Universitatea Tehnică din Cluj-Napoca
Facultatea de Automatică și Calculatoare
Departamentul Calculatoare

Segmentarea imaginilor color bazată pe region growing

Student: Ureche Simona
30234

anul 2025

1. Introducere

Segmentarea imaginilor reprezintă o etapă fundamentală în procesarea imaginilor, având ca scop împărțirea unei imagini în regiuni sau segmente uniforme obținute fie prin creșterea unui bloc de pixeli, fie prin împărțirea unor regiuni mai mari care nu sunt omogene. În acest proiect problema abordată este implementarea unui algoritm de segmentare a imaginilor color bazat pe tehnica de *region growing.* Această tehnică iterativă începe cu procesarea unui pixel și adaugă progresiv pixelii vecini pe baza unor criterii de similitudine.

Segmentarea bazată pe region growing este utilă în aplicații precum:

- **Medicină** – delimitarea tumorilor și a leziunilor în imagini medicale (RMN, CT)
- **Imagini Satelitare/Aeriene** - delimitarea zonelor de vegetație, detectarea zonelor afectate de dezastre naturale
- **Robotică** - recunoașterea obiectelor pentru navigarea autonomă.

2. Consideraţii teoretice

a)Thresholding (Segmentare bazată pe praguri): compara toți pixelii unei imagini gri cu praguri specificate și atribuie fiecare pixel la diferite categorii pe baza rezultatelor comparației.

b)Edge-based: metodă funcționează pe premisa că limitele obiectelor din imagini corespund de obicei unor schimbări bruște de intensitate, culoare sau textura.

c) Clustering: metodă care grupează pixelii în clustere bazate pe distanța în spațiul de culoare.

1) După partea de selectare a imaginii color si extragerii caracteristicilor acesteia, urmează partea de filtrare cu filtru Gaussian „trece-jos” de dimensiune 5x5 pe fiecare dintre canalele R-G-B, pentru eliminarea zgomotului. Filtrul “trece-jos” permite trecerea doar pentru frecvențele joase, rezultând blur în imaginea finală.

$$I_D(i,j) = \frac{1}{c}\sum_{u=0}^{w-1}\sum_{v=0}^{w-1} H(u,v)\cdot I_S(i+u-k, j+v-k)$$

$$c = \sum_{u=0}^{w-1}\sum_{v=0}^{w-1} H(u,v)$$

H(u, v) – filtrul

Is(i+u-k, j+v-k) – valoarea pixelului corespunzator din imaginea initiala Is, translatata in functie de pozitia filtrului

⇨ Convolutia intre filtrul Gaussian si imaginea sursa Is => o imagine estopata Id cu zgomot redus

2) Urmează partea de conversie a imaginii din RGB în L*u*v folosind funția cvtColor implementată deja în OpenGL, urmând ca procesarea sa fie făcută pe canalele u*v* - COLOR_RGB2Luv.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \leftarrow \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$L \leftarrow \begin{cases} 116 * Y^{1/3} - 16 & \text{for } Y > 0.008856 \\ 903.3Y & \text{for } Y \leq 0.008856 \end{cases}$$

$$u' \leftarrow 4 * X/(X + 15 * Y + 3Z)$$

$$v' \leftarrow 9 * Y/(X + 15 * Y + 3Z)$$

$$u \leftarrow 13 * L * (u' - u_n) \quad \text{where} \quad u_n = 0.19793943$$

$$v \leftarrow 13 * L * (v' - v_n) \quad \text{where} \quad v_n = 0.46831096$$

- În continuare, calculăm deviația standard pe fiecare canal și stocam rezultatele in 3 variabile ch1_std, ch2_std, ch3_std. Deviația standard depinde de contrastul imaginii și se calculează:

