

Prăguire - *Thresholding*



1 Introducere

În cursul anterior a fost discutată pe scurt operația de binarizare a unei imagini, operație ce presupune transformarea unei imagini în tonuri de gri, într-o imagine cu doar două valori - alb și negru. Acest lucru se realizează de obicei prin prăguire - *thresholding*, în modul următor: valoarea $f(x, y)$ a fiecărui pixel (x, y) din imagine este comparată cu un prag t . Dacă $f(x, y) \leq t$ atunci pixelul devine negru, altfel pixelul devine alb.

Thresholding-ul este cea mai simplă metodă de segmentare și are ca rol partităionarea în obiecte / regiuni de interes (de obicei pixelii albi) și fundal (de obicei pixelii negri).

Selectarea pragului se poate face interactiv, acesta fiind introdus de către utilizator pe baza percepțiilor sale vizuale sau automat, utilizând proprietățile statistice ale repartiției valorilor de gri în imagine. Pentru stabilirea acestor proprietăți se recurge la histograma imaginii. În general se presupune că, un *vârf* - maxim local - al histogramei corespunde unei structuri în imagine, iar plasarea unui prag de separare se face în *valea* - minim local - dintre două astfel de vârfuri. Această presupunere corespunde realității numai pentru anumite tipuri de imagini, în special cele obținute în condiții controlate de achiziție și iluminare, de exemplu în industrie sau în microscopie. În cazul imaginilor corupte de zgomot sau a celor reprezentând scene complexe din lumea reală această abordare nu conduce la rezultate utilizabile. În unele dintre aceste situații procesarea imaginii prin filtrare sau aplicarea metodei de thresholding asupra gradientului imaginii poate conduce la o segmentare corectă.

Orice metodă automată de thresholding pornește de la anumite presupuneri relativ la proprietățile imaginii, cum ar fi natura bi-modală a histogramei, adică de la existența a două maxime locale bine delimitate.



2 Competențe conferite

La sfârșitul acestui curs, studenții vor fi capabili să înțeleagă:

- Ce este operația de prăguire - *thresholding*
- Cum se pot folosi informațiile statistice oferite de histogramă în stabilirea unui prag de thresholding
- Cum se poate determina un prag global automat pentru o imagine în tonuri de gri.
- Cum se pot calcula praguri adaptabile.



3 Durata medie de studiu individual

Parcurserea de către studenți a acestei unități de învățare se face în 2-3 ore

4 Praguri globale

În cazul imaginilor cu iluminare uniformă tehnica de thresholding presupune stabilirea unui prag global unic, în funcție de care se realizează partaționarea imaginii. Se presupune că imaginea conține o zonă (zone) de interes cu nivelul de gri al pixelilor într-o anumită plajă de valori, plasată pe un fond relativ uniform. Pixelii se partaționează în două clase C_1 reprezentând pixelii zonei de interes și C_2 reprezentând pixelii de fundal.

Câteva dintre cele mai simple metode de selectare a unui prag global automat sunt:

- Pe baza mediei aritmetice: $t = mean(f)$

- Prag **mid-range**:

$$t = round \left(\frac{\max(f) + \min(f)}{2} \right)$$

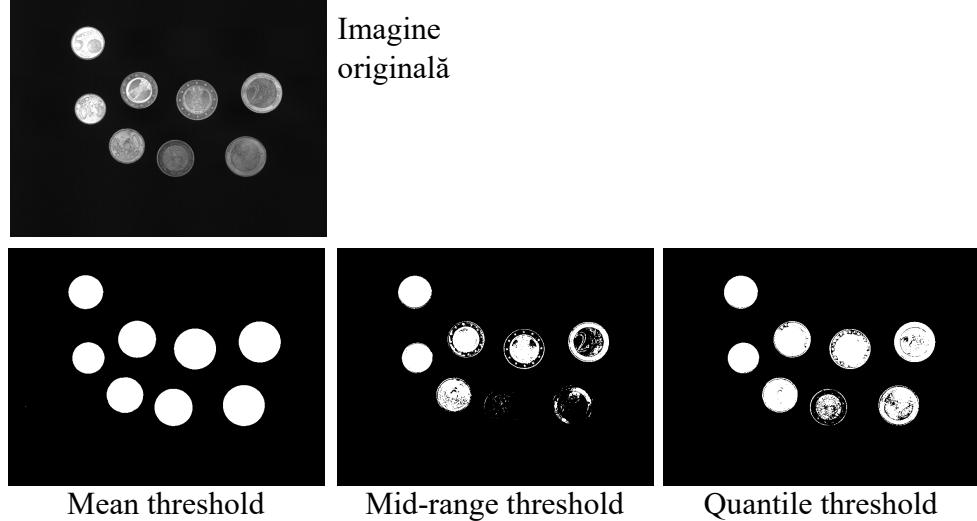
- **Qantile threshold** - presupune faptul ca se poate estima procentul b pixelilor de fundal în raport cu numărul total de pixeli

$$t = min \left\{ j \mid \sum_{k=0}^{k=j} n_k \right\} \geq n * b$$

unde n_k = nr de pixeli de culoarea k , n = nr de pixeli din imagine. Acest tip de binarizare poate fi utilizat, de exemplu, pentru imagini reprezentând text scris pe un fundal uniform.



Exemplu: În figură sunt prezentate exemple de binarizare folosind metodele simple prezentate anterior:



Metodele prezentate mai sus nu țin cont de repartizarea nivelurilor de gri din imagine și dau rezultate satisfăcătoare doar pentru anumite situații, în care delimitarea obiectului / obiectelor de fundal este foarte clară.

De obicei sunt utilizate metode de thresholding care au la bază forma histogramei (de exemplu minimele/ maximele din histogramă) sau informații statistice, obținute tot pe baza histogramei. Pentru a înțelege algoritmii care urmează vom introduce câteva noțiuni statistice de bază precum media și varianța (abaterea medie pătratică). Apoi vor fi prezentate câțiva dintre algoritmii cunoscuți de *thresholding* cu un prag global.

Se consideră un set de date $\{x_1, x_2, \dots, x_k\}$. Atunci:

Media:

$$\mu = \frac{1}{k} \sum_{j=1}^k x_j \quad (1)$$

Abaterea medie pătratică:

$$\sigma^2 = \frac{1}{k} \sum_{j=1}^k (x_j - \mu)^2 \quad (2)$$

Varianța:

$$var = \sigma = \sqrt{\frac{1}{k} \sum_{j=1}^k (x_j - \mu)^2} \quad (3)$$

După cum se poate observa în formule, abaterea medie pătratică se calculează cumulând abaterile fiecărui individ față de media setului de date. Aceste abateri sunt ridicate la

pătrat pentru a avea doar valori pozitive. Apoi suma se împarte la numărul de indivizi din set. Astfel abaterea medie pătratică reflectă cu cât diferă în medie un individ față de media setului, deci oferă informații respectiv la gradul de uniformitate al setului de date. Atunci când abaterea medie pătratică este mică, indivizii sunt foarte asemănători, deci avem un grad ridicat de uniformitate al setului de date. Atunci când abaterea medie pătratică este mare, avem o variație semnificativă între indivizi, deci un grad ridicat de neuniformitate.

