Soft computing

Jednostavne operacije za obradu slike

Uklanjanje šuma
Povećanje kontrasta
Morfološke operacije

Segmentacija
Pronalaženje povezanih regiona

Jednostavne operacije nad slikama

- Slike smo reprezentovali kao matrice brojeva

- Ovo nam omogućava jednostavne operacije nad slikama:
  - Sabiranje
  - Oduzimanje
  - Logičke operacije
  - Afine transformacije

- Ove operacije su veoma jednostavne i brze, a imaju širok spektar primena

Jednostavne operacije nad slikama

- Dve ulazne slike
- Operator se primenjuje na odgovarajuće parove piksela ulaznih slika (*pairwise*)

```
for all pixel positions x, y:
  out[x,y] = func( image1[x,y] , image2[x,y]
)
```

- Ulazne slike moraju biti iste veličine

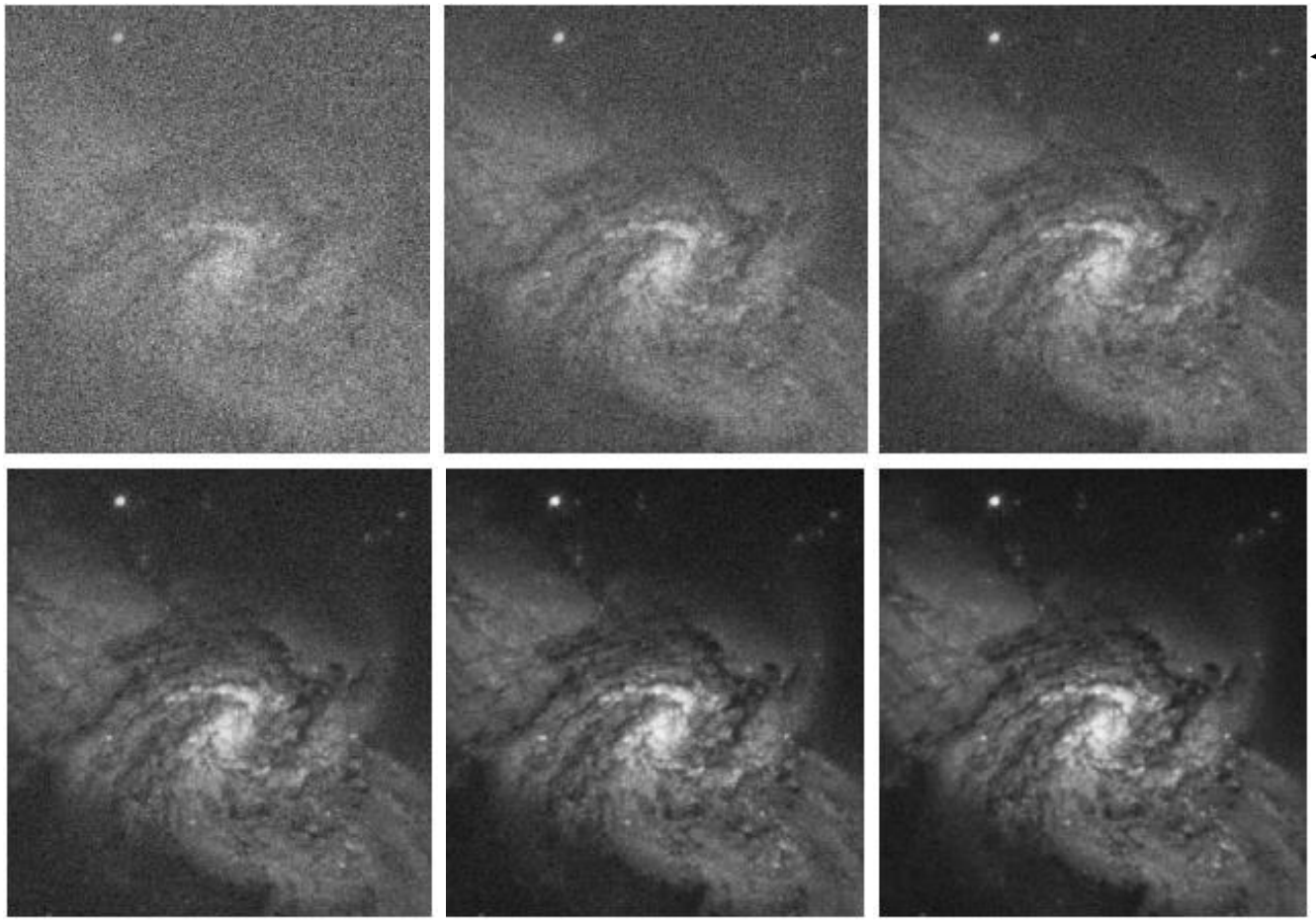
Sabiranje

$$g(x,y) = f_1(x,y) + f_2(x,y)$$

- *Weighted blend:* $g(x,y) = \alpha_1 f_1(x,y) + \alpha_2 f_2(x,y)$

Sabiranje

- NASA je snimila sliku iste galaksije više puta
  - Svaka slika sadrži šum
  - Slike su poravnate (nije menjan ugao kamere)



Sabiranjem slika postaje sve jasnija

- Signal I ostaje isti, a šum se menja
- Kada sabiramo dodajemo uvek $I \pm$ slučajni šum

Oduzimanje

$$g(x,y) = f_1(x,y) - f_2(x,y)$$

- Pošto bi vrednosti piksela rezultujuće slike trebali biti nenegativni:
  - $g(x,y) = |f_1(x,y) - f_2(x,y)|$
  - *Shift* piksela tako da se najniže vrednosti mapiraju na 0

-

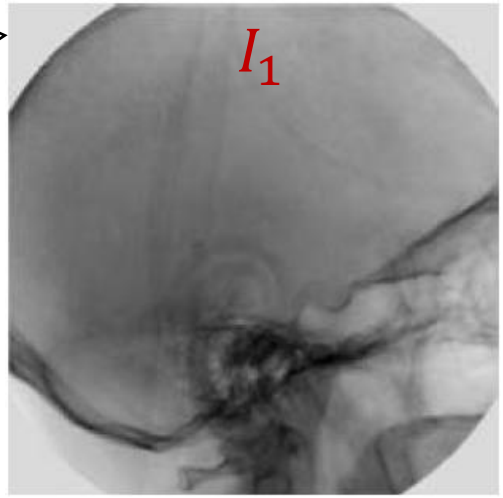
=

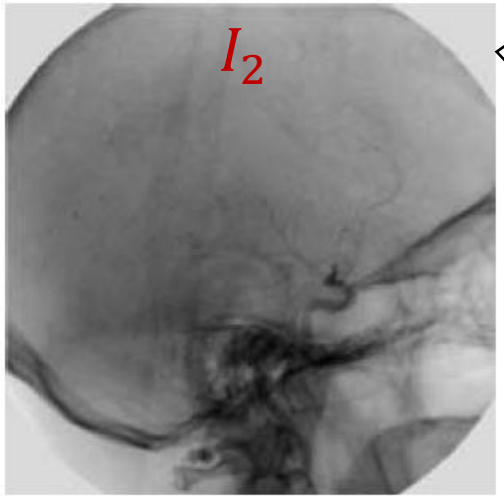
Zgodno za pronalaženje razlike između dve slike

Oduzimanje (maska)

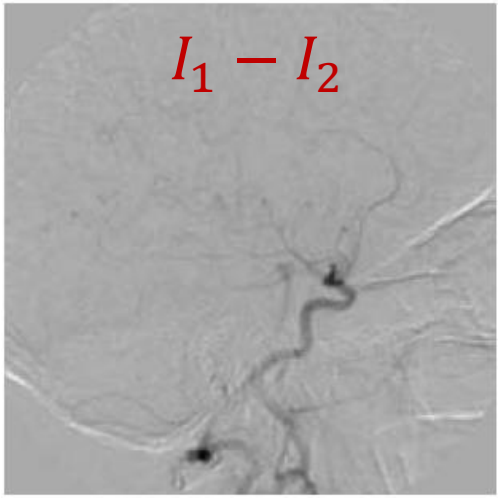
- Imamo dve varijante iste scene, ali snimljene pod različitim uslovima

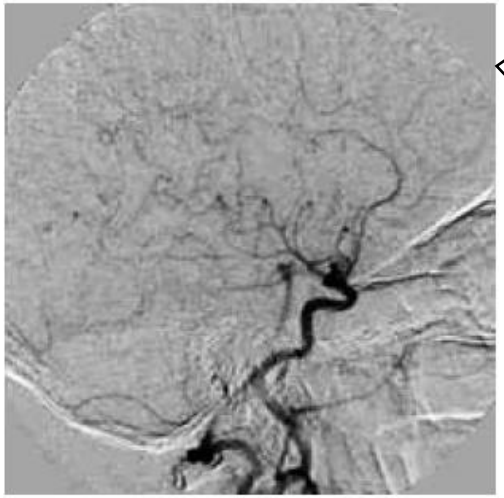
Pre





Nakon ubrizgavanja jodnog kontrastnog sredstva u krvotok





Poboljšan kontrast

Dobili smo jasnu mapu kako se sredstvo propagira kroz krvne sudove mozga

Oduzimanje

- Detekcija pokreta
  u statičnoj sceni (stacionarna kamera)

Pozadina je statična i ne menja se u uzastopnim frejmovima videa

Tražimo značajne promene i pretpostavljamo da one odgovaraju pokretima

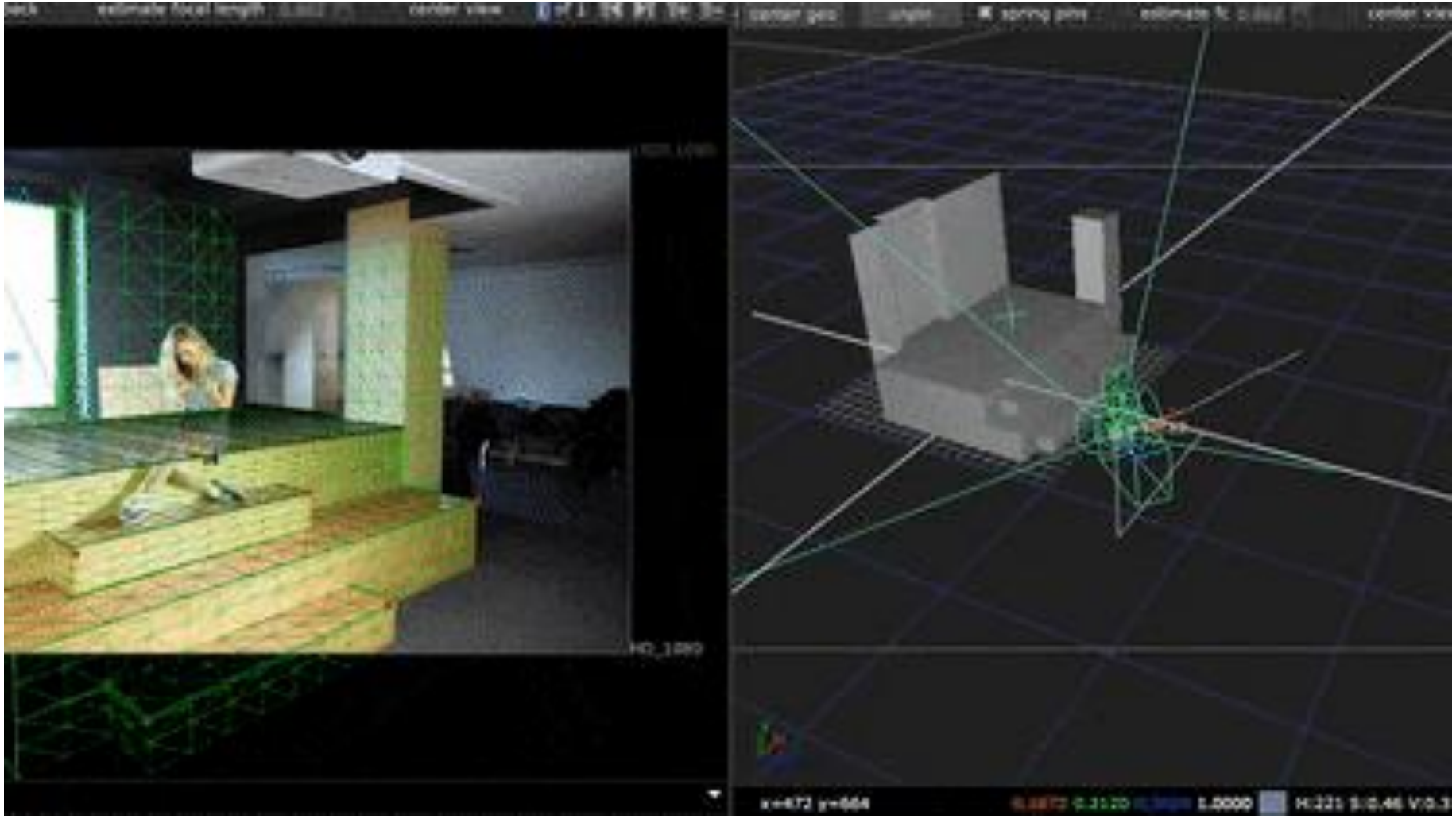
https://www.pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-opencv

Oduzimanje – osnova za:

- M*otion tracking*
  - Nadgledanje
  - Praćenje vozila na putu
  - Prebrojavanje ljudi koji ulaze ili izlaze iz prodavnice ...

