGR). Din acest motiv, argumentul care specifica tipul converrsiei din functia cvtColor are prefixul Color_BGR2xxx.

1.3. Activități practice

1. Intelegeti si implementati exemplul de la punctul 1.2 din descrierea lucrarii

2. Plecand de la implementarea de la puntul 1.2, se vor calcula histogramele (vezi capitolul 8 [4]) componentelor de culoare normalizate (0 .. 255) ale canalelor H, S, V. Se vor afisa aceste histograme. Pt. afisare exista functia predefinita `showHistogram` (definita in modulul *Functions.cpp*)

```
showHistogram (nume_fereastra, hist, hist_cols, hist_height, true)
```

unde:

hist - vector de tip int in care se calculeaza valorile histogramei
hist_cols – lungimea vectorului histograma (256)
hist_height – inaltimea ferestrei in care se afiseaza histograma

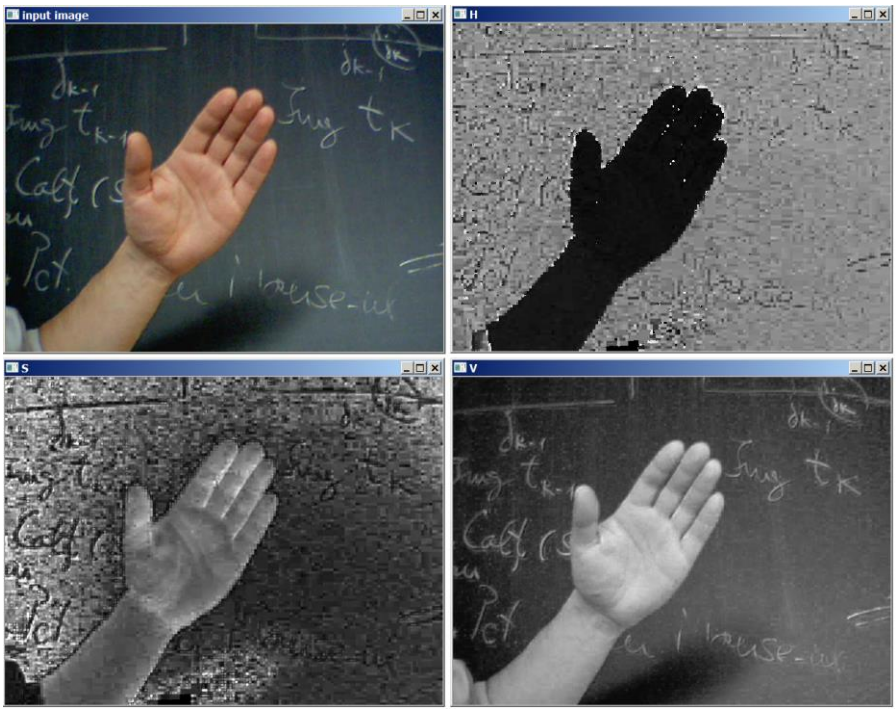

Fig. 1.2. Exemplu cu rezultatele care ar trebui obtinute la punctul 2 (imaginea sursa, canalul H, canalul S, canalul V).

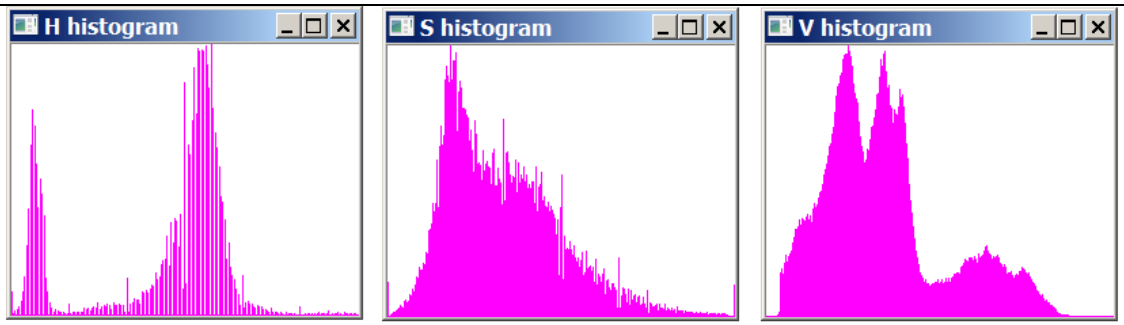

Fig. 1.3. Exemplu cu rezultatele care ar trebui obtinute la punctul 2 (histograma canalului H, histograma canalului S, histograma canalului V).