Valorile datelor statistice pot fi calculate eficient într-o imagine sau într-un interval al nivelurilor de gri pe baza histogramei relative: $h : \{0, 1, \dots, 255\} \rightarrow [0, 1]$, $h(k) = n_k/n$, unde n_k este numărul de pixeli de culoare k , iar n este numărul total de pixeli din imagine. Funcția h :

- - poate fi considerată funcție de repartiție a nivelurilor de gri
- -notăm $p(k) := h(k)$ - probabilitatea de apariție a unui anumit nivel de gri în imagine

Valorile μ și σ^2 pot fi calculate pe baza histogramei prin:

$$\mu = \frac{\sum_{k=0}^{255} kp(k)}{\sum_{k=0}^{255} p(k)}$$

$$\sigma^2 = \frac{\sum_{k=0}^{255} (k - \mu)^2 p(k)}{\sum_{k=0}^{255} p(k)}$$

Dar $\sum_{k=0}^{255} p(k) = 1$, deci:

$$\mu = \sum_{k=0}^{255} kp(k) \quad (4)$$

$$\sigma^2 = \sum_{k=0}^{255} (k - \mu)^2 p(k) = \sum_{k=0}^{255} k^2 p(k) - \mu^2 \quad (5)$$

Se observă că pe baza formulelor de mai sus ambele valori pot fi calculate pe baza probabilității nivelor de gri printr-o singură trecere prin histograma relativă.

Media și abaterea medie pătratică pot fi calculate pentru un interval al nivelurilor de gri, de exemplu $[t_1, t_2]$, pe baza histogramei relative, prin formulele:

$$\mu_{[t_1, t_2]} = \frac{\sum_{k=t_1}^{t_2} kp(k)}{\sum_{k=t_1}^{t_2} p(k)} \quad (6)$$

$$\sigma_{[t_1, t_2]}^2 = \frac{\sum_{k=t_1}^{t_2} (k - \mu_{[t_1, t_2]})^2 p(k)}{\sum_{k=t_1}^{t_2} p(k)} \quad (7)$$

4.1 Algoritmul *ISODATA clustering - Intermeans*

Un algoritm iterativ clasic și foarte simplu de partitioanarea a imaginii este algoritmul de clustering numit **isodata** sau **intermeans**, prezentat în [1]. Acest algoritm poate fi folosit atât pentru partitioanarea unei imagini în mai multe regiuni (capitolul de segmentare), precum și, într-o variantă simplificată, pentru clasificarea pixelilor în două clase C_1 și C_2 , reprezentând regiunea de interes și fundalul. Această clasificare se face relativ la un prag t în modul următor. Pixelii pentru care valoarea este mai mare decât pragul t sunt considerați din clasa C_1 , iar ceilalți pixeli sunt atribuiți clasei C_2 . Determinarea pragului optimal de partitioanare se face iterativ în modul descris mai jos.

Algoritm

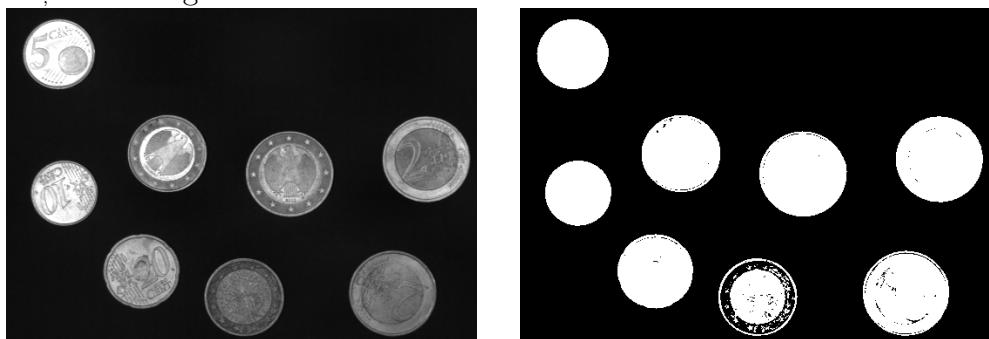
1. Se selectează un prag inițial $t = t_0$ - de exemplu poate fi aleasă ca valoare inițială media nivelurilor de gri sau media între valoarea maximă și valoarea minimă din imagine;
2. Se partitioanează imaginea pe baza pragului t în cele două clase C_1 - toți pixelii $> t$ - și C_2 - toți pixelii $\leq t$;
3. Se calculează mediile μ_1 și μ_2 pentru pixelii aparținând clasei C_1 și respectiv C_2 ;
4. Se calculează noul prag $t = \frac{\mu_1 + \mu_2}{2}$;
5. Pașii 2 - 4 se repetă până când diferența dintre pragurile calculate la două iteratii succesive este suficient de mică relativ la o valoare prestabilită, adică până la stabilizarea claselor.

Calculul valorilor μ_1 și μ_2 se poate realiza eficient prin folosirea histogramei relative și a formulelor 6 și 7 prezentate în secțiunea anterioară.

Un algoritm în pseudo-cod poate fi găsit în [1]



Exemplu de binarizare cu un prag calculat cu algoritmul **isodata**. În stânga este imaginea originală și în dreapta imaginea binarizată. Pragul obținut de algoritm este $t = 83$.



4.2 Prag global optimal - Metoda Otsu

Un prag global optimal este un prag care optimizează o anumită funcție obiectiv, de exemplu minimizarea erorii de clasificare a unui pixel. O metodă de determinare a unui prag global care minimizează această eroare este metoda **Otsu**. Această metodă are la bază presupunerea existenței unei histograme bi-modale a imaginii, pixelii putând fi clasificați în două clase C_1 și respectiv C_2 . Scopul este de a găsi un prag t astfel încât pe de-o parte indivizii din interiorul unei clase să fie cât mai similari, deci varianța pentru clasa respectivă să fie cât mai mică, iar pe de altă parte doi indivizi din clase diferite să fie cât mai diferenți, deci centrele claselor (reprezentate prin valorile medii) să fie cât mai distanțate. Pragul astfel calculat urmărește maximizarea raportului varianță inter-clasă (*between-class variance*) / varianță intra-clasă (*within-class variance*).

Considerăm funcția de probabilitate a nivelurilor de gri (dată prin histograma normalizată) $p : \{0, 1, \dots, 255\} \rightarrow [0, 1]$, $C_1(t)$ și $C_2(t)$ cele două clase de pixeli reprezentând fondul, respectiv obiectul și un prag t , $0 < t < 255$.

Pentru clasa $C_1(t)$	Pentru clasa $C_2(t)$
$P_1(t) = \sum_{k=0}^t p(k)$ = probabilitatea ca un pixel să aparțină clasei $C_1(t)$	$P_2(t) = \sum_{k=t+1}^{255} p(k)$ = probabilitatea ca un pixel să aparțină clasei $C_2(t)$
$\mu_1(t) = \frac{1}{P_1(t)} \sum_{k=0}^t kp(k)$ - media	$\mu_2(t) = \frac{1}{P_2(t)} \sum_{k=t+1}^{255} kp(k)$ - media
$\sigma_1^2(t) = \frac{1}{P_1(t)} \sum_{k=0}^t (k - \mu_1)^2 p(k)$ - varianța	$\sigma_2^2(t) = \frac{1}{P_2(t)} \sum_{k=t+1}^{255} (k - \mu_2)^2 p(k)$ - varianța

Tabela 1: Proprietățile statistice ale celor două clase $C_1(t)$ și $C_2(t)$.