- Praćenje pokreta kamere
  - Deo većeg procesa u snimanju filmova koji se zove *match move*
  - Ubacivanje kompjuterski generisane grafike u stvarne scene sa pokretnom kamerom
  - Odgovarajuća pozicija, skaliranje, orijentacija...

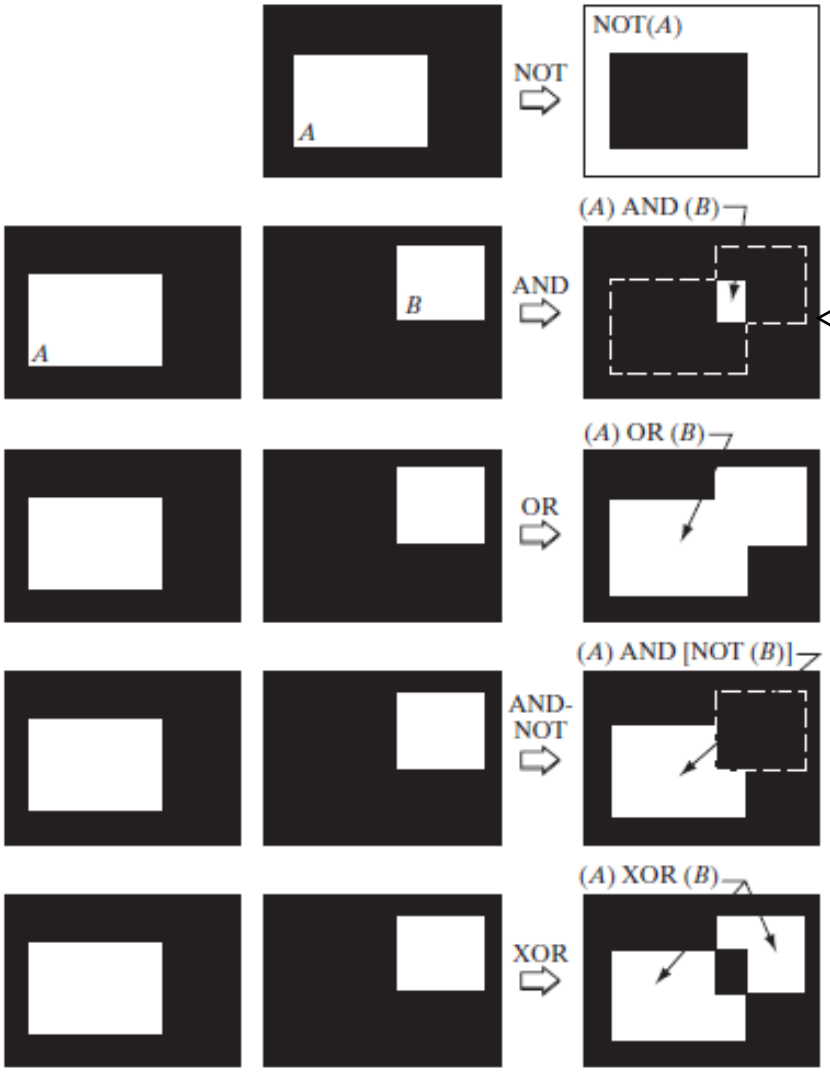
Match move



Oduzimanje – osnova za:

- Video kompresija
  - Enkodirati samo razlike između frejmova
  - Detekcija/predikcija pokreta se koristi u video kompresijama poput MPEG

Logičke operacije



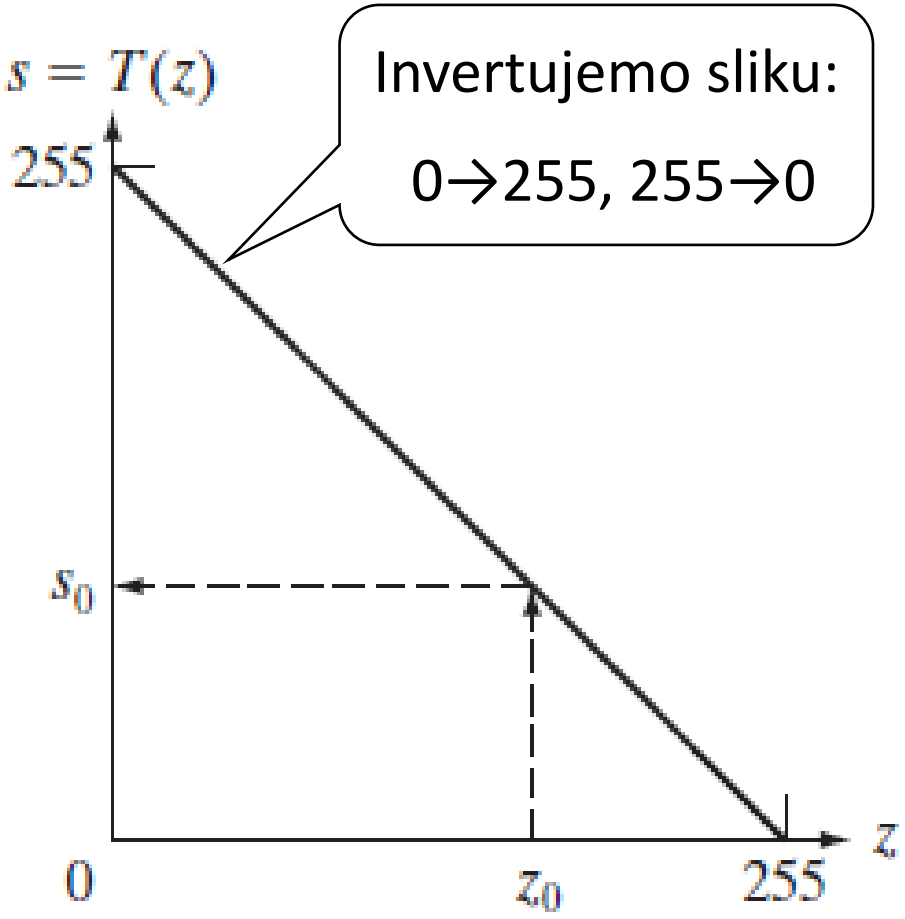
Osnovne logičke operacije nad binarnim slikama

0 – crni pikseli
1 – beli pikseli

Primena:

- Kombinovanje slika
- Segmentacija (primena maske)
- *Green screening*

Negativ



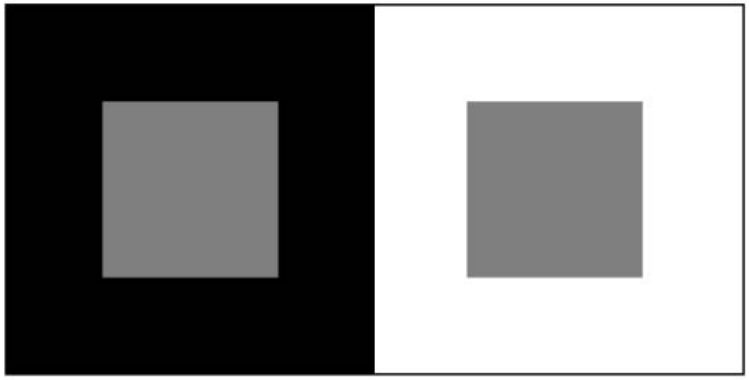
Razlog:

želimo da beli ili sivi detalji na crnoj pozadini budu uočljiviji

Negativ

Original Image

Negative Image



Lokalno uprosečavanje

$$F(x,y) \quad * \quad H(u,v) \quad = \quad G\,(x,y)$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$$\frac{1}{9} * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

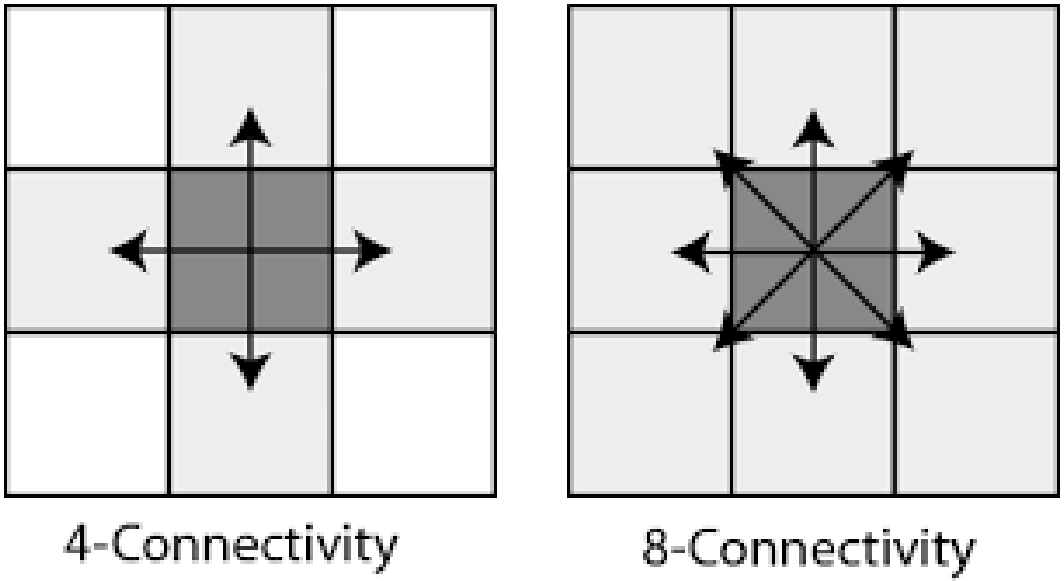
"box filter"

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$G = F * H$$

Lokalno uprosečavanje

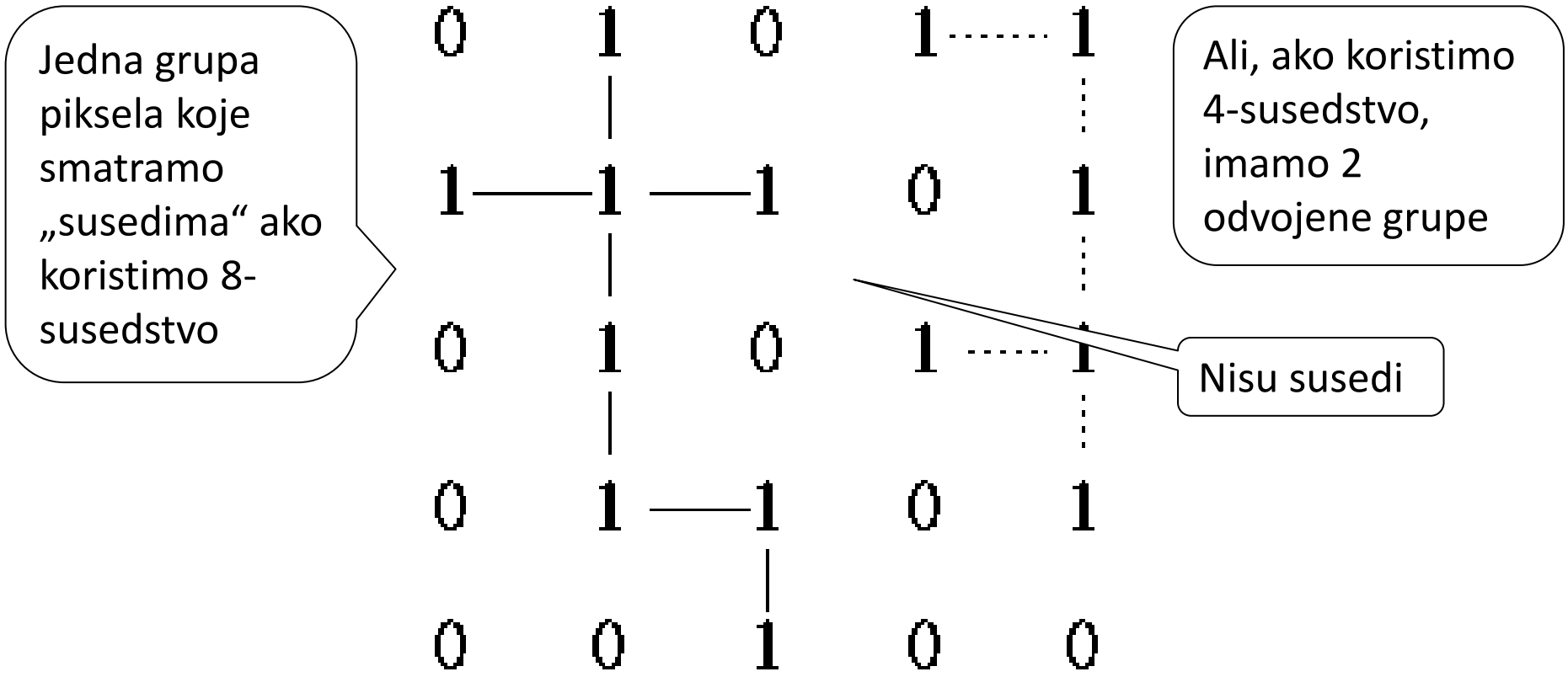
- Definišimo lokalno susedstvo piksela:
  - Sliku smo reprezentovali putem matrice
  - Možemo definisati susedstvo piksela preko susednih elemenata