3. Se va face o analiza a histogramelor H, S, V pentru toate imaginile de test ce contin o mana si se va decide care componenta de culoare este cea mai potrivita pentru segmentarea mainii (spararea pixelilor de man de cei de fundal)

4. Se va binariza matricea componentei de culoare alese pt. segmentare cu un prag arbitrar ales astfel incat sa se separe cat mai bine pixelii de mana fata de cei de fundal si se va afisa rezultatul (fundalul cu negru, mana cu alb).

5. Calculati un prag de binarizare automat pentru componenta de culoare aleasa folosind implementarea algoritmului din capitolul 8.5 [4] si binarizati componenta respectiva cu acest prag. Comparati rezultatul cu cel de la pct. 3. Testati binarizarea pe toate imaginile de test din arhiva *img.zip.*

6. Adaugati o functie care afiseaza la linia de comanda coordonatele pe care ati facut click si valorile componentlor de culoare H,S,V ale pixelului pe care se face click pe imaginea sursa(rgb) (vezi `myBGR2HSV()`, `testMouseClick()`, `MyCallBackFunc()`). Verificati corectitudinea implementarii folosind utilitarul `ImageWatch`.

7. Adaugati o functie care afiseaza pe imaginea sursa (rgb) folosind functia `putText` valorile componentlor de culoare H,S,V ale pixelului peste care se deplaseaza mouse-ul (`EVENT_MOUSEMOVE`) si valorile coordonatelorcurente (vezi `myBGR2HSV()`, `testMouseClick()`, `MyCallBackFunc()`).Verificati corectitudinea implementarii folosind utilitarul `ImageWatch`.

Observatii:
- deoarece functiei de callback puteti sa ii transmiteti un singur parametru (matricea hsv), va trebui sa declarati matrivea sursa (rgb) global (in afara corpului functiei);
- ca sa nu alterati sursa, la afisare (in if-ul aferent evenimentului mouse-move din functia de callback) faceti o copie/clona a imaginii sursa(rgb) intr-o alta matrice temp in care veti adauga textul: `Mat temp = rgb.clone();`

```
char msg[100];
sprintf(msg, "string fotmatare",valori);
putText(temp, msg, Point(5, 20), FONT_HERSHEY_SIMPLEX, 0.5, CV_RGB(0, 255, 0), 1, 8);
imshow("rgb", temp);
```

8. 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 algoritmii implementați!!!

1.4. Bibliografie

[1] Intel, Color models, https://software.intel.com/en-us/node/503873.
[2[Open Computer vision Library, Reference guide, cvtColor() function, http://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html#cvtcolor , https://docs.opencv.org/4.9.0/d8/d01/group__imgproc__color__conversions.html

[3] Open Computer vision Library, Reference guide, split() function https://docs.opencv.org/4.x/d2/de8/group__core__array.html#ga0547c7fed86152d7e9d0096029c8518a

[4] S. Nedevschi, T. Marita, R. Danescu, F. Oniga, R. Brehar, I. Giosan, C. Vancea, R. Varga, Procesarea Imaginilor - Îndrumător de laborator, editia a 2-a, Editura U.T. Press, Cluj-Napoca, https://biblioteca.utcluj.ro/carti-online-cu-coperta.html, 2023.

[5] Image Watch, Image Watch - Visual Studio 2015 | Microsoft Learn, Image Watch for Visual Studio 2022 - Visual Studio Marketplace

1.5. Anxa

Folosirea “plug-in”-ului “Image Watch”

Image Watch “plug-in” pentru Visual Studiu permite vizualizarea imaginilor (structuri de tip *Mat*) din memori in timp ce depanati o aplicatie. Folosirea plug-in-ului poate fi utila in gasirea erorilor sau la intelegerea unei anumite parti de cod prin vizualizarea continutului intermediar / final al imaginilor procesate.