În tabelul 1 sunt definite proprietățile statistice ale celor două clase $C_1(t)$ și $C_2(t)$, calculate folosind relațiile 6 și 7. În plus are loc relația

$$P_1(t) + P_2(t) = 1$$

Varianța inter-clase - *between-class variance* este definită prin [2]:

$$\sigma_B^2(t) = P_1(t)(\mu_1(t) - \mu)^2 + P_2(t)(\mu_2(t) - \mu)^2 \quad (8)$$

În urma unor calcule matematice ecuația se reduce la:

$$\sigma_B^2(t) = P_1(t)P_2(t)(\mu_1(t) - \mu_2(t))^2 \quad (9)$$

unde $\mu = \sum_{k=0}^{255} kp(k)$ este media nivelurilor de gri pentru întreaga imagine.

Varianța intra-clase - *within-class variance* este definită prin (citat Burger):

$$\sigma_W^2(t) = P_1(t)\sigma_1^2(t) + P_2(t)\sigma_2^2(t) \quad (10)$$

Pentru evaluarea calității pragului de separare se utilizează raportul

$$\eta(t) = \frac{\sigma_B^2(t)}{\sigma_W^2(t)}.$$

Un prag optim este acela care maximizează acest raport. Un astfel de prag separă cel mai bine clasele prin maximizarea varianței inter-clasă și minimizarea varianței intra-clasă.

Varianța totală pentru imagine este constantă și este:

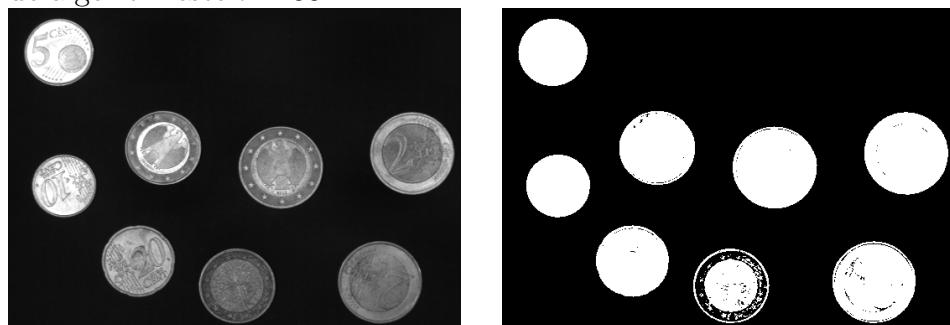
$$\sigma^2(t) = \sum_{k=0}^{255} ((k - \mu)^2 p(k) = \sigma_B^2(t) + \sigma_W^2(t)$$

Se observă că, dacă $\sigma_B^2(t)$ crește atunci $\sigma_W^2(t)$ scade. În această situație, pentru maximizarea factorului $\eta(t)$ este suficientă determinarea unui prag t care să maximizeze $\sigma_B^2(t)$.

În practică se determină valoarea lui $\sigma_B^2(t)$ pentru fiecare prag $0 < t < 255$ și se selectează acel prag t care maximizează pe $\sigma_B^2(t)$. Dacă maximul se obține pentru mai mult de o singură valoare, atunci pragul optim se consideră a fi media diferențelor valori pentru care s-a obținut acest maxim.



Exemplu de binarizare cu un prag calculat cu metoda **Otsu**. În stânga este imaginea originală și în dreapta imaginea binarizată. Pragul obținut de algoritm este $t = 83$.



Se observă faptul că acest algoritm a obținut același prag ca algoritmul isodata.

Observație Algoritmul Otsu produce în cazul histogramelor bimodale rezultate similare cu algoritmul intermeans/isodata. În situația unor histograme uni- sau multimodale, rezultatele pot fi diferite, precum se poate vedea din următorul exemplu.

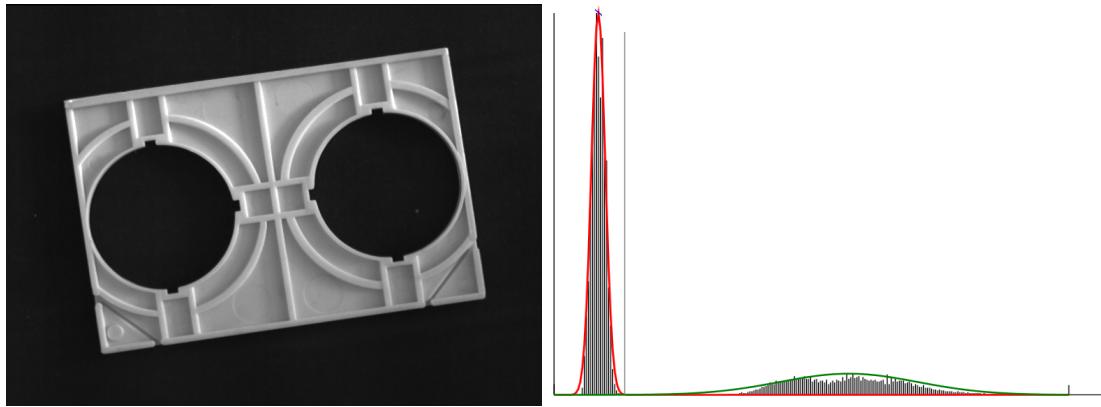
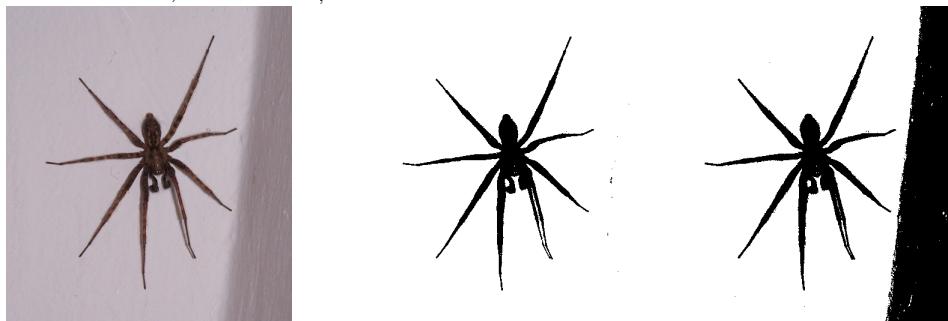


Figura 1: Imagine în tonuri de gri reprezentând un obiect de culoare deschisă pe un fundal negru, împreună cu histograma imaginii.



Exemplu comparativ binarizare *intermeans* versus *Otsu*. În stânga este imaginea originală, în centru rezultatul binarizării cu *intermeans* și în dreapta rezultatul binarizării cu *Otsu*. Pragul obținut de *intermeans* este $t = 118$, iar cel obținut de *Otsu* este $t = 150$.



Se observă faptul că în acest caz (histograma este trimodală) cei doi algoritmi determină rezultate diferite.

4.3 Algoritmul pragului de eroare minimă - *Minimum Error Threshold*

Acest algoritm prezentat pe larg în [1] pornește de la presupunerea cele două clase C_1 și C_2 pot fi reprezentate prin 2 repartiții gaussiene. Un exemplu este prezentat în figura 1. Se observă faptul că histograma este bimodală. Unul dintre vârfuri (cel mai mic) reprezintă pixelii obiectului, iar celălalt reprezintă pixelii fundalului aproape negru. Cu roșu respectiv verde au fost trasate cele două gaussiene corespunzătoare celor două clase.