$$\sigma = \sqrt{\sum_{g=0}^{L}(g - \mu)^2 \cdot p(g)}$$

$$\sigma = \sqrt{\sum_{i=0}^{H-1}\sum_{j=0}^{W-1}(I(i,j) - \mu)^2}$$

3) Se stochează fiecare regiune intr-o matrice de etichete de dimensiunea imaginii. Este inițializata la 0, ulterior folosindu-ne de etichete pentru a marca cu un numar diferit de 0 pixelii care nu au fost încă parcurși.
4) Se parcurge imaginea pixel cu pixel:
   4.1) Este căutată prima celulă nemarcată in matricea de etichete a imaginii.
   4.2) Se folosește o coadă pentru a gestiona pixelii care urmeaza să fie procesați, iar pentru fiecare pixel din coadă se verifică vecinii săi(8 vecinătăți). Dacă vecinul nu e etichetat si respectă metrica de similaritate stabilită(valoare metrică < prag (introdus de către utilizator de la tastatura)) este adăugat la regiune și marcat cu eticheta curentă.

   a. Valoarea pragului *T* se va alege în funcție de parametrii modelului de culoare construit anterior (modelul HSV, YCbCr sau L*u*v): $T = value \times \text{dist}([0,0]^T, [ch1\_std, ch2\_std]^T)$ și *value* = 0.1 ... 3.

   b. Funcția dist(P_1, P_2) definește metrica de distanță între două puncte din spațiul modelului de culoare și poate avea mai multe forme:

   b1. distanța Euclidiană: $\text{dist}_{\text{Euclidian}}(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

4.3) Algoritmul de region growing:

repetă

- pentru fiecare vecin (i, j) din cei 8 ai pixelului din poziția *front* a listei:
  - dacă $labels(i, j) = 0$ (pixel neprocesat încă) și disimilaritatea calculată ca $dist([ch1(i, j), ch2(i, j)]^T, [ch1\_avg, ch2\_avg]^T) < T^*$ atunci:
    - adaugă pixelul (i, j) în lista FIFO la poziția *last*
    - pixelul (i, j) primește eticheta k: $labels(i, j) = k$ (este adăugat la regiunea curentă)
    - se actualizează valoarea medie a regiunii curente pe canalele de lucru:

$$ch0\_avg = \frac{N \times ch0\_avg + ch0(i,j)}{N+1} \quad ch1\_avg = \frac{N \times ch1\_avg + ch1(i,j)}{N+1} \quad ch2\_avg = \frac{N \times ch2\_avg + ch2(i,j)}{N+1}$$