Pentru deschiderea ferestrei “Image Watch” accesati din Visual Studio meniul “View” -> “Other Windows” -> “Image Watch”:

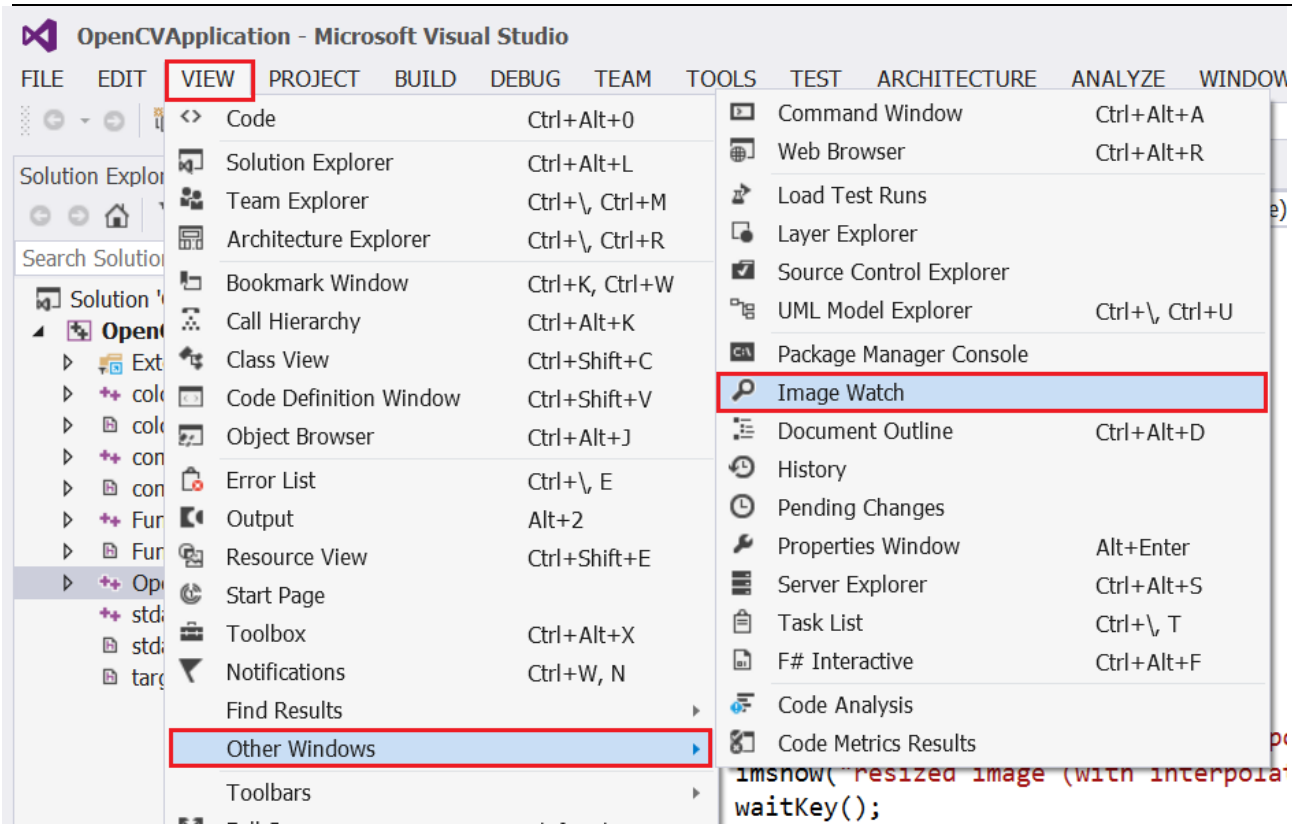

Fig. 1.4. Accesul la fereastra “Image Watch” in Visual Studio.

Pentru utilizare, inserati un breakpoint in functia pe care vreti sa o depanati. In fereastra “ImageWatch” veti putea vizualiza continutului imaginilor (variabilelor de tip Mat) aferente contextului local al functiei depanate.

Puteti face zoom-in/out pe o anumita zona din imagine (mouse scroll) si puteti vizualiza continutul canalelor imaginii la o anumita locatie (pixel): x y | c0 c1 … cN (canalele ci sunt afisate in ordinea in care apar in memorie). Mai multe informatii despre utilizarea Image Watch gasiti in [5].