Tipično, za definiciju prostorne povezanosti

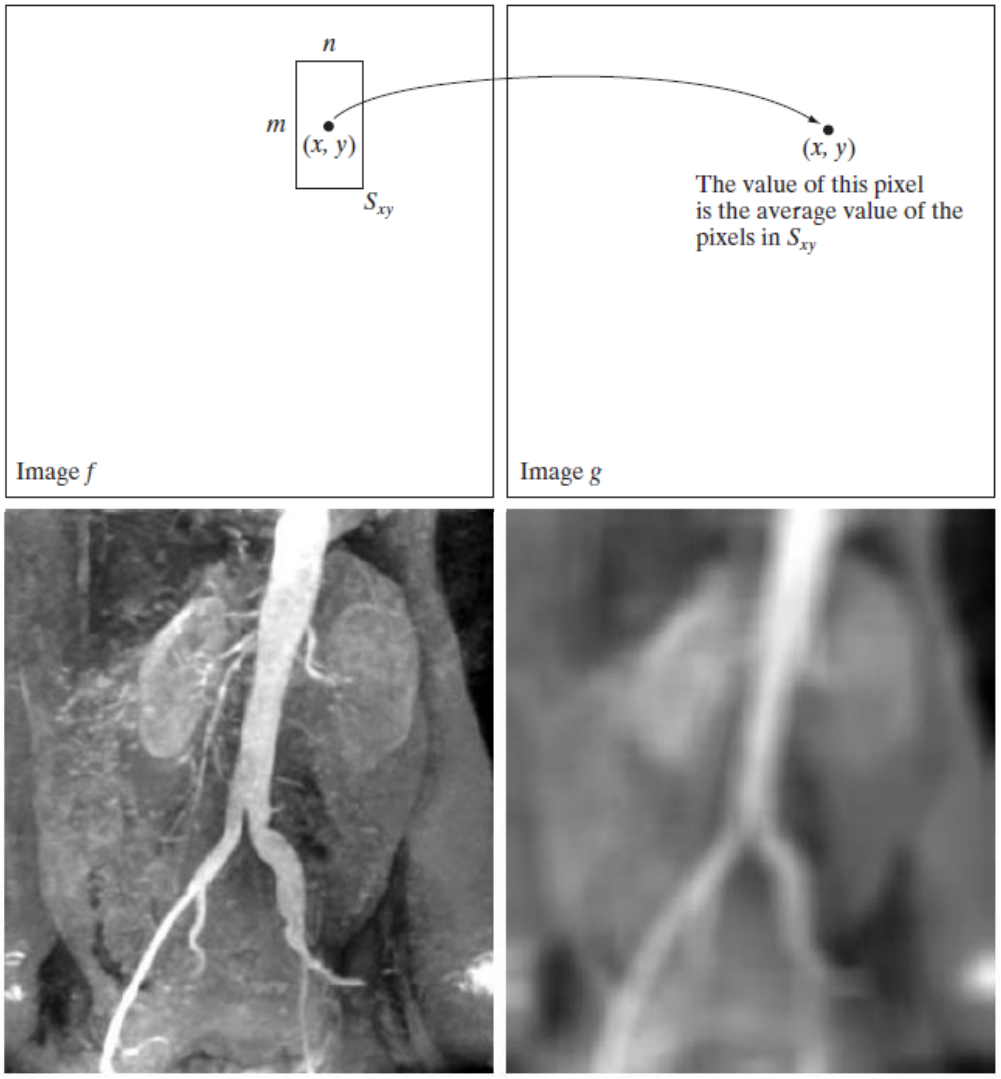
Povezanost piksela

- Pojam susedstva možemo definisati i za grupe piksela
- Rezultati će se razlikovati u zavisnosti od korišćenog pojma susedstva

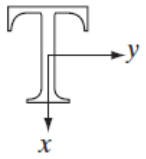
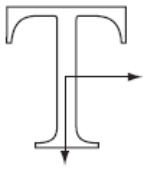
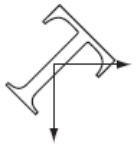
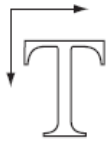
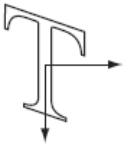


Lokalno uprosečavanje

- Vrednost piksela zamenjujemo prosečnom vrednošću njegovog susedstva
  - Zavisi od definicije susedstva
- Uklanjamo šum
- Ali dobijamo zamućeniju sliku
- Ima sofisticiranijih metoda koje prepoznaju gde su veće promene i ne koriste piksele sa druge strane oblasti koja predstavlja promenu

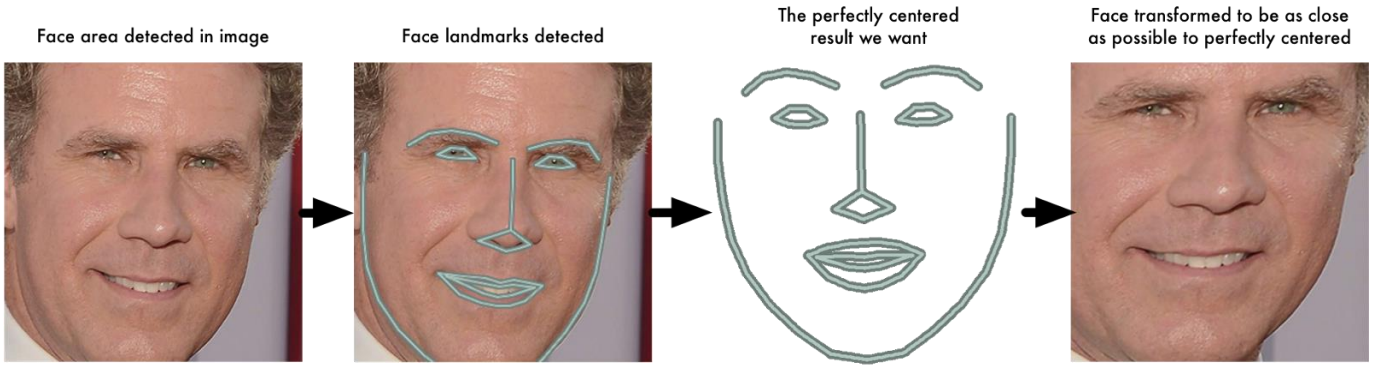


Affine transformacije

Transformation Name	Affine Matrix, T	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = w$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = c_x v$ $y = c_y w$	
Rotation	$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v\cos\theta - w\sin\theta$ $y = v\cos\theta + w\sin\theta$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$x = v + t_x$ $y = w + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v + s_v w$ $y = w$	
Shear (horizontal)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = s_h v + w$	

$$[x \;\; y \;\; 1] = [v \;\; w \;\; 1]\,\mathbf{T} = [v \;\; w \;\; 1] \begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$$

Affine transformacije



Rotacija lica: radi pojednostavljenja prepoznavanja

https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78

Isti otisak, ali različita pozicija i veličina – teško za poređenje

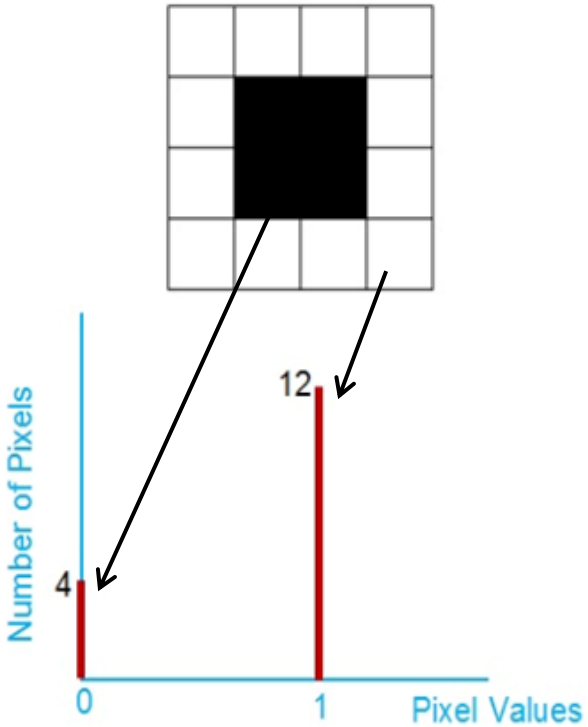
Poboljšanje kontrasta

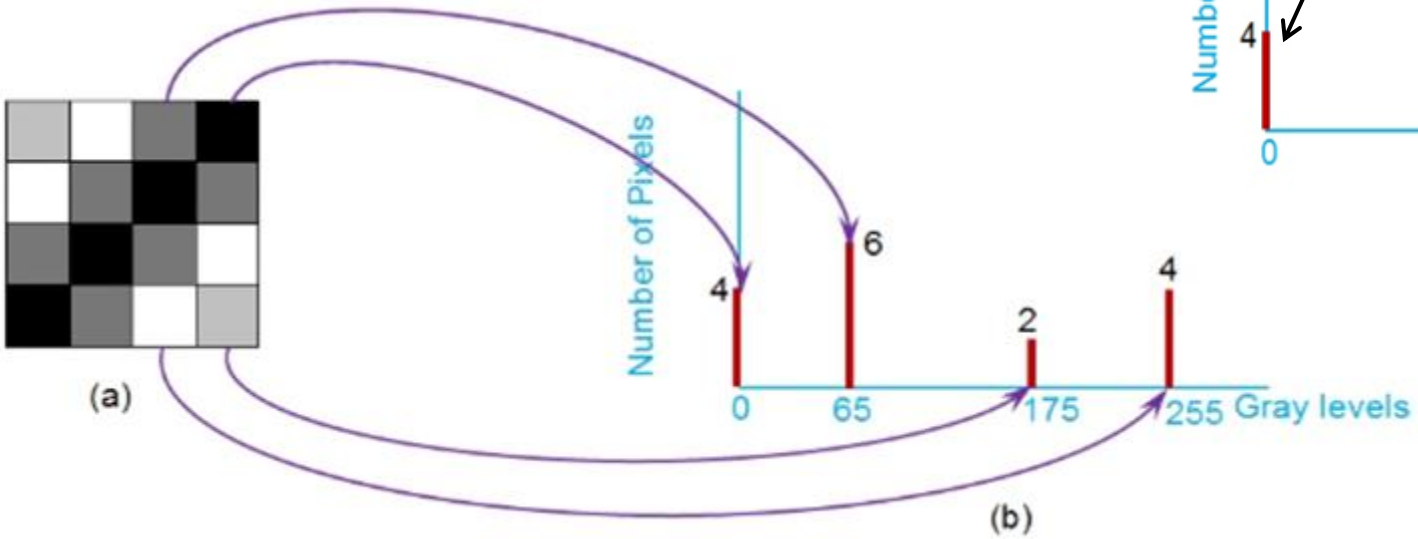
- Koristimo statističke osobine slike
  (sažetak sadržaja slike koji pomaže njenoj interpretaciji)

- Primer: histogram

Histogram

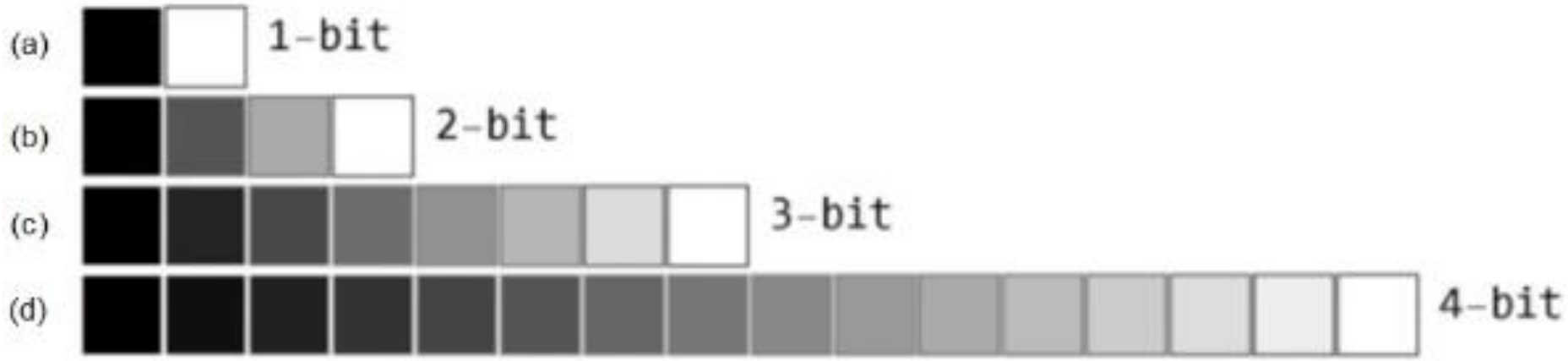
- Grafički prikaz brojnosti piksela datog intenziteta
  - Na x osi se predstavlja intenzitet piksela
  - Za svaki intenzitet na x osi prebrojimo koliko ima piksela tog intenziteta
  - visina stupca prikazanog na histogramu će biti proporcionalna tom broju





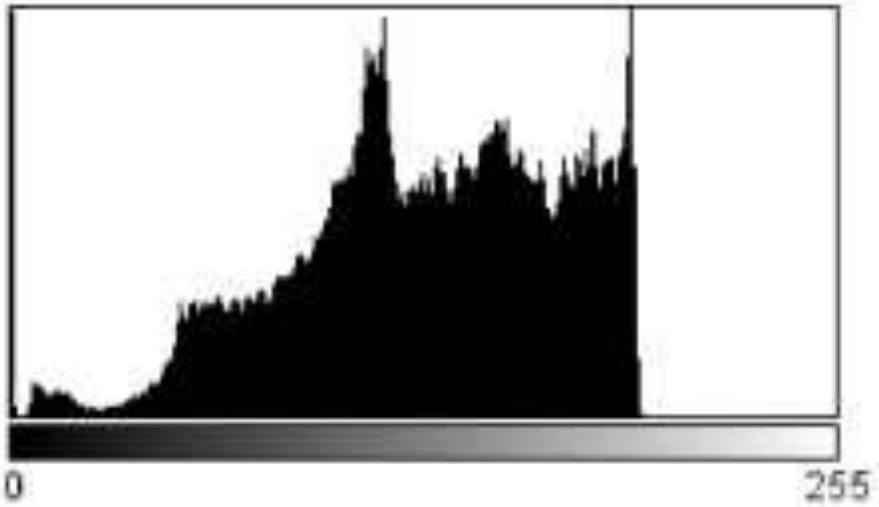
Histogram monohromatske (grayscale) slike