Din faptul că se presupune o repartitie gaussiană pentru fiecare dintre cele două clase ajungem la: $P(k|C_i)$ = probabilitatea ca valoarea k să apară în clasa C_i e dată de repartitia

Gauss

$$P(k|C_i) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(k-\mu_i)^2}{2\sigma_i^2}} \quad (11)$$

unde μ_i - media valorilor și σ_i^2 - varianța din clasa C_i , $i = 1, 2$.

Binarizarea cu acest algoritm are ca scop clasificarea unui pixel cu valoarea $k \Rightarrow$ într-una din clasele C_1 (obiect) sau din C_2 (fundal). Această clasificare se poate face pe baza unei **decizii Bayesiene** [1] în modul următor: dacă $P(C_1|k) > P(C_2|k) \Rightarrow$ atribui k clasei C_1 , altfel atribui clasei C_2 , unde:

- $P(C_1|k)$ - probabilitatea ca să fie clasa C_1 , dacă s-a găsit valoarea k
- $P(C_2|k)$ - probabilitatea ca să fie clasa C_2 , dacă s-a găsit valoarea k

Acest mod de decizie minimizează riscul de a face o clasificare greșită. Din teoria probabilităților, aceste probabilități condiționale pot fi scrise:

$$P(C_i|k) = \frac{P(k|C_i)P(C_i)}{p(k)}$$

Deci inegalitatea $P(C_1|k) > P(C_2|k)$ devine $P(k|C_1)P(C_1)/p(k) > P(k|C_2)P(C_2)/p(k)$. Cum $p(k)$ este termen comun poate fi ignorat din formuă și înlocuind $P(k|C_i)$ cu expresia în ecuația 11 obținem:

$$P(k|C_i)P(C_i) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(k-\mu_i)^2}{2\sigma_i^2}} P(C_i) \quad (12)$$

Algoritmul urmărește să obțină valori cât mai mari ale probabilității $P(C_1|k)$ pentru pixeli k din clasa C_1 și cât mai mici pentru cei din clasa C_2 și pentru $P(C_2|k)$ valori cât mai mari pentru pixeli k din clasa C_2 și cât mai mici pentru cei din clasa C_1 .

Pentru a simplifica ecuația 12 putem să logaritmăm, deoarece funcția logaritm este crescătoare și nu modifică inegalități. Obținem:

$$\begin{aligned} \ln(P(k|C_i)P(C_i)) &= \ln\left(\frac{1}{\sqrt{2\pi}\sigma_i}\right) + \ln\left(e^{-\frac{(k-\mu_i)^2}{2\sigma_i^2}}\right) + \ln(P(C_i)) = \\ &= -\frac{1}{2} \left[\ln(2\pi) + \frac{(k-\mu_i)^2}{\sigma_i^2} + \ln(\sigma_i^2) - 2\ln(P(C_i)) \right] \end{aligned} \quad (13)$$

Valoarea probabilității $P(C_i|k)$ trebuie să fie cât mai mare pentru un pixel k din clasa C_i , rezultă că și valoarea expresiei din 13 trebuie să fie cât mai mare pentru pixelii din C_i . Deoarece termenul $\ln(2\pi)$ este constant, nu influențează maximizarea expresiei și poate fi ignorat. În plus problema de maximizare devine o problemă de minimizare prin definirea erorii:

$$\epsilon_i(k) = \frac{(k-\mu_i)^2}{\sigma_i^2} + \ln(\sigma_i^2) - 2\ln(P(C_i)) \quad (14)$$

unde $\epsilon_i(k)$ este măsură potențială a erorii de clasificare a valorii observate k în clasa C_i [1].

Valorile $\mu_1, \mu_2, \sigma_1, \sigma_2$ sunt determinate de pragul de binarizare t , care împarte valorile de gri în clasele C_1 și C_2 , deci de fapt sunt $\mu_1(t), \mu_2(t), \sigma_1(t), \sigma_2(t)$ și deci și $\epsilon_i(k)$ este de fapt $\epsilon_i(k, t)$:

$$\epsilon_i(k, t) = \frac{(k - \mu_i(t))^2}{\sigma_i^2(t)} + \ln(\sigma_i^2(t)) - 2\ln(P(C_i(t))) \quad (15)$$

Calitatea clasificării pe baza pragului t poate fi estimată pe baza funcției

$$e(t) = \sum_{k=0}^t p(k)\epsilon_1(k, t) + \sum_{k=t+1}^{255} p(k)\epsilon_2(k, t) \quad (16)$$

Dar pentru pragul t , clasele C_1 și C_2 sunt determinate de t și la fel ca în tabelul de la Otsu $P(C_1) = \sum_{k=0}^t p(k) = P_1(t)$ și $P(C_2) = \sum_{k=t+1}^{255} p(k) = P_2(t)$.

Din calcule matematice, formula 16 de mai sus devine:

$$e(t) = 1 + P_1(t) \ln(\sigma_1^2(t)) + P_2(t) \ln(\sigma_2^2(t)) - 2P_1(t) \ln(P_1(t)) - 2P_2(t) \ln(P_2(t)) \quad (17)$$

care trebuie minimizată.

Algoritm: pentru determinarea pragului de eroare minimă

Intrare: Histograma unei imagini în tonuri de gri

Iesire: Pragul t de binarizare

Initializează ϵ_{min}

$t_{min} \leftarrow 0$

pentru $t = 0, 255$ **executa**

calculează $P_1, P_2 = 1 - P_1, \mu_1, \mu_2, \sigma_1, \sigma_2$

pe baza histogramei relative

calculează $\epsilon = 1 + P_1 \ln(\sigma_1^2) + P_2 \ln(\sigma_2^2) - 2P_1 \ln(P_1) - 2P_2 \ln(P_2)$

daca $\epsilon < \epsilon_{min}$ **atunci**

$\epsilon_{min} \leftarrow \epsilon$

$t_{min} \leftarrow t$

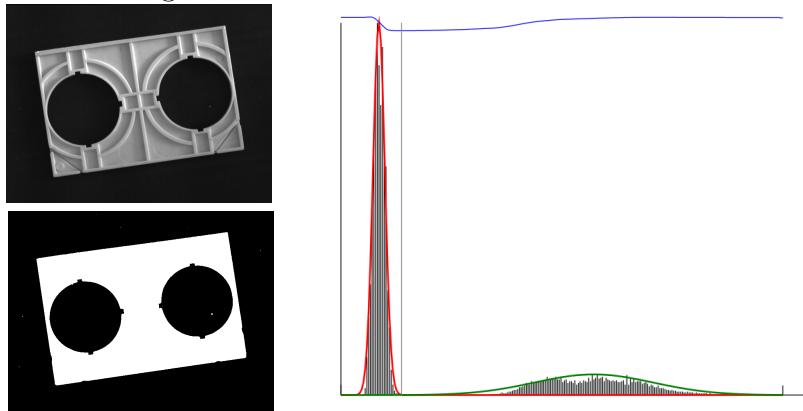
sfarsit_daca

sfarsit_for

return t_{min}



Exemplu binarizare cu algoritmul *minimum error thresholding*. În stânga sunt imaginea originală și rezultatul binarizării. În dreapta este histograma imaginii, împreună cu graficele celor două gaussiene care reprezintă repartițiile claselor C_1 și C_2 și graficul funcției de eroare (albastru). Pragul obținut este valoarea indicată de dreapta verticală de culoare neagră.