    - incrementează N;
- șterge elementul din poziția *front* a listei

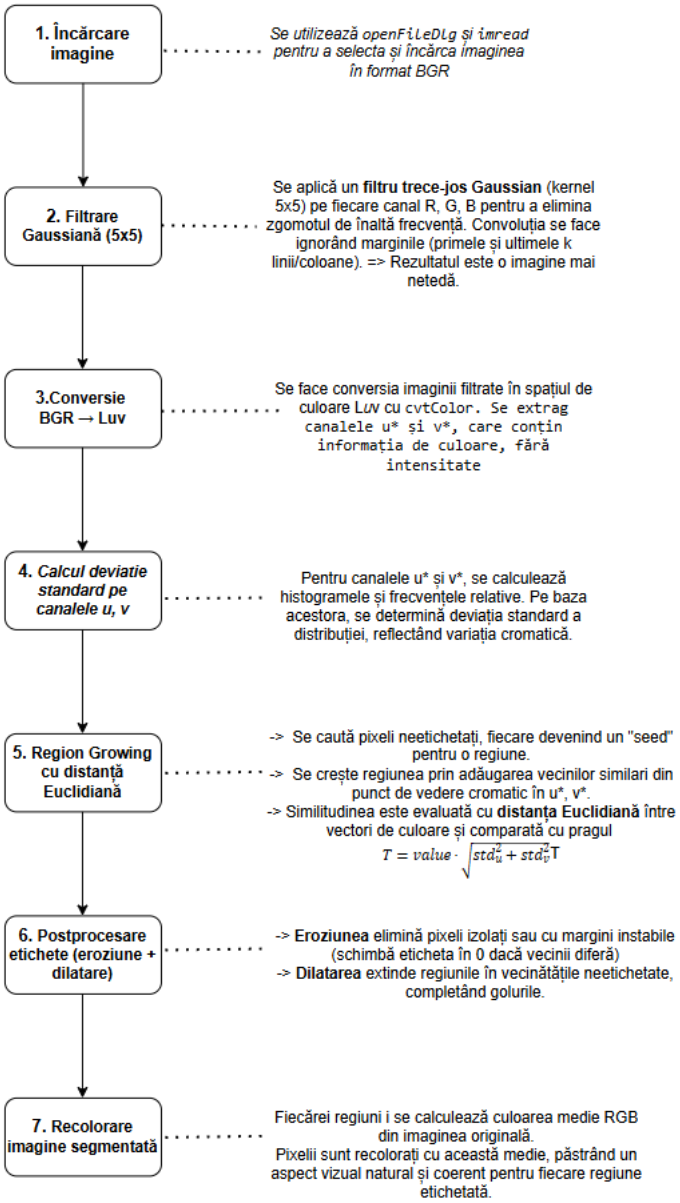
până când lista FIFO devine goală

4.4) Se trece la următoarea celulă rămasă nemarcată și se repetă pașii. Algoritmul este finalizat când nu mai există celule nemarcate.

5) În final realizăm partea de postprocesare a imaginii segmentate prin reducerea zgomotului folosind metode de dilatare si eroziune aplicate succesiv:
   a. Dilatarea(**A** $\oplus$ **B**): se copiază matricea labels într-o matrice new_labels. Pentru fiecare pixel etichetat (!=0), dacă are cel puțin un vecin (4/8-conectat) cu altă etichetă, devine neetichetat (=0). La final, new_labels înlocuiește labels. - Extinde regiunile și umple goluri mici.
   b. Eroziunea (**A** $\ominus$ **B)**: se copiază matricea labels într-o matrice new_labels. Pentru fiecare pixel etichetat (!=0), toți vecinii neetichetați (=0) primesc eticheta sa. Dacă un pixel neetichetat are toți vecinii neetichetați, rămâne neetichetat. La final new_labels înlocuiește labels. - Îngustează regiunile și elimină zgomotul.

3. Specificații de implementare

a. *Descrierea metodei utilizate*



b. *Detalierea conceptelor teoretice și algoritmilor*

1. ***Filtrare Gaussiană:***

Filtrul Gaussian este un filtru de tip trece-jos, utilizat pentru netezirea imaginii și reducerea zgomotului. Acesta implică aplicarea unei ***operații de convoluție*** *între o mască (kernel) și imaginea sursă ISI_SIS. Formula generală este:*

$$I_D(i,j) = \sum_{u=0}^{w-1} \sum_{v=0}^{w-1} H(u,v) \cdot I_S(i+u-k, j+v-k)$$

unde w este dimensiunea kernelului, iar k=w−1/2. Nucleul este poziționat peste imagine, iar pixelul curent este înlocuit cu media ponderată a vecinilor.Filtrele Gaussiane conțin doar valori pozitive, astfel că rezultatul este normalizat prin împărțirea cu suma valorilor nucleului c:

$$I_D(i,j) = \frac{1}{c} \sum_{u,v} H(u,v) \cdot I_S(i+u-k, j+v-k)$$

⇨ ***Acest filtru estompează detaliile fine și evidențiază structura generală a imaginii.***

2. ***Conversia BGR → Luv***

*Pentru o segmentare eficientă pe bază de culoare, imaginea este convertită în spațiul L*u*v folosind funcția cvtColor (OpenCV). L*u*v este un spațiu perceptual uniform, în care:*

- ***L**** *reprezintă luminozitatea (lightness),*
- ***u**** *și* ***v**** *codifică crominanța (informație despre culoare)*