- X-osa: vrednosti intenziteta piksela
  - Broj bita neophodan da predstavi sliku → broj nijansi koje je moguće predstaviti
  - *N* bitova → 2^N mogućih vrednosti piksela u opsegu $[0, 2^N - 1]$



Histogram monohromatske (grayscale) slike

Histogram je grafička reprezentacija količine svakog nivoa sive u slici

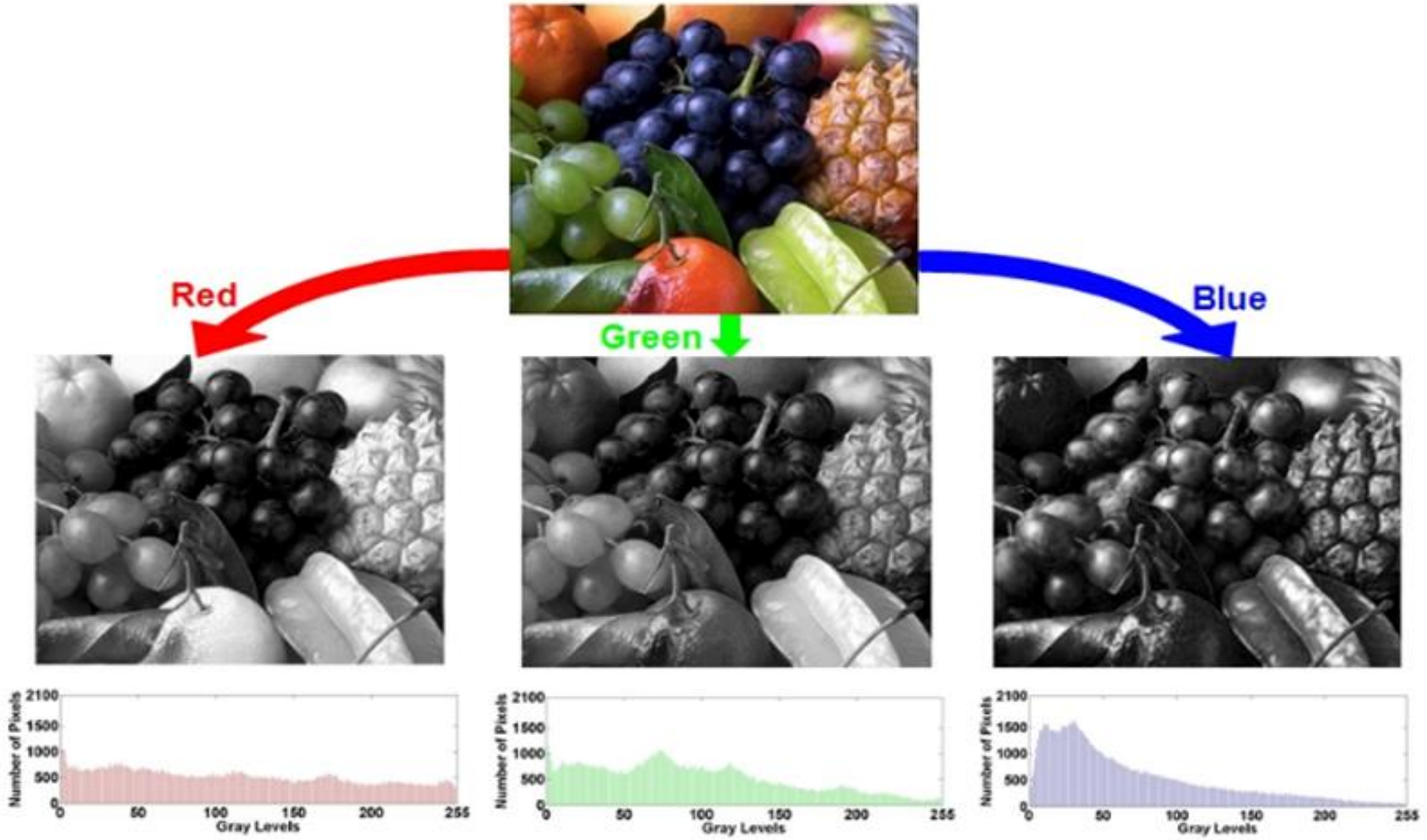


Count: 179463
Mean: 126.565
StdDev: 43.800

Min: 0
Max: 213
Mode: 192 (2241)

Histogram slike u boji

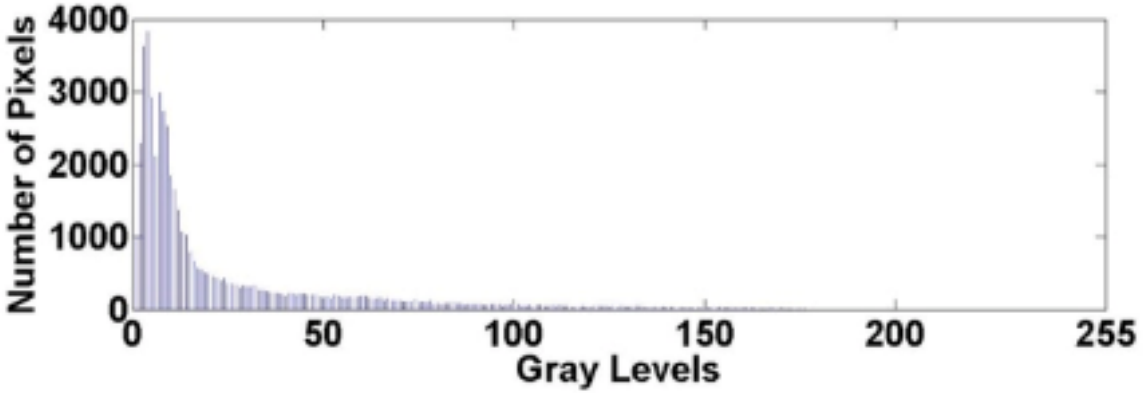
- Za slike u boji, moguće je analizirati svaki kanal zasebno



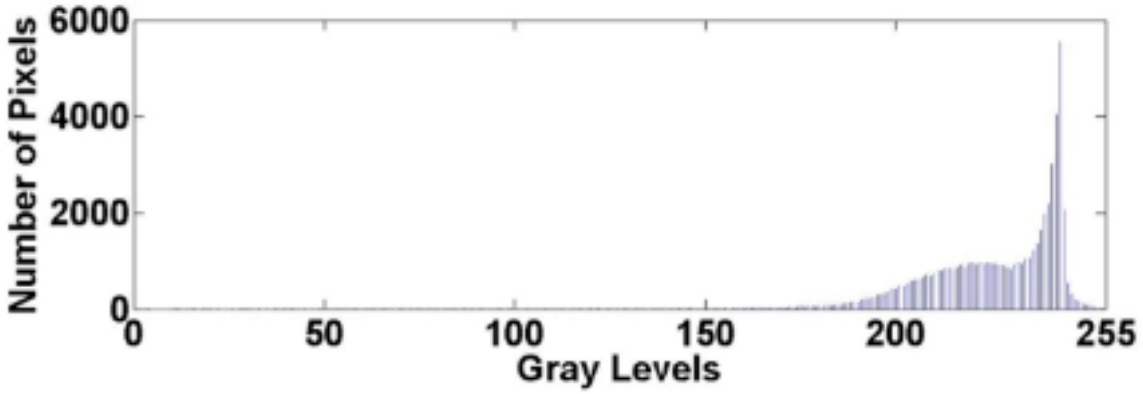
Šta možemo saznati sa histograma?

- Raspon vrednosti na histogramu: osvetljenje
  - Prosečna vrednost nam govori o osvetljenosti slike
  - *min* i *max* nam govore o rasponu intenziteta koji je snimljen
- Standardna devijacija: kontrast
  - Visoka: slika nešto predstavlja (barem u nekom svom delu)
  - Niska: monotona sliku

Osvetljenje slike



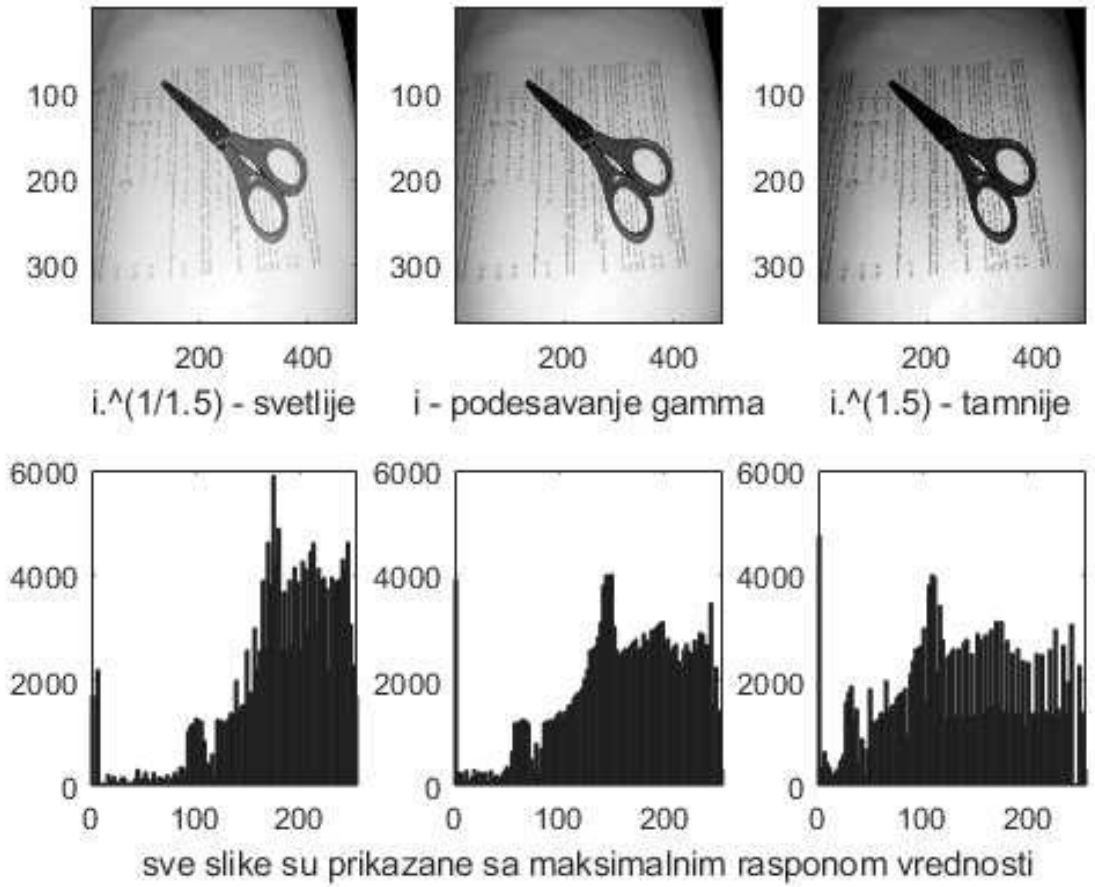
Nisko osvetljenje: vrednosti histograma su koncentrisane uglavnom na levoj strani



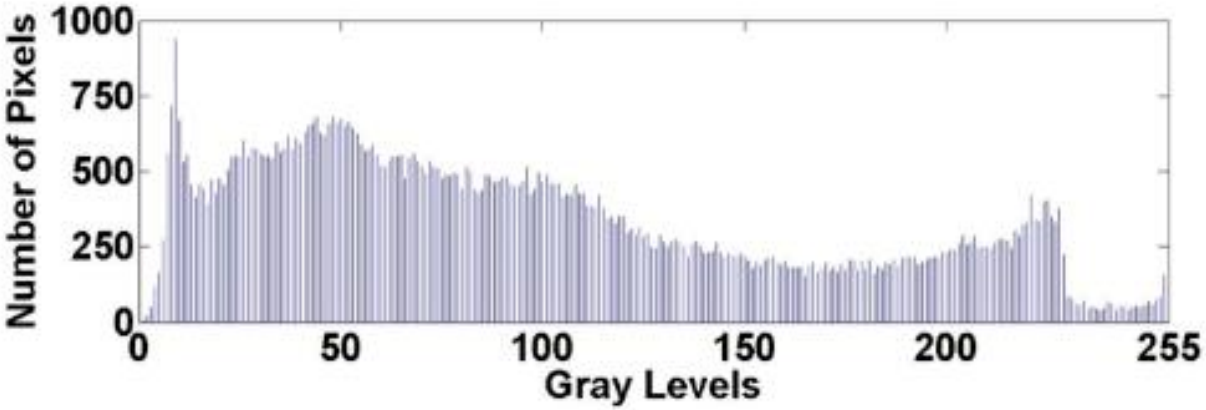
Visoko osvetljenje: vrednosti histograma su koncentrisane uglavnom na desnoj strani

Osvetljenje slike

- Ako ima previše piksela sa visokim ili niskim intenzitetom, tada je obično kontrast slike nizak i slika je previše svetla ili previše tamna