4.4 Praguri multiple

În cazul histogramelor multi-modale - care au mai mult de două maxime locale distincte - se pot utiliza mai multe praguri pentru separarea zonelor de interes. Informații detaliate pot fi găsite în literatura de specialitate. De exemplu metoda Otsu poate fi extinsă pentru un număr variabil - $(K - 1)$ - de praguri: t_1, t_2, \dots, t_{K-1} care determină K clase C_1, C_2, \dots, C_K

Varinația inter-clasă devine:

$$\sigma_B^2 = \sum_{i=1}^K P_i(\mu_i - \mu)^2$$

unde C_i = clasa pixelilor din intervalul $(t_{i-1}, t_i]$

$$P_i = \sum_{k=t_{i-1}+1}^{t_i} p(k)$$

și

$$\mu_i = \frac{1}{P_i} \sum_{k=t_{i-1}+1}^{t_i} kp(k)$$

Se vor alege acele praguri $t_1 < t_2 < \dots < t_{K-1}$ care maximizează σ_B .

Exemplu pentru 3 clase cu 2 praguri t_1 și t_2 . Varinăța inter-clasă devine:

$$\sigma_B^2 = P_1(\mu_1 - \mu)^2 + P_2(\mu_2 - \mu)^2 + P_3(\mu_3 - \mu)^2$$

Pentru clasa C_1	Pentru clasa C_2	Pentru clasa C_3
$P_1 = \sum_{k=0}^{t_1} p(k)$	$P_2 = \sum_{k=t_1+1}^{t_2} p(k)$	$P_3 = \sum_{k=t_2+1}^{255} p(k)$
$\mu_1 = \frac{1}{P_1} \sum_{k=0}^{k_1} kp(k)$	$\mu_2 = \frac{1}{P_2} \sum_{k=k_1+1}^{k_2} kp(k)$	$\mu_3 = \frac{1}{P_3} \sum_{k=k_2+1}^{255} kp(k)$

Se vor alege acele praguri $0 < k_1 < k_2 < 255$ care maximizează σ_B

Algoritm: Otsu cu 2 praguri pentru 3 clase.

Intrare: Histograma unei imagini în tonuri de gri

Iesire: Pragurile t_1 și t_2

$\sigma_B_max \leftarrow 0$

$t_1 \leftarrow 0$

$t_2 \leftarrow 1$ pentru $k_1 = 0, 253$ executa

pentru $k_2 = k_1 + 1, 254$ executa

calculează $P_1, P_2, P_3, \mu_1, \mu_2, \mu_3, \sigma_B$

pe baza histogramei relative

dacă $\sigma_B > \sigma_B_max$ atunci

$\sigma_B_max \leftarrow \sigma_B$

$t_1 \leftarrow k_1$ $t_2 \leftarrow k_2$

sfarsit_daca

sfarsit_for

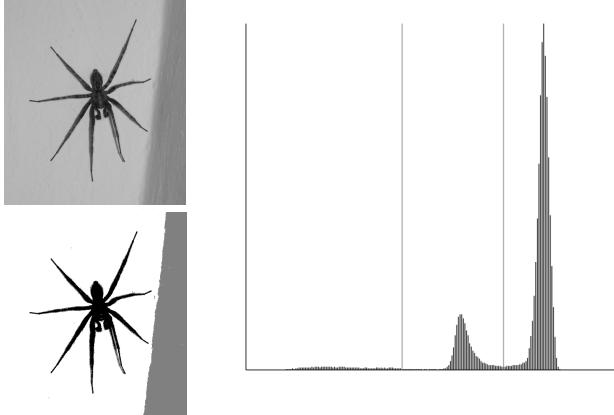
sfarsit_for

return t_1 și t_2



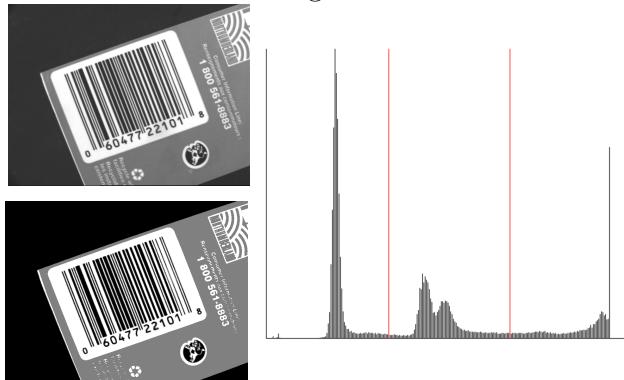
Exemple În figură se poate vedea două exemple ale aplicării algoritmului Otsu cu două praguri.

Imaginea 1:



$$t_1 = 97, t_2 = 160$$

Imaginea 2:



$$t_1 = 91, t_2 = 181$$

4.5 Extragerea fundalului - metoda triunghiului

Metoda triunghiului se poate aplica cu succes atunci când un procent mare din pixeli aparțin zonei de fundal. Poate fi vorba de fundal de culoare închisă sau de culoare deschisă. Ideea algoritmului este aceea, de a plasa un prag imediat după vârful din histogramă corespunzător pixelilor de fundal. O exemplificare schematică este prezentată în figura 2.

Etapele algoritmului sunt:

- Se caută valoarea maximă din histogramă
- Se caută o valoarea minimă cât mai departe de poziția maximului (poate fi de o parte sau de alta a maximului).
- Se consideră latura D care ”unește” maximul de minim
- Se alege ca prag t acea valoare k pentru care se obține cea mai mare distanță dintre $H[k]$ și $D + offset$

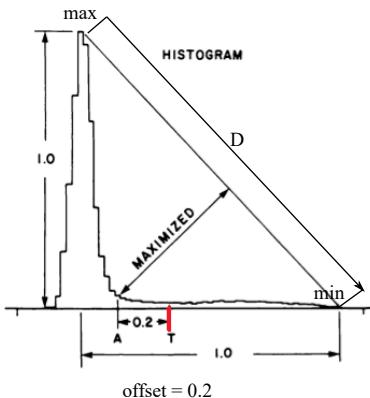


Figura 2: Pragul se stabilește pe baza maximului histogramei



To Do:

- Implementați unul dintre algoritmii de *thresholding* în *framework*-ul de laborator.
- Căutați în documentația online funcțiile de binarizare din OpenCv pentru algoritmii prezenți la curs. Studiați parametrii și verificați cum funcționează pe imagini în tonuri de gri.



Să ne reamintim

- Binarizarea unei imagini cu un prag global t se realizează comparând fiecare pixel din imagine cu t . Dacă valoarea pixelului este mai mare ca t , atunci devine alb, altfel negru. Evident se poate face acest lucru și complementar.
- Multe dintre metodele uzuale de binarizare automată cu un prag global folosesc informațiile statistice calculate pe baza histogramei.
- Există și metode de binarizare bazate pe forma histogramei (metoda triunghiului), care țin cont de maxime și minime din histogramă.
- Binarizarea prin folosind un prag se numește *prăguire - thresholding*.
- *Thresholding*-ul este cea mai simplă metodă de segmentare a unei imagini și este utilizabilă în contexte foarte simple, când se urmărește separarea unuia sau a mai multor obiecte aflate pe un fundal relativ uniform

5 Praguri variabile (adaptable)

Factori precum iluminarea neuniformă pot avea ca rezultat imposibilitatea segmentării corecte utilizând un prag global unic. În această situație, se pot calcula praguri diferite pentru regiuni diferite în imagine.