3***. Calculul deviației standard***

Se calculează ***deviația standard*** *pentru canalele u* și v*, pentru a estima variația cromatică din imagine. Aceasta se face pe baza distribuției pixelilor:*

$$\sigma = \sqrt{\sum_i f_i \cdot (x_i - \mu)^2}$$

Deviația standard este folosită în segmentare pentru stabilirea pragului de similitudine:

$$T = value \cdot \sqrt{\sigma_u^2 + \sigma_v^2}$$

4. ***Algoritmul Region Growing***

Algoritmul de segmentare Region Growing pornește de la un seed (nucleu) și crește regiunea adăugând pixeli vecini similari. Se folosește o coadă FIFO (queue) și o matrice de etichete labels, inițializată cu 0.

repetă

- pentru fiecare vecin (i, j) din cei 8 ai pixelului din poziția *front* a listei:
  - dacă *labels*$(i, j) = 0$ (pixel neprocesat încă) și disimilaritatea calculată ca dist($[ch1(i, j), ch2(i, j)]^T$, $[ch1\_avg, ch2\_avg]^T$) < T* atunci:
    - adaugă pixelul (i, j) în lista FIFO la poziția *last*
    - pixelul (i, j) primește eticheta k: *labels*$(i, j) = k$ (este adăugat la regiunea curentă)
    - se actualizează valoarea medie a regiunii curente pe canalele de lucru:

$$ch0\_avg = \frac{N \times ch0\_avg + ch0(i,j)}{N+1} \quad ch1\_avg = \frac{N \times ch1\_avg + ch1(i,j)}{N+1} \quad ch2\_avg = \frac{N \times ch2\_avg + ch2(i,j)}{N+1}$$

    - incrementează N;
  - șterge elementul din poziția *front* a listei

până când lista FIFO devine goală

Se incrementează k și se reia de la pasul 4.1 până când se epuizează toți pixelii din imagine.

Distanța între vectorii de culoare se evaluează prin distanța Euclidiană:

distanța Euclidiană: $\text{dist}_{\text{Euclidian}}(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

5. ***Postprocesare – Eroziune și Dilatare***

După segmentare, pot rămâne pixeli izolați sau zone neetichetate. Pentru corectarea acestor imperfecțiuni, se aplică operații:

- ***Eroziune****: elimină pixelii de la margini (dacă au vecini cu altă etichetă, devin 0).*
- ***Dilatare****: extinde regiunile în jurul pixelilor etichetați (etichetează vecinii neprocesați).*

Aceste operații se aplică în succesiune (2–3 pași de eroziune, apoi dilatare repetată), până când imaginea etichetată este coerentă.

6. ***Colorarea regiunilor – imagine finală***

Pentru a obține o imagine segmentată color, fiecărei regiuni i se atribuie culoarea medie RGB a pixelilor săi (din imaginea originală). Astfel se păstrează coerența vizuală între segmentele rezultate.

c. Ghid de utilizare

1. **Compilare:** Include librăriile OpenCV în proiectul tău C++.
2. **Rulare:** Apelează funcția gaussian_filtered() din main().
3. **Interactiv:** Introdu valoarea value pentru pragul T.