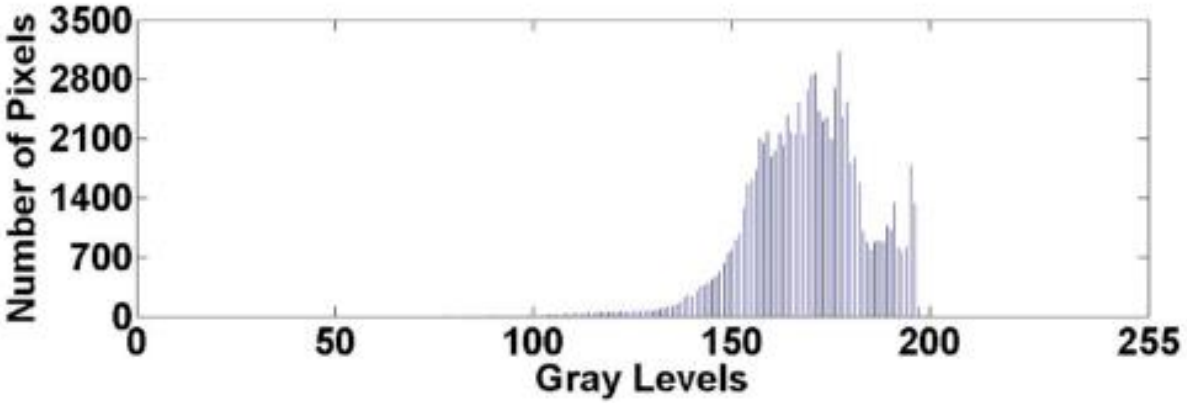


Kontrast



(a)

Vrednosti ravnomerno raspodeljene po intenzitetima → dobar kontrast u osvetljenju

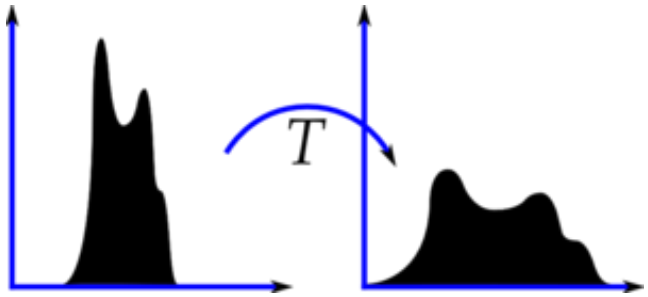


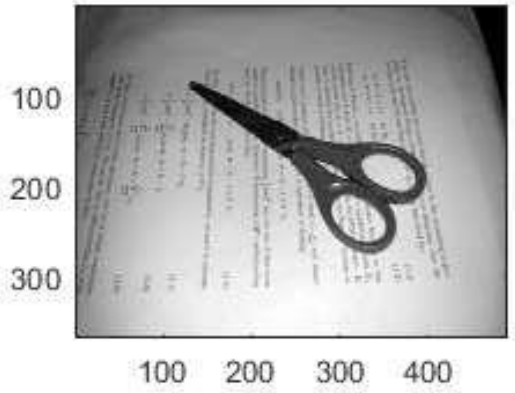
(b)

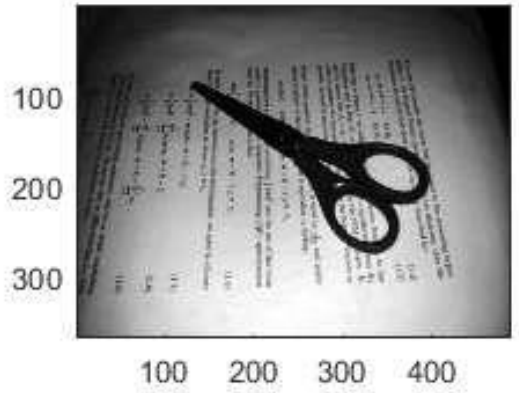
Koncentracija vrednosti oko jednog određenog intenziteta → loš kontrast u osvetljenju

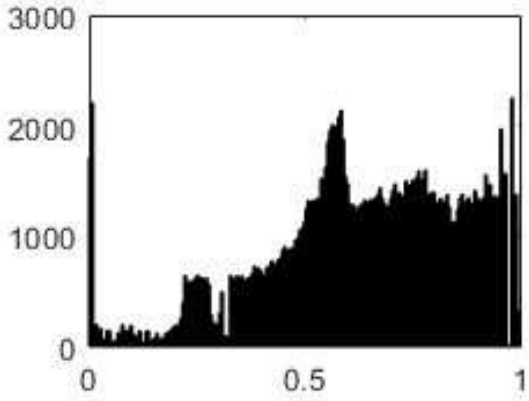
Poravnavanje histograma (*Histogram Equalization*)

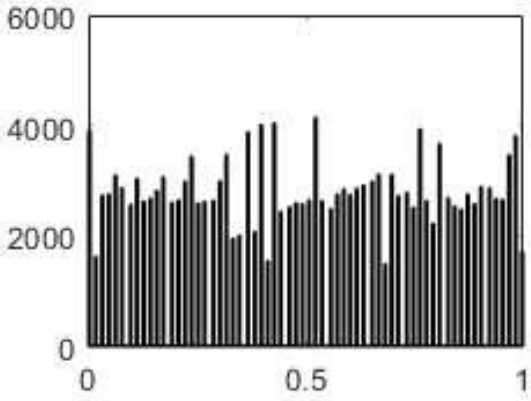
- Modifikujemo distribuciju intenziteta piksela tako da postoji otprilike jednak broj piksela za svaki od intenziteta



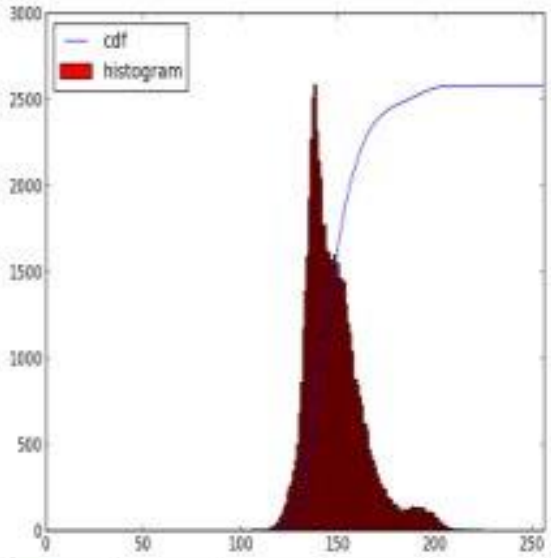


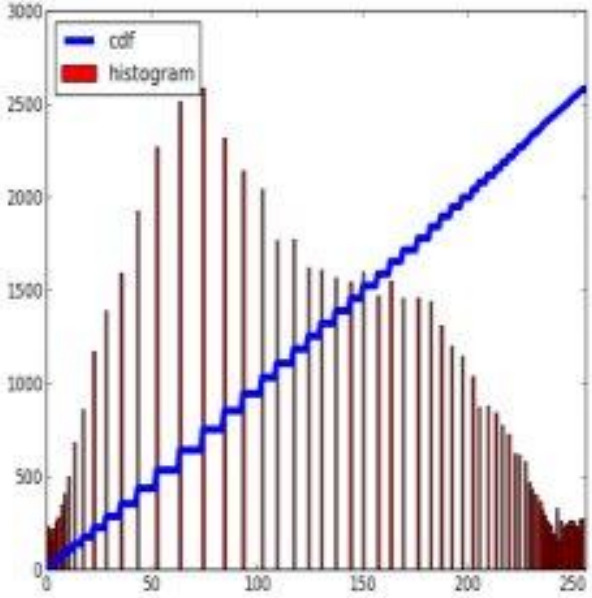






Poravnavanje histograma





https://docs.opencv.org/3.1.0/d5/daf/tutorial_py_histogram_equalization.html

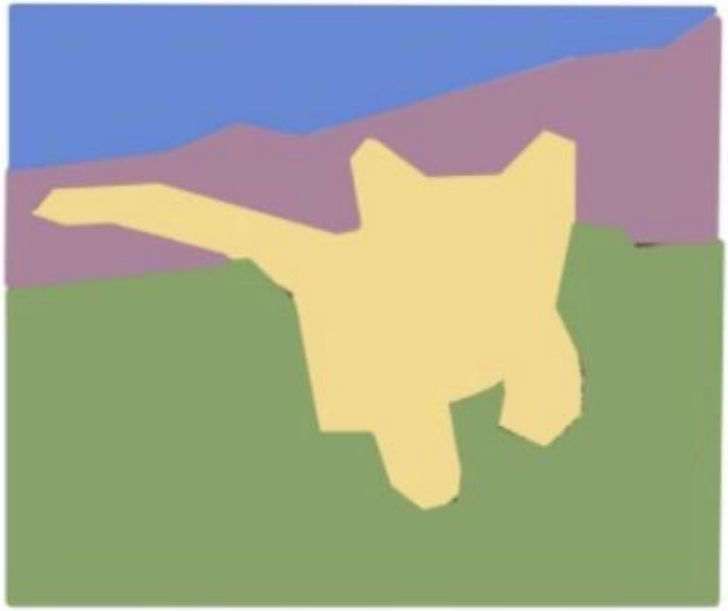
Poravnavanje histograma (*Histogram Equalization*)

- Ova operacija pojačava kontrast na slici
- Veoma korisno za slike gde su i pozadina i objekti oboje previše svetli ili previše tamni
- Na primer, ovo pomaže kod rendgena da se bolje vidi struktura kostiju i da se pobojšaju slike koje su *overexposed* ili *underexposed*
- Reverzibilna transformacija (možemo je invertovati da dobijemo originalnu sliku)
- Mana: može da poveća kontraste na pozadini a smanji koristan signal

Segmentacija slike

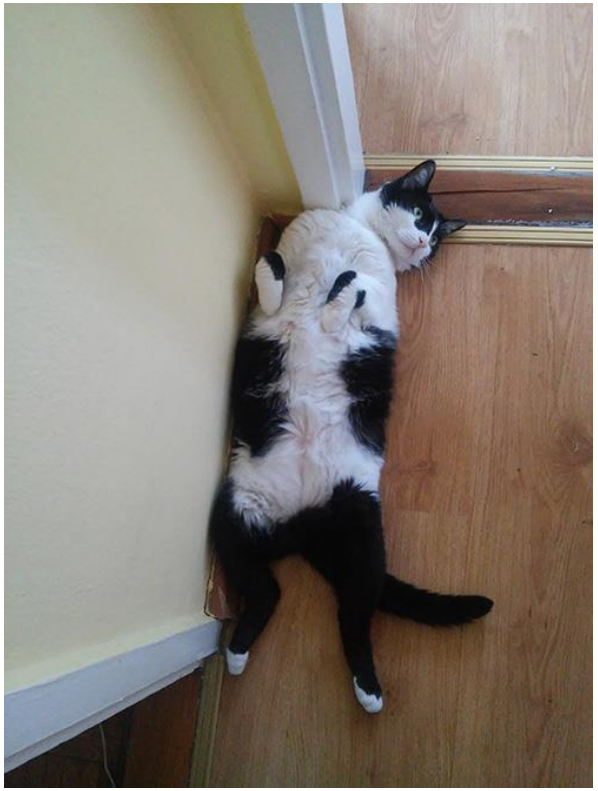
Segmentacija slike

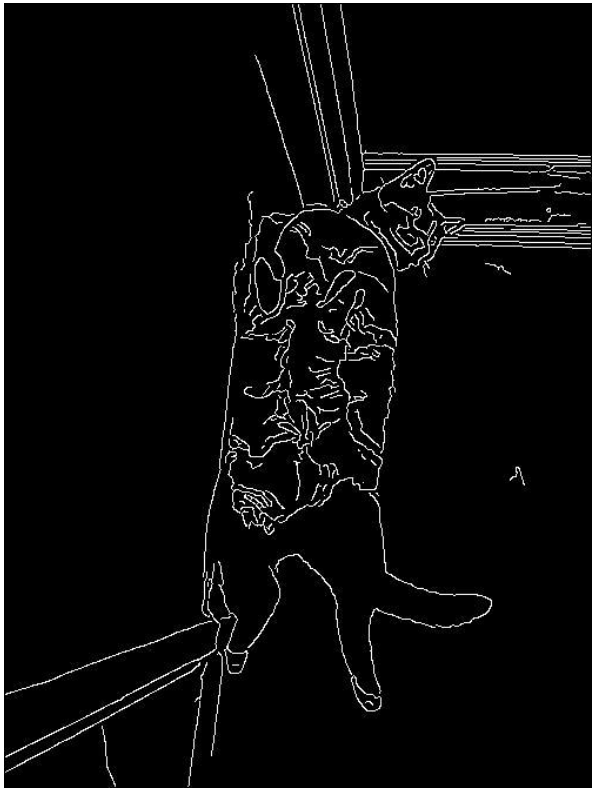
- Postupak podele slike na regione sa sličnim atributima
  - Izdvajanje objekta sa pozadine (*foreground/background separation, background subtraction*)
  - Semantička anotacija delova slike (klasifikacija piksela slike u odgovarajuće kategorije)