O abordare în acest sens ar fi partităionarea imaginii în subimagini mai mici care nu se suprapun și determinarea unui prag separat pentru fiecare subimagine.

O altă metodă este aceea a pragului local adaptiv, care pentru fiecare pixel (x, y) din imagine determină un prag nou în funcție de proprietățile locale - media și varianța - sau de valorile extreme - minim și maxim - într-o vecinătate dată a acelui pixel.

Considerând o vecinătate S_{xy} a unui pixel (x, y) media și varianța sunt date de:

$$\mu_{xy} = \sum_{(s,t) \in S_{xy}} f(s, t) h_{xy}(f(s, t))$$

$$\sigma_{xy}^2 = \sum_{(s,t) \in S_{xy}} (f(s, t) - \mu_{xy})^2 h_{xy}(f(s, t))$$

$f(s, t)$ = nivelul de gri al pixelului (s, t) , $h_{xy}(f(s, t))$ = histograma normalizată a vecinătății S_{xy} .

Pentru calculul pragului local adaptiv pot fi utilizate diferite formule, de exemplu:

$$t_{xy} = a\sigma_{xy} + b\mu_{xy} \quad (18)$$

$$t_{xy} = a\sigma_{xy} + b\mu \quad (19)$$

unde μ reprezintă media nivelului de gri pe întreaga imagine, a și b reprezintă constante nenule.

$$t_{xy} = a \max_{xy} + (1 - a) \min_{xy} \quad (20)$$

unde \max_{xy} și \min_{xy} reprezintă valorile de gri maximă și respectiv minimă pentru vecinătatea S_{xy} .

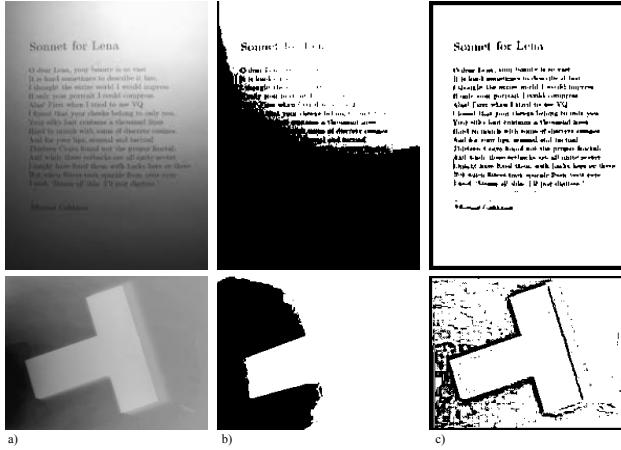
Expresia 18 poate fi simplificată, pentru a crește eficiența de calcul, eliminând primul termen. se obține:

$$t_{xy} = b\mu_{xy} \quad (21)$$

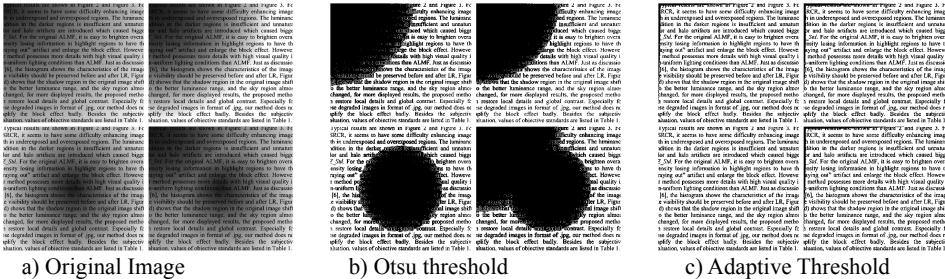
Se observă faptul că, în cazul ecuației 18, în regiunile relativ uniforme varianța se apropie de 0. Dacă $b < 1$ atunci pragul este sub valoarea medie și deoarece zona este relativ uniformă pixelii vor fi setați la valoarea 255, dacă $b > 1$ atunci pragul va fi peste medie și majoritatea pixelilor vor fi setați la 0. Acest lucru are drept consecință punerea în evidență a contururilor asa cum reiese din imaginile prezentate în exemple.



Exemplu de binarizare cu prag adaptabil după formula 18 cu $a = 0.5$, $b = 0.9$ pentru prima imagine și $a = 0.4$, $b = 0.98$ pentru a doua imagine. În stânga sunt imaginile originale, în mijloc rezultatele binarizării cu Otsu, iar în dreapta rezultatele binarizării cu prag adaptabil.



Exemplu de binarizare cu prag adaptabil după formula 21 cu $b = 0.8$. În stânga este imaginea originală, în mijloc rezultatul binarizării cu Otsu, ar în dreapta este rezultatul binarizării cu prag adaptabil.



Calculul mediei și varianței în vecinătatea fiecărui pixel presupune complexitate crescută și deci un algoritm lent. Pentru a eficientiza acest algoritm de *thresholding* adaptiv, de obicei se consideră $a = 0$, iar $b < 1$, adică formula:

$$t_{xy} = b\mu_{xy}.$$

Cu toate acestea complexitatea unui astfel de algoritm este de ordinul $\Theta(M \times N \times k^2)$, unde $M \times N$ este dimensiunea imaginii, iar k timesk dimensiunea vecinătății luat în considerare.

Calcul rapid al mediei într-o vecinătate - Imaginea integrală

Pentru calculul mediei în vecinătatea unui pixel x , μ_{xy} , se utilizează *imaginea integrală*, care permite reducerea sumei în fereastra la 4 operații aritmetice.

În continuare vom defini imaginea integrală și vom descrie modul de calcul al mediei unei regiuni dreptunghiulare din imagine.

Considerând o imagine $f(x, y)$, imaginea integrală în punctul (x, y) , $I(x, y)$ este definită în modul următor:

$$I(x, y) = \sum_{s=0}^x \sum_{t=0}^y f(s, t).$$

Această valoare poate fi calculată în fiecare punct (x, y) pe baza valorilor calculate anterior $I(x - 1, y - 1)$, $I(x, y - 1)$ și $I(x - 1, y)$:

$$I(x, y) = \begin{cases} f(0, 0), & \text{pentru } x = 0, y = 0 \\ I(0, y - 1) + f(0, y), & \text{pentru } x = 0 \\ I(x - 1, 0) + f(x, 0), & \text{pentru } y = 0 \\ I(x - 1, y) + I(x, y - 1) - I(x - 1, y - 1) + f(x, y), & \text{altfel} \end{cases}$$



Exemplu de calcul al imaginii integrale.

1	2	2	4	3	2	4
4	4	1	2	3	4	4
2	2	2	3	1	1	2
1	2	1	1	3	3	4
3	2	2	3	3	1	2
3	3	3	4	3	3	4
1	2	1	1	2	2	4

Imaginea originală

1	3	5	9	12	14	18
5	11	14	20	26	32	40
7	15	20	29	36	43	53
8	18	24	34	44	54	68
11	23	31	44	57	68	84
14	29	40	57	73	87	107
15	32	44	62	80	96	120

Imaginea integrală

Suma valorilor pixelilor dintr-o fereastră dreptunghiulară din imagine, delimitată de colțul stânga sus (x_0, y_0) și colțul dreapta jos (x_1, y_1) poate fi calculată astfel:

$$\text{sum}(x_0, y_0, x_1, y_1) = \begin{cases} A, & \text{pentru } x_0 = 0, y_0 = 0 \\ A - B, & \text{pentru } y_0 = 0 \\ A - D, & \text{pentru } x_0 = 0 \\ A + C - B - D, & \text{altfel} \end{cases}$$

unde $A = I(x_1, y_1)$, $B = I(x_0 - 1, y_1)$, $C = I(x_0 - 1, y_0 - 1)$, $D = I(x_1, y_0 - 1)$. Aceste operații sunt ilustrate în exemplul următor.