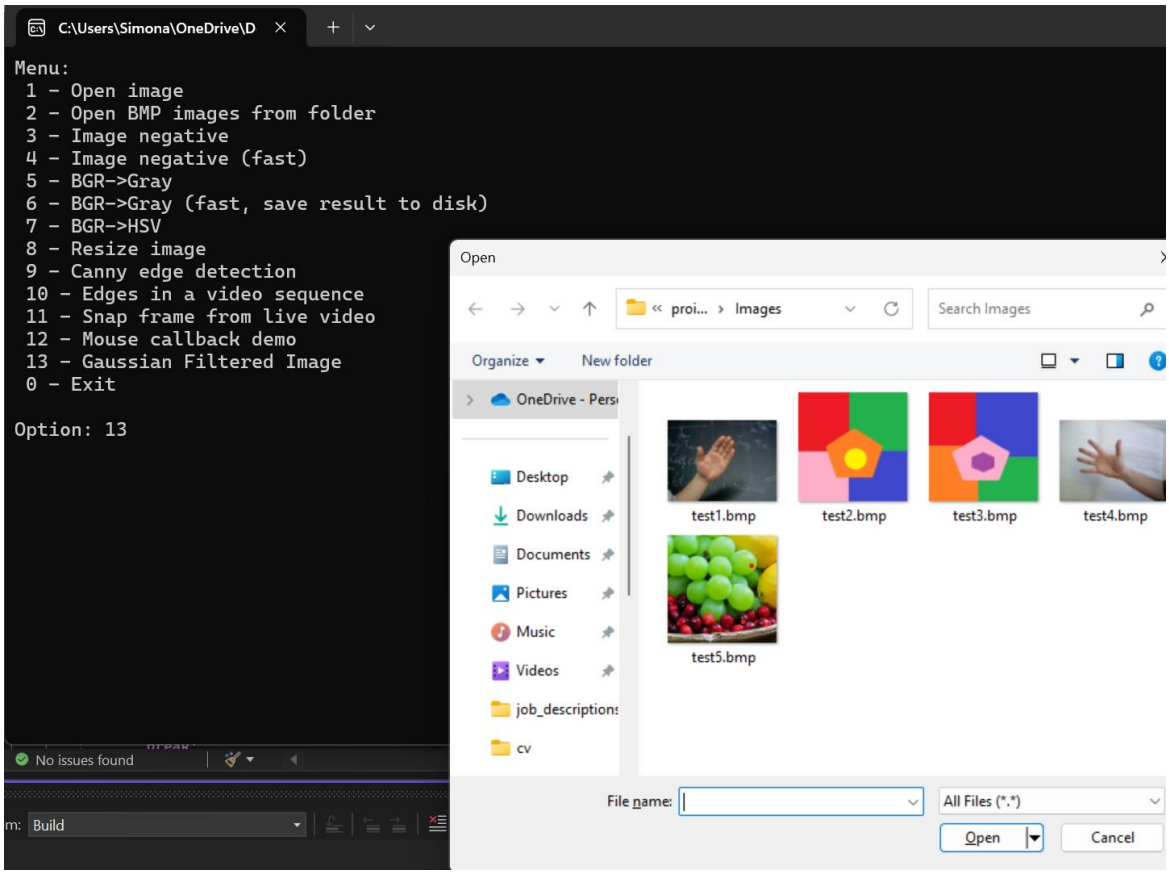
Output:

- Imagine originală
- Imagine filtrată (Gaussian)
- Imagine etichetată (înainte de postprocesare)
- Imagine etichetată (după postprocesare)

4. Rezultate experimentale

4.1. Descrierea setului de date

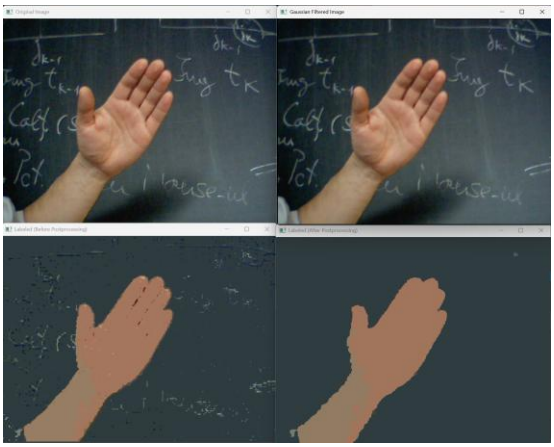
Pentru testarea metodei propuse, au fost utilizate imagini în format .bmp, selectate manual pentru a acoperi o varietate de scene și texturi. Aceste imagini au dimensiuni moderate, adecvate pentru procesare eficientă în timp real și pentru aplicarea tehnicii de *region growing*.