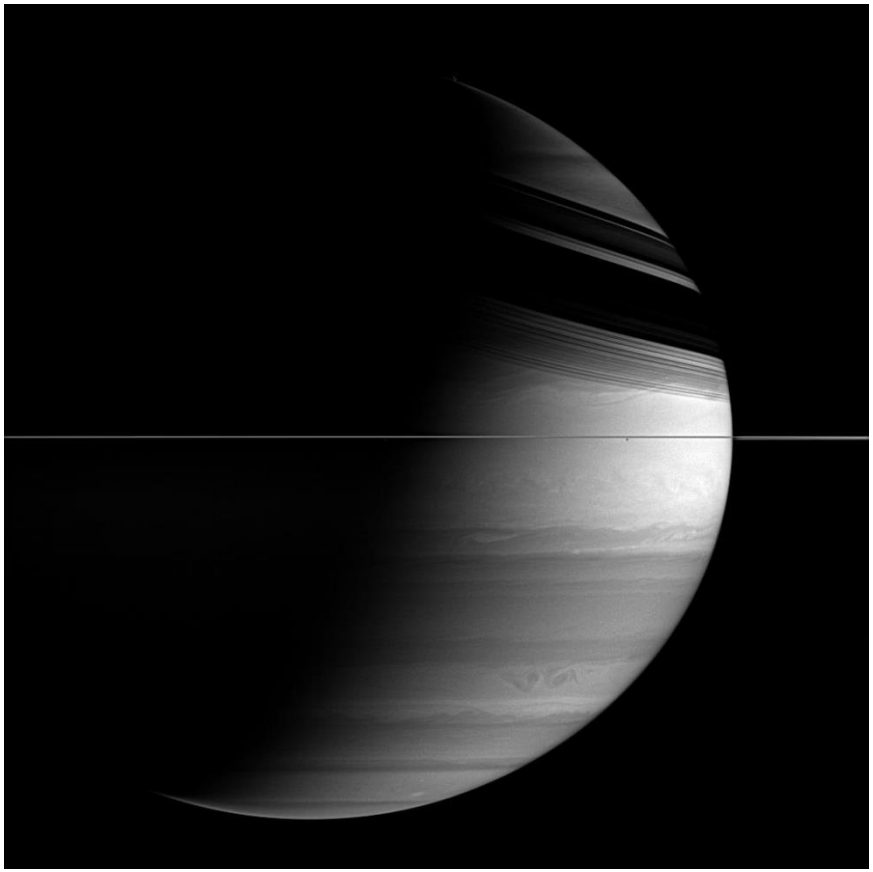
Na osnovu čega segmentišemo?

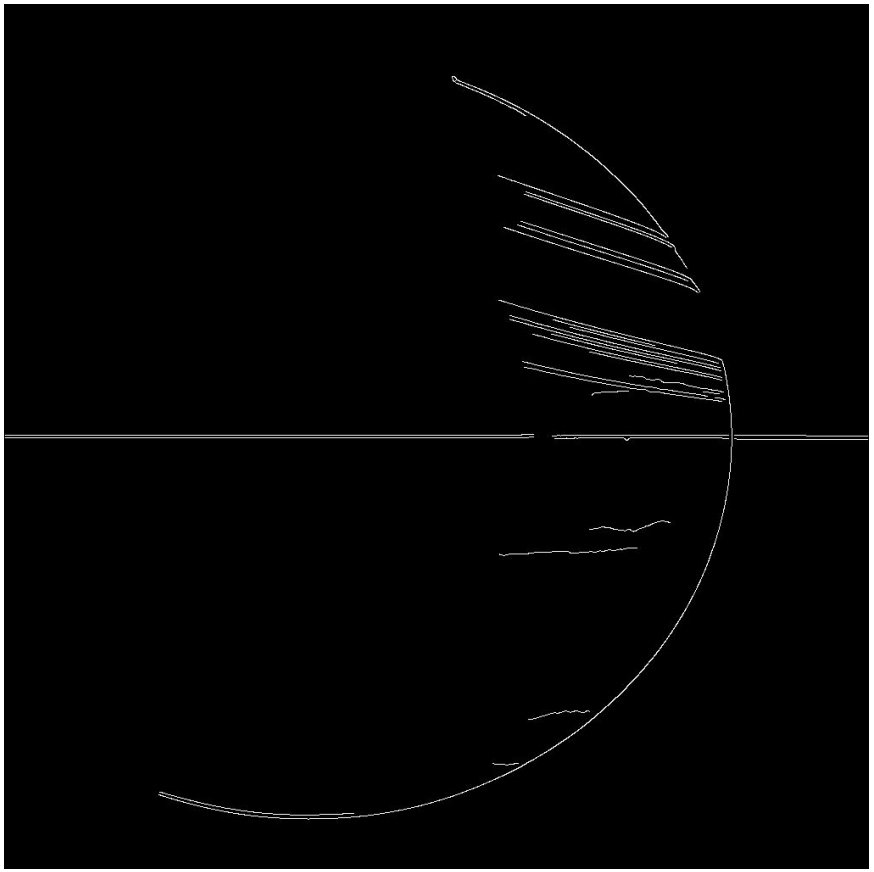
- Koji tip informacija (atribute) možemo koristiti za segmentaciju?
  - Osvetljenost, boja, ivice, mere teksture,...





Na osnovu čega segmentišemo?

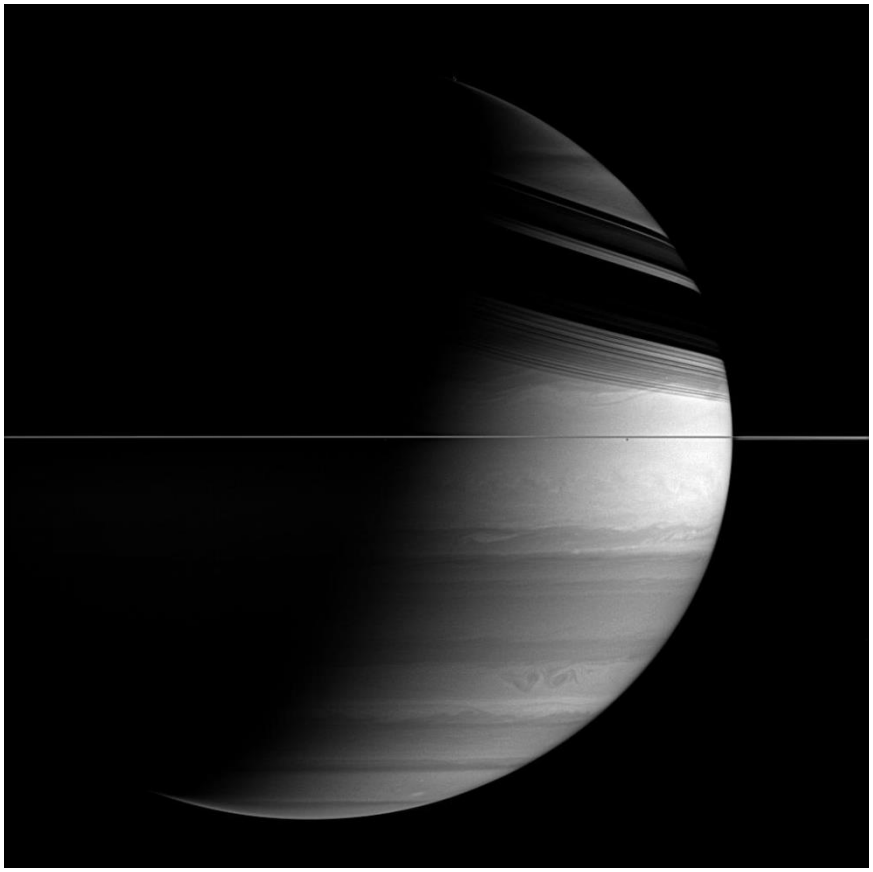


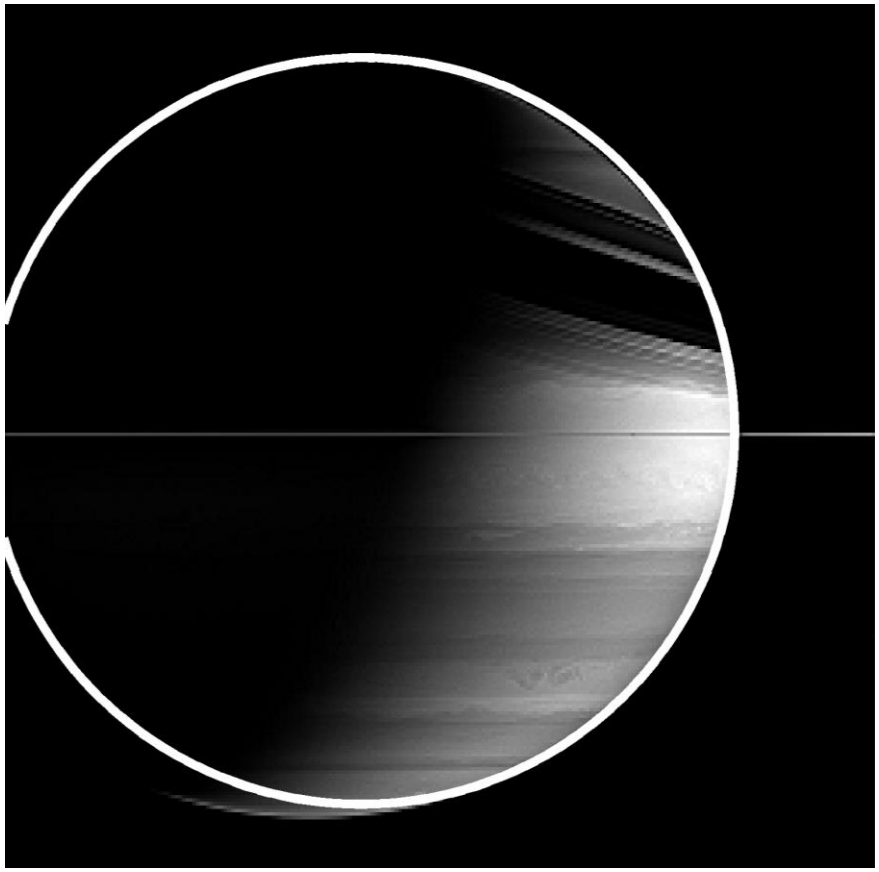


- Ivice su važan atribut, ali nisu uvek dovoljne
- Dobro je obuhvatiti više izvora informacija

Na osnovu čega segmentišemo?

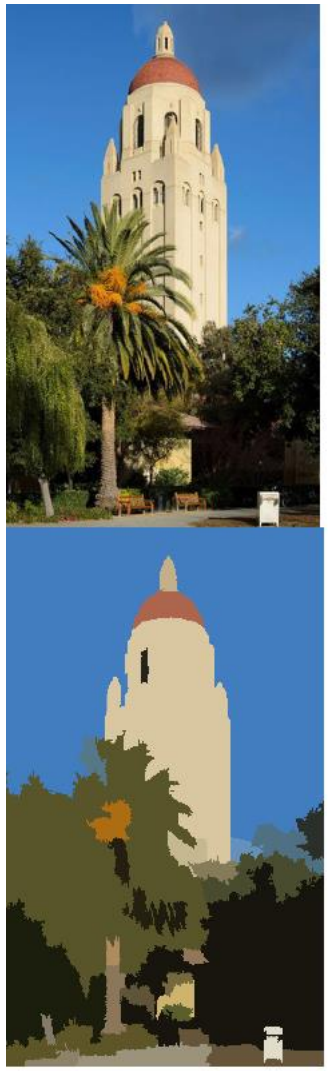
- Hough pronalaženje krugova:





Segmentacija slike

- Kako znamo da je segmentacija dobra?
  - Pikseli iste kategorije (sličnih vrednosti) formiraju povezani region
  - Susedni pikseli u različitim kategorijama i imaju različite vrednosti
- Segmentacija je najčešće kritičan korak u analizi slike jer je to tačka u kojoj se prelazi sa piksela na objekat kao jedinice obrade
- Ako je segmentacija uspešna, ona će olakšati ostale korake analize slike

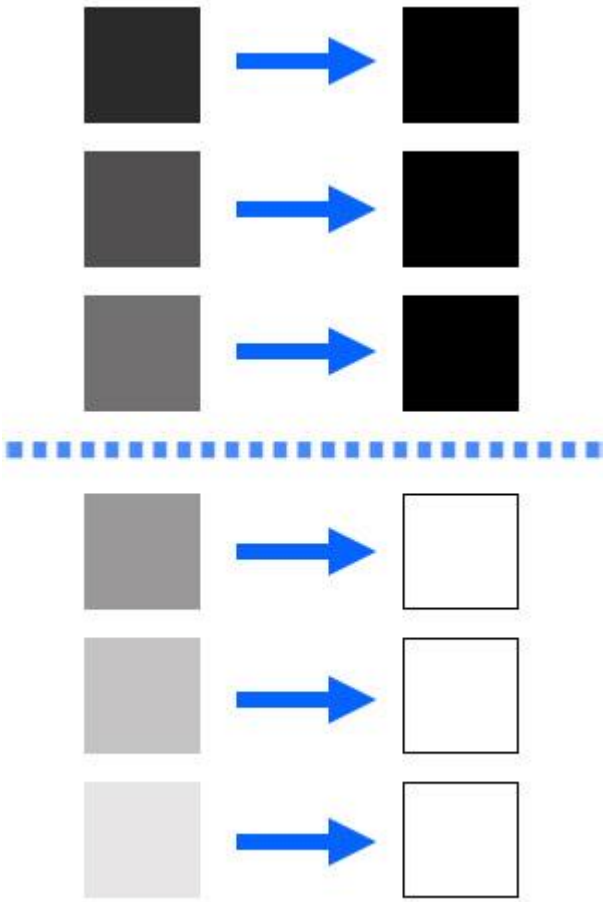


Segmentacija slike

- Gruba podela pristupa:
  - Histogram (određivanje praga)
  - Klasterovanje (poseban termin predavanja)
  - Praćenje ivica (morfološki pristupi, aktivne konture,...) ...