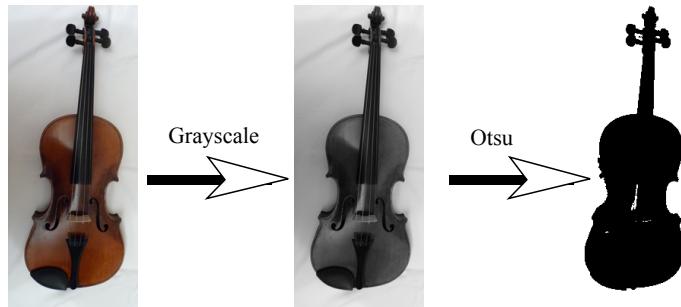


Figura 3: Binarizarea unei imagini color conținând un obiect închis la culoare pe un fundal negru prin aplicarea metodei Otsu asupra imaginii desaturate, adică transformată în imagine *grayscale*



Exemplu de calcul al sumei pixelilor într-un dreptunghi din imagine pe baza imaginii integrale.

Imaginea originală			Imaginea integrală			
	x_0	x_1		x_0	x_1	
y_0	1 2 2 4 3 2 4 4 4 1 2 3 4 4	1 3 5 9 12 14 18 5 11 14 20 26 32 40	y_0	7 15 20 29 36 43 53 8 18 24 34 44 54 68	y_0	C D
y_1	2 2 2 3 1 1 2 1 2 1 1 3 3 4 3 2 2 3 3 1 2 3 3 3 4 3 3 4 1 2 1 1 2 2 4	11 23 31 44 57 68 84 14 29 40 57 73 87 107 15 32 44 62 80 96 120	y_1	B A		
sum = 19			sum = A + C - B - D = 57 + 11 - 23 - 26 = 19			

6 Binarizarea imaginilor color

Binarizarea unei imagini color are în principiu același scop ca binarizarea imaginilor în tonuri de gri: obținerea unui obiect / zone de interes. Modul în care se realizează acest lucru depinde însă de tipul de imagine și/sau de proprietățile de culoare ale obiectului / zonei de interes. De exemplu, în cazul unui obiect de culoare închisă pe un fundal de culoare deschisă, cel mai simplu este să se transforme imaginea color într-o imagine în tonuri de gri, de exemplu prin formula:

$$g(x, y) = 0.5 * R(x, y) + 0.95 * G(x, y) + 0.11 * B(x, y),$$

după care se aplică o operație de thresholding. Un astfel de exemplu este prezentat în figura 3.

În numeroase situații această abordare nu conduce la rezultate utilizabile. De exemplu atunci când ne interesează obiecte de o anumătă culoare (semne de circulație, culoarea pielii etc.). Un astfel de exemplu este prezentat în fig 4.

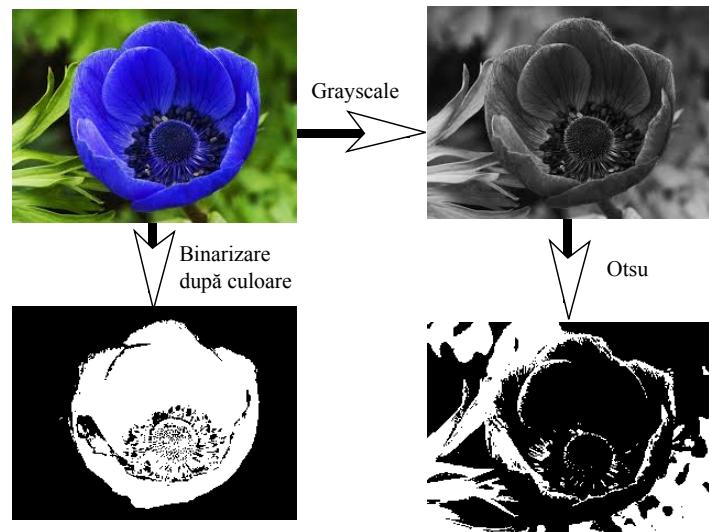


Figura 4: Binarizarea unei imagini color după culoare.

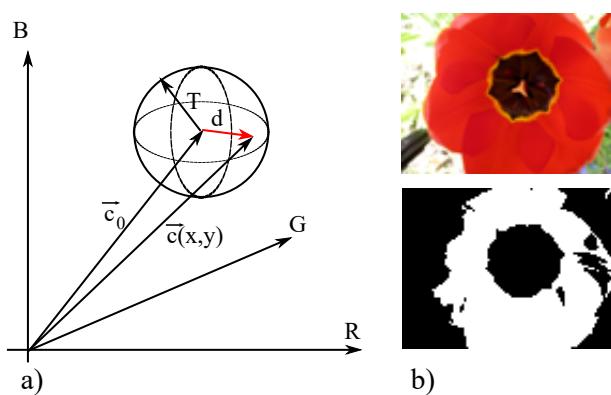


Figura 5: Binarizarea 3D a imaginilor color: a) Culori similare; b) Exemplu de binarizare a unei imagini color.

6.1 Culori asemănătoare în cubul RGB

În cazul binarizării după culoare, se va considera o culoare de referință $c_0 = (R_0, G_0, B_0)$, față de care se va efectua binarizarea și anume, pixelii **asemănători** cu c_0 vor deveni albi, iar ceilalți vor deveni 0. Problema care se pune este, ce înseamnă că două culori sunt asemănătoare?

Considerând cubul de culoare, o anumită culoare dată este un vector de trei componente. În acest spațiu tridimensional, doi vectori se consideră similari, dacă distanța dintre ei este mai mică decât un anumit prag dat (fig. 5 a.).

Binarizarea față de culoarea de referință c_0 , considerând un prag dat t , se realizează în modul următor: se consideră fiecare pixel (x, y) cu culoarea $c(x, y) = (R(x, y), G(x, y), B(x, y))$. Dacă această culoare se află la distanță cel mult t , față de c_0 , atunci pixelul va deveni alb (pixel de interes) altfel va deveni negru. Se consideră distanța euclidiană:

$$dist(c(x, y), c_0) = \sqrt{(R(x, y) - R_0)^2 + (G(x, y) - G_0)^2 + (B(x, y) - B_0)^2} \leq t.$$

Un exemplu de binarizare prin această metodă este prezentat în figura 5 b. Trebuie observat faptul că, dacă doi pixeli au același ton de culoare (*hue*), dar luminozitate foarte diferită, atunci, în cazul binarizării 3D, nu vor fi considerați ca fiind asemănători. Dacă se dorește selectarea unei anumite culori, indiferent de luminozitate, este necesară o altfel de abordare.