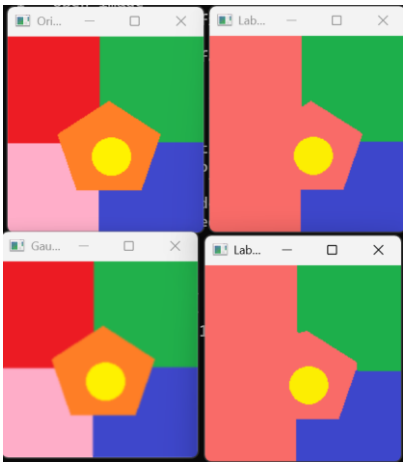


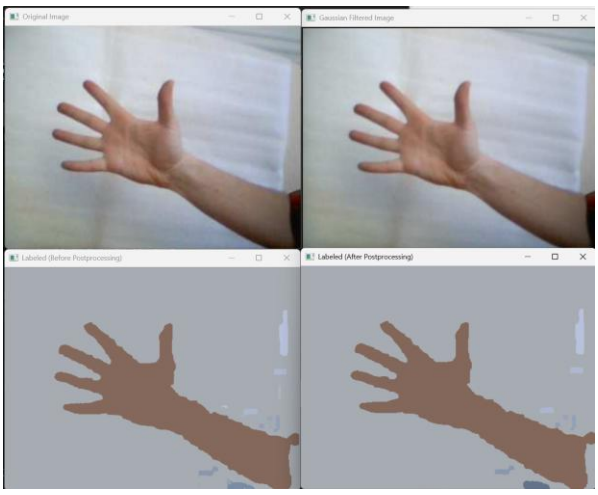
4.2. Etapele de procesare și rezultate vizuale

Pentru fiecare imagine testată, au fost generate patru rezultate intermediare:

- ***Imaginea originală (RGB)*** *– reprezintă punctul de plecare pentru procesare.*
- ***Imaginea filtrată Gaussian*** *– s-a aplicat un filtru de tip low-pass pentru reducerea zgomotului și netezirea tranzițiilor de culoare, ceea ce facilitează segmentarea ulterioară.*
- ***Imagine etichetată înainte de postprocesare*** *– rezultatul algoritmului region growing, care grupează pixelii în regiuni omogene pe baza unei măsuri de disimilaritate în spațiul de culoare Luv*.*
- ***Imagine etichetată după postprocesare*** *– în urma aplicării operațiilor morfologice (eroziune și dilatare), zgomotul a fost eliminat, iar regiunile au fost consolidate, obținându-se o segmentare mai curată și mai coerentă.*







5. Concluzii

- În cadrul acestui proiect a fost implementat un algoritm de segmentare a imaginilor color bazat pe tehnica *region growing*, operând în spațiul de culoare L*u*v*. Algoritmul a fost precedat de o filtrare Gaussiană (5x5) pentru reducerea zgomotului, urmată de conversia în spațiul L*u*v*. Segmentarea a fost realizată utilizând distanța Euclidiană între componentele u* și v*, iar etichetele obținute au fost colorate conform mediei componentelor RGB ale fiecărei regiuni. De asemenea, s-a aplicat o etapă de postprocesare.

- Toate cerințele specificate în temă au fost atinse:
  - filtrarea imaginii originale;
  - conversia și procesarea în spațiul L*u*v*;
  - aplicarea algoritmului de *region growing* cu un prag adaptativ;
  - afisarea segmentării inițiale și postprocesate;
  - respectarea structurii algoritmului și utilizarea metodelor impuse (distanță Euclidiană, etichetare, afișare RGB).

- Rezultatele obținute au demonstrat o segmentare coerentă a regiunilor, în special în imaginile cu contraste u* și v* bine delimitate. În imaginile zgomotoase sau cu variații subtile de culoare, calitatea segmentării a fost îmbunătățită prin filtrarea Gaussiană și postprocesarea morfologică. Parametrul value are o influență directă asupra granularității segmentării – valori mai mici generează segmente mai detaliate, în timp ce valori mari unesc regiuni mai extinse.

Bibliografie

- **Thresholding:**https://www.sciencedirect.com/topics/computer-science/threshold-based-segmentation#:~:text=Threshold%2Dbased%20segmentation%20is%20to,et%20al.%2C%202020**).**
- **Edge-based:**https://www.tandfonline.com/doi/abs/10.1080/02757259209532148
- **Clustering:**https://medium.com/@amit25173/clustering-in-image-processing-explained-2cfc55ccae15

Notă: Nu se va include codul integral în documentație.