Segmentacija slike pomoću praga

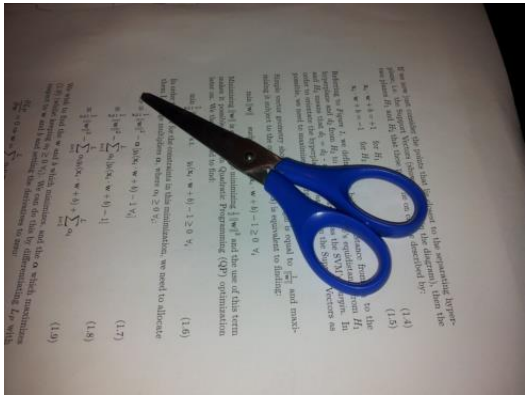
- Najjednostavniji način za segmentaciju slike
- *Thresholding*
- Poredimo osvetljenost piksela sa pragom (ili više pragova) i svrstavamo piksele u dve (ili više) kategorija

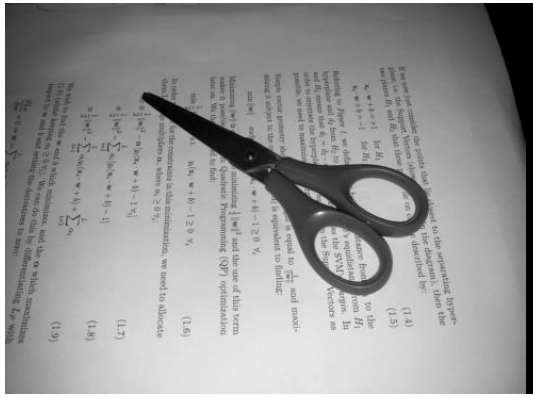


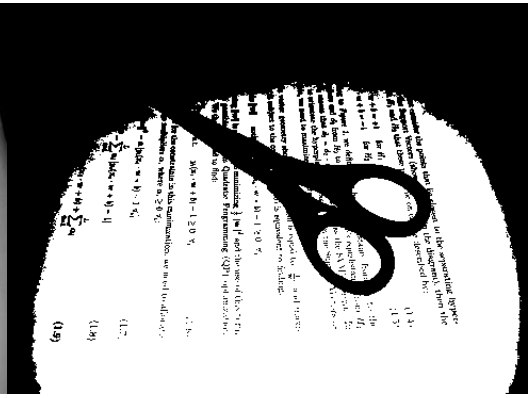
Segmentacija pomoću **globalnog** praga

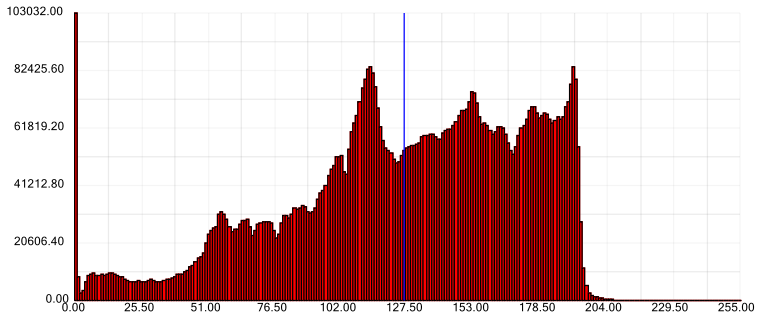
- Najjednostavniji vid segmentacije pomoću praga je da poredimo vrednosti piksela sa samo jednim pragom
  1. Sliku ćemo pretvoriti u *grayscale*
  2. Odabraćemo vrednost praga *T*
  3. Klasifikujemo piksele prema intenzitetu

$$B[x,y] = \begin{cases} 0, & f(x,y) \leq T \\ 1, & f(x,y) > T \end{cases}$$



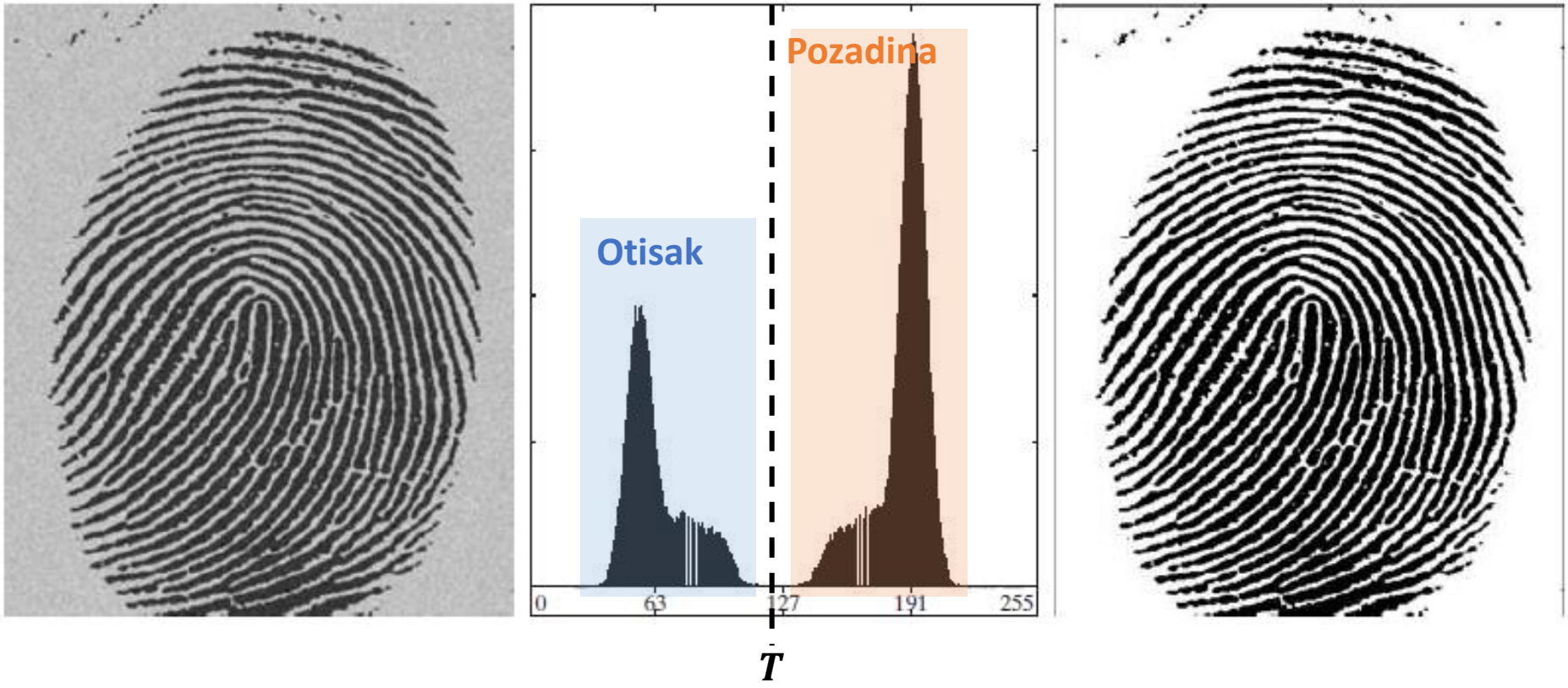




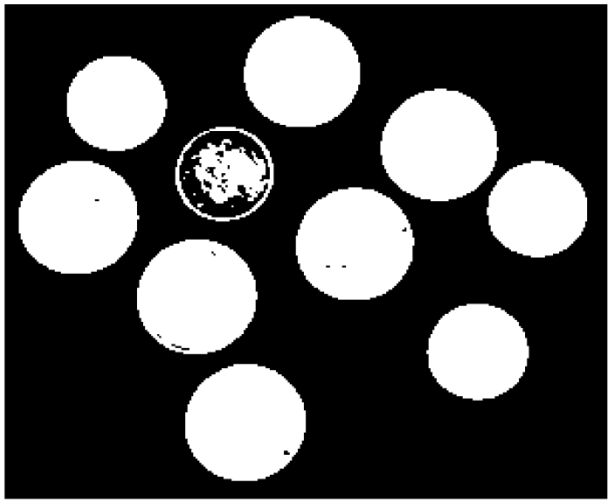


Kako odrediti vrednost praga?

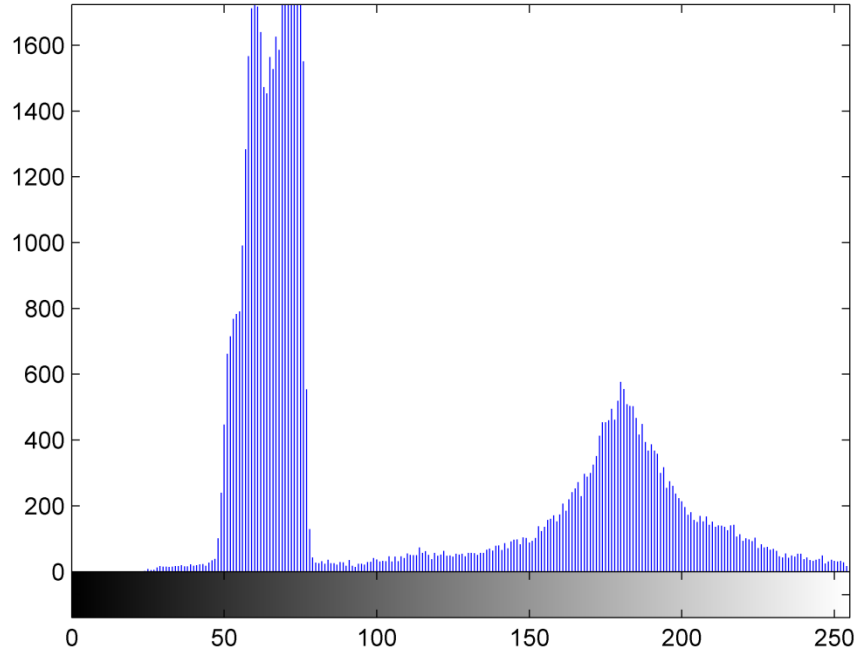
- *Otsu*: automatski pronalazi prag na osnovu histograma
- Ideja je prikazana na slici:
  - Multimodalni histogram: jasno razdvojeni intenziteti linija otiska i pozadine
  - Optimalan prag je obično u „dolinama“ histograma