6.2 Diagrama cromatică

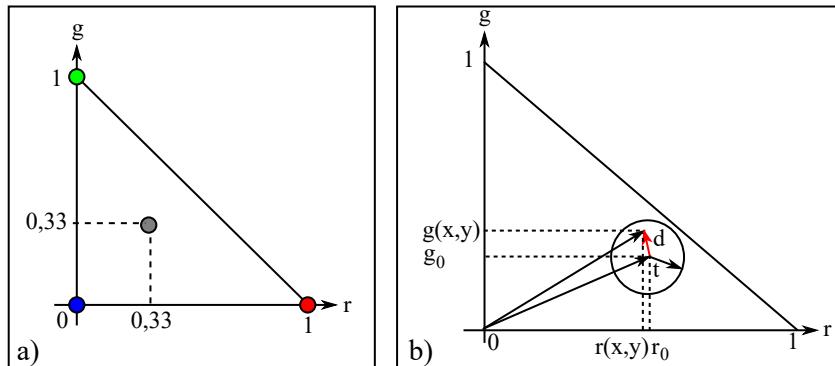


Figura 6: Diagrama cromatică: a) Reprezentare, punctul gri este punctul pe care se proiectează toate nuanțele de gri, inclusiv alb și negru. b) Culori asemănătoare considerate într-un disc de rază d în jurul culorii de referință.

Una dintre posibilitățile binarizării față de o culoare de referință, fără a ține cont de luminozitate, este aceea de a proiecta toate culorile cubului 3D pe un plan 2D. Acest lucru se poate realiza prin raportarea fiecărui canal de culoare la luminozitatea culorii și anume, pentru o culoare (R, G, B) se vor calcula valorile normalizate:

$$r = \frac{R}{R + G + B}$$

$$g = \frac{G}{R + G + B}$$

$$b = \frac{B}{R + G + B} = 1 - r - g$$

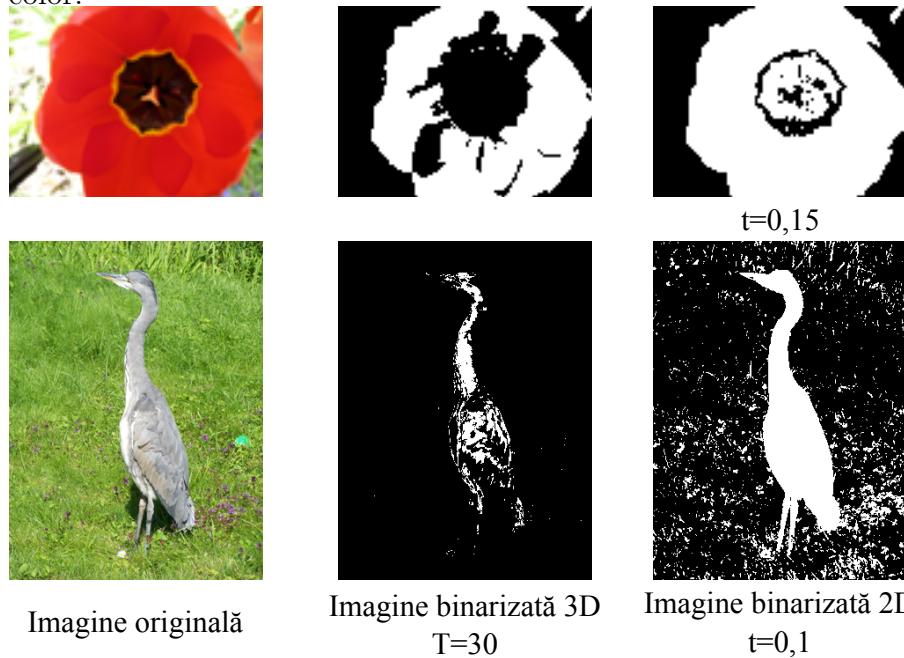
Se observă faptul că doar r și g sunt independente, b putând fi calculat din cele două valori r și g . Fiind dată o culoare de referință normalizată $c_0 = (r_0, g_0)$, o altă culoare $c = (r, g)$ se consideră asemănătoare cu c_0 , dacă distanța d între cele două culori este mai mică decât un prag t :

$$d := \sqrt{(r - r_0)^2 + (g - g_0)^2} \leq t$$

În fig.6 este reprezentată diagrama cromatică, precum și discul culorilor asemănătoare cu o culoare de referință c_0 . Se observă faptul că, în această abordare, toate nuanțele de gri se proiectează în diagrama cromatică pe același punct, deci sunt considerate aceeași culoare, reprezentată prin cercul gri din figură. Se observă faptul că, toate nivelurile de gri sunt proiectate într-un singur punct din diagrama cromatică. Acest mod de reprezentare nu permite distincția dintre alb și negru.



Exemplu comparativ de binarizare în 3D și în 2D pentru două imagini color.



6.3 Binarizarea în alt spațiu de culoare

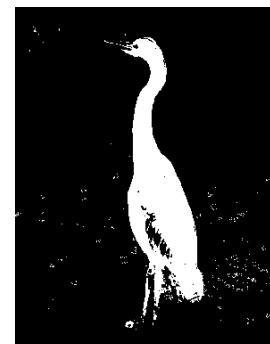
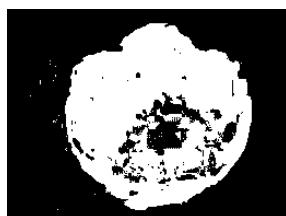
O metodă frecvent folosită de binarizare pe baza culorii este aceea în spațiul de culoare HSV, deoarece componenta H - Hue reprezintă tonul culorii, indiferent de luminozitate și saturatie. Astfel asemănarea dintre culori va fi dată doar de tonul culorii.

Această metodă presupune transformarea imaginii din RGB în spațiul HSV, iar apoi compararea tuturor valorilor de *Hue* din imagine cu valoarea de referință.

Un exemplu de binarizare în spațiul de culoare HSV este prezentat în figura din exemplu. Se consideră componenta *Hue* în grade cu valori între $[0^0, 360^0]$, unde roșu pur este considerat la 0^0 (dar și la 360^0 !)



Exemplu de binarizare în spațiul de culoare HSV.



To Do:

- Implementați unul dintre algoritmii de *thresholding* cu un prag local sau pentru imagini color în *framework*-ul de laborator.
- Căutați în documentația online funcțiile de binarizare din OpenCv pentru algoritmii de *thresholding* adaptiv.



Să ne reamintim

- Binarizarea unei imagini în tonuri de gri cu un prag adaptiv se realizează calculând un prag nou pentru fiecare pixel din imagine. Aceste praguri locale se calculează în funcție de proprietățile statistice ale imaginii.
- Rezultatele binarizării cu prag adaptiv diferă de rezultatele binarizării cu un prag global.
- Pentru eficientizarea calculelor se recomandă folosirea unei imagini integrale.
- Binarizarea unei imagini color se realizează în funcție de scopul urmărit.
- Asemănarea dintre culori poate fi considerată în 3D prin distanță Euclidiană sau în 2D folosind diagrama cromatică.
- Binarizarea imaginilor color poate fi făcută cu succes într-un spațiu de culoare perceptual (de exemplu HSV)



Rezumat: în acest capitol

- a fost discutată problema partiționării unei imagini în zonă de interes (obiect) și fundal cu ajutorul operației de prăguire - *thresholding*
- au fost prezentate câteva dintre cele mai cunoscute și utilizate metode de *thresholding* automat cu un prag global
- au fost prezentate metode de binarizare cu un prag adaptiv
- s-a discutat problema binarizării imaginilor color



Test de autoevaluare:

Bibliografie

- [1] Burger W, Burger M.J „Principals of Digital Image Processing. Core Algorithms”, Springer 2009
- [2] Gonzalez R C; Woods R.E, „Digital Image Processing, global edition”, Perason 2018