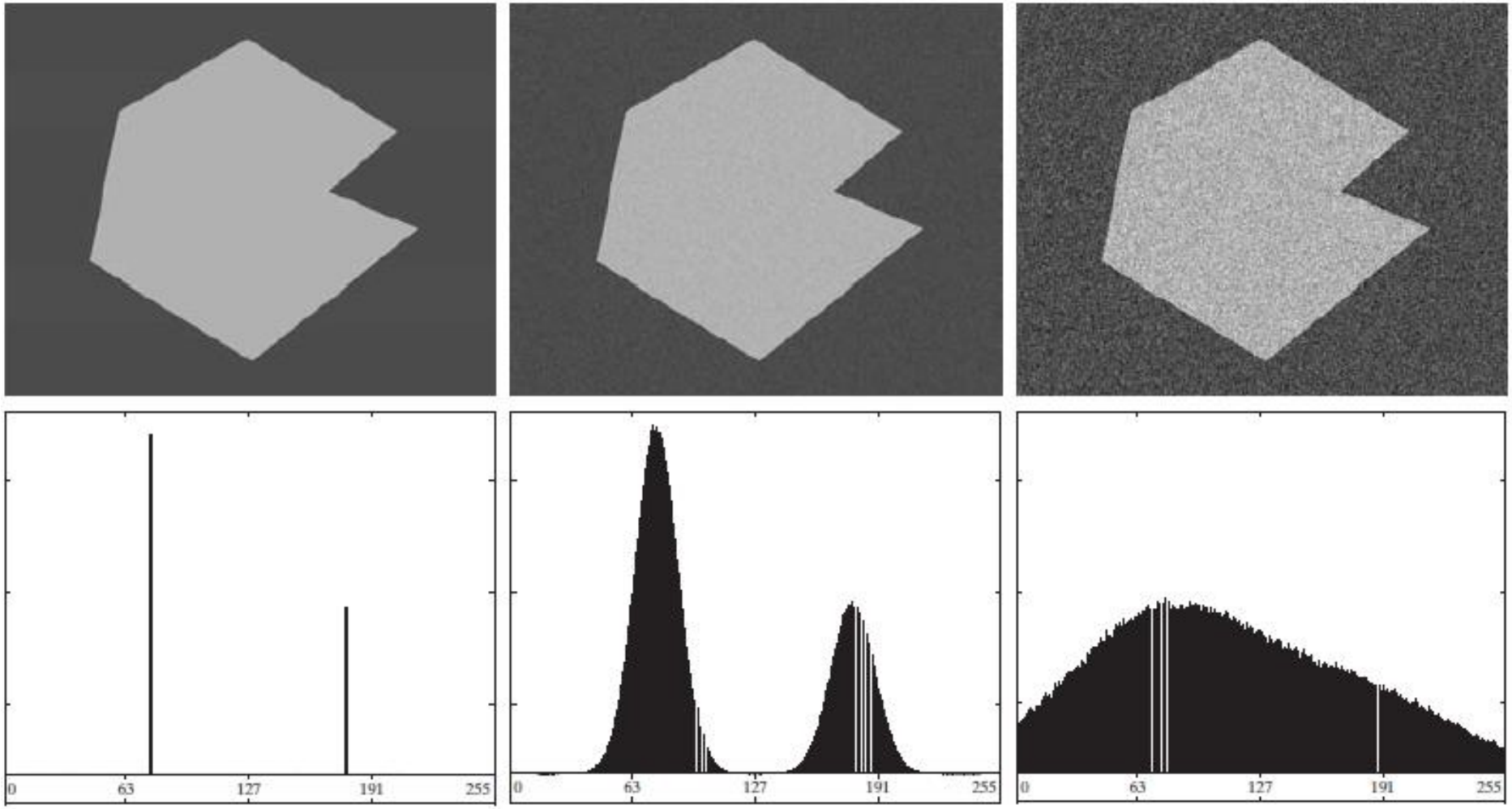
Otsu – prag je u “dolini” histograma



- Veoma jednostavno
- Potpuno automatski

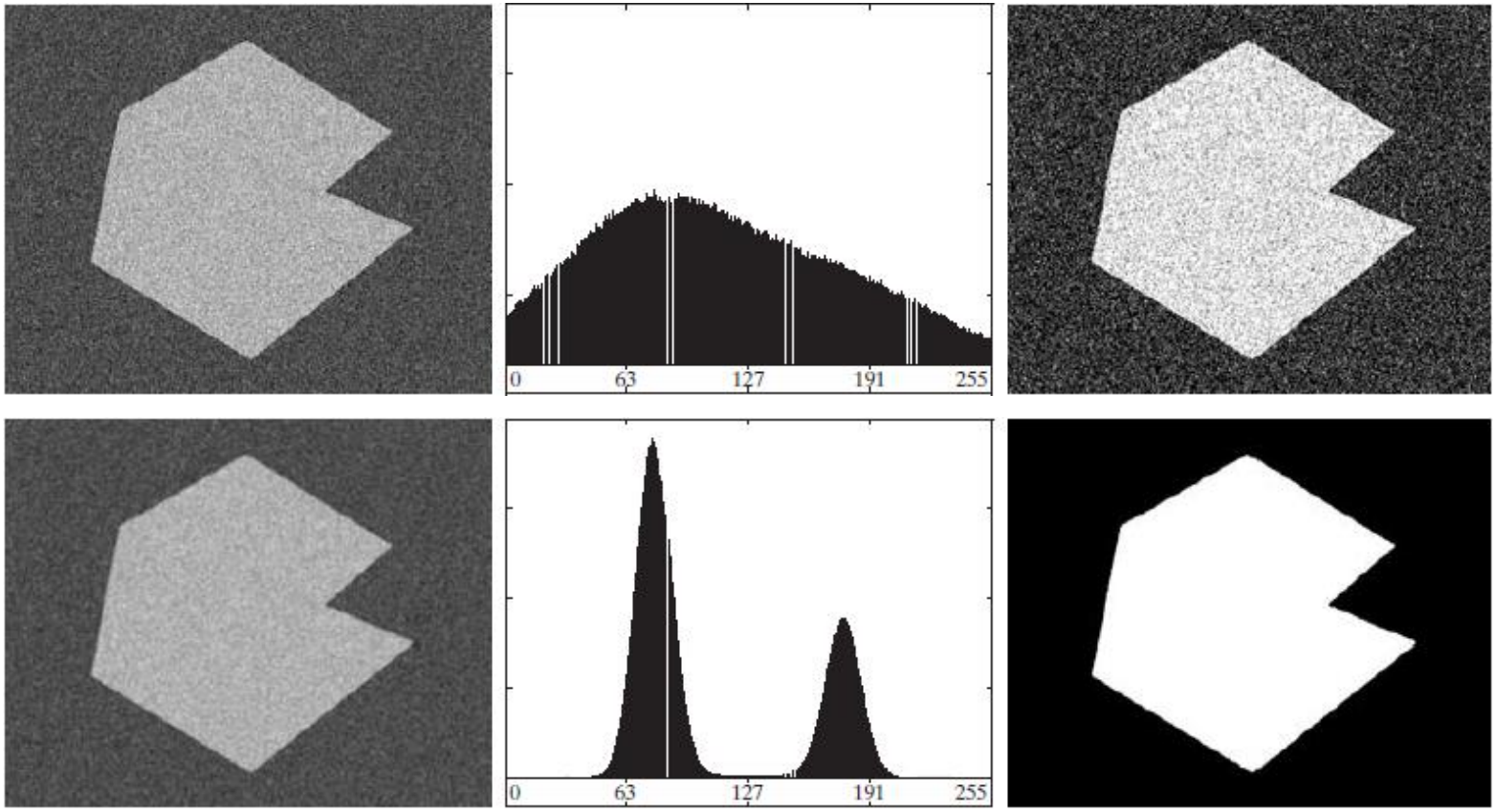


Problem šuma



Problem šuma – rezultat primene *Otsu* praga

na zašumljenim slikama, segmentacija pomoću *Otsu* praga nije dobra

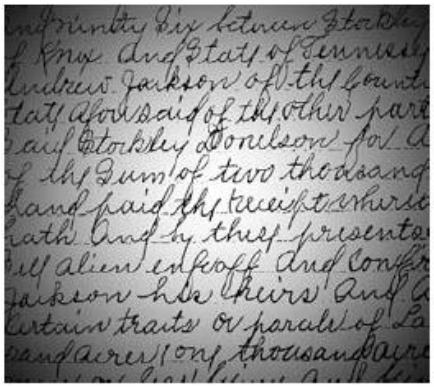


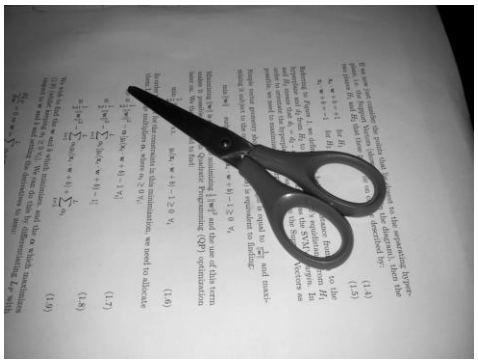
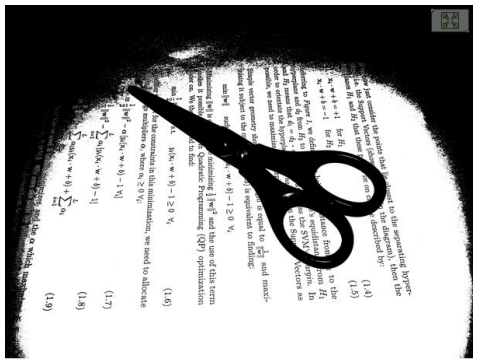
problem možemo ispraviti tako što pre primene *Otsu* algoritma uklonimo šum

Otsu nedostaci

- *Otsu* zanemaruje prostorne relacije
  - Koristi se samo histogram
  - Regije sa sličnim vrednostima intenziteta piksela se na histogramu nalaze blizu, bez obzira što mogu biti na sasvim različitim pozicijama slike
  - Otsu će ih tretirati kao da se radi o istoj regiji

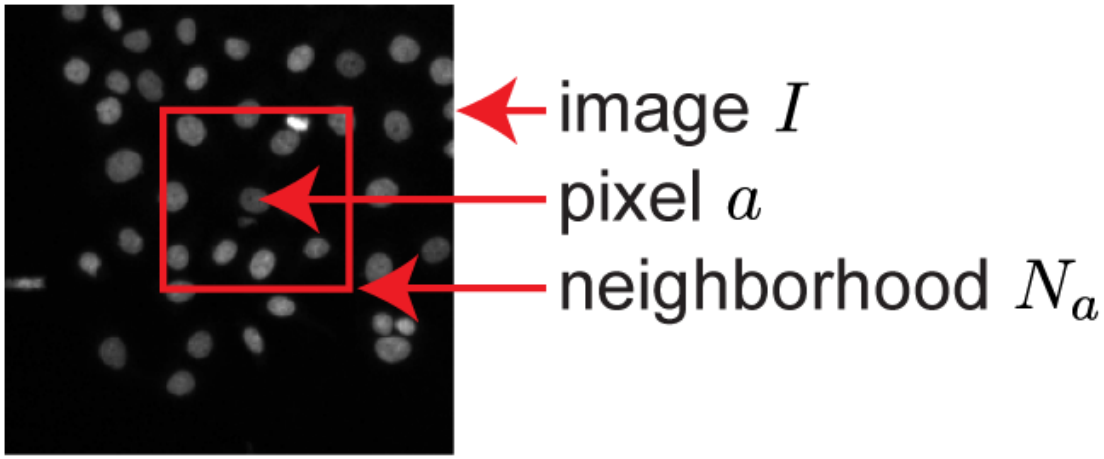
- Nejednako osvetljenje (generalni problem kod korišćenja globalnog praga):



Segmentacija pomoću **lokalnog** praga

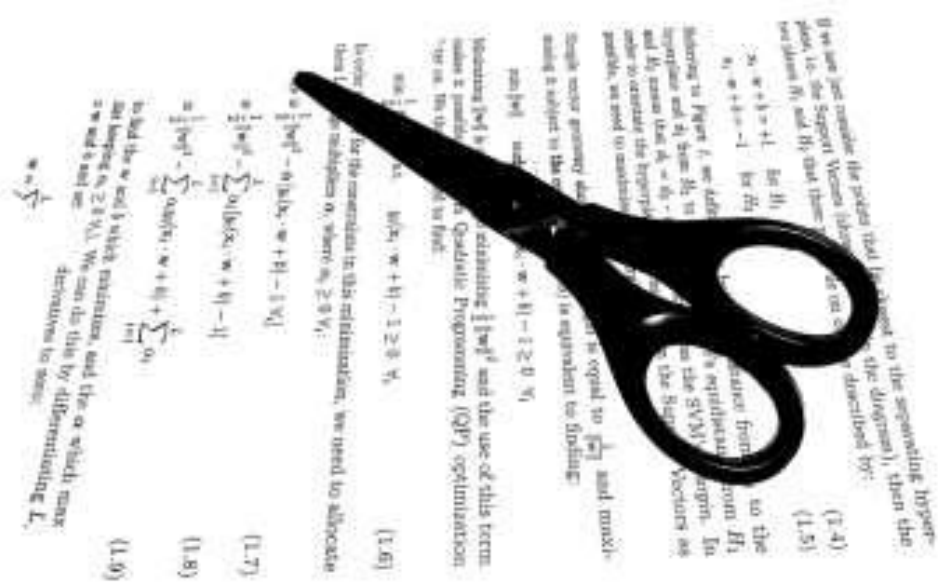
- Rešenje nejednakog osvetljenja: tražimo lokalni prag
  - Izaberemo prag pojedinačno za svaki piksel na osnovu lokalnih vrednosti piksela u njegovom susedstvu
  - Za svako susedstvo od 50×50 piksela, izabraćemo prag za segmentaciju tih piksela



Global threshold: $\tau = f(\{i \in I\})$

Local threshold: $\tau_a = f(\{i \in N_a\})$

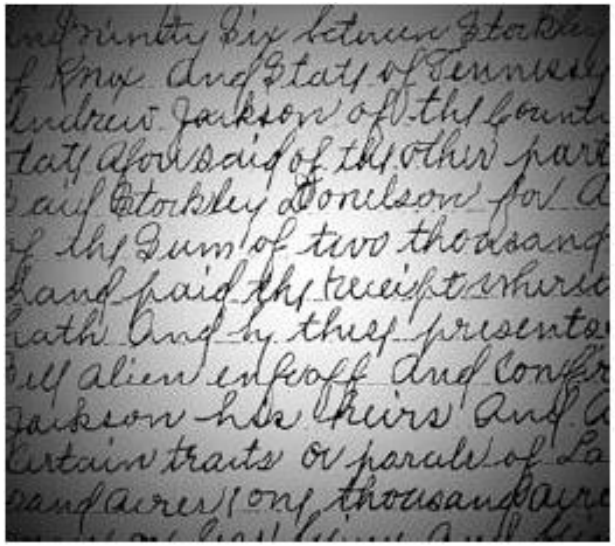
Segmentacija pomoću **lokalnog** praga

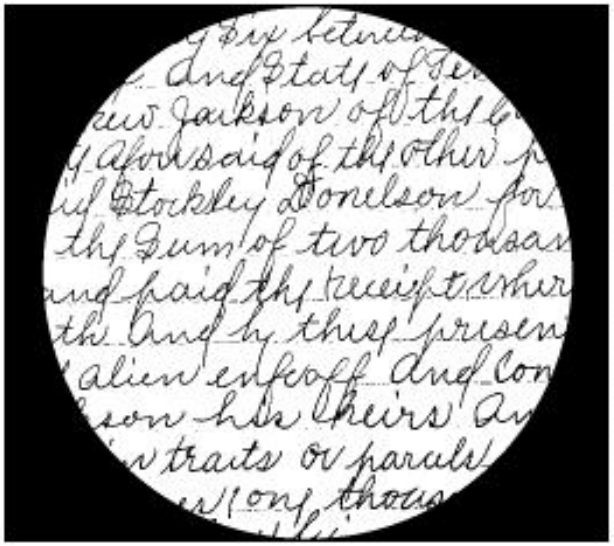


- Ovo je proizvelo kvalitetnu segmentaciju ne samo makaza nego i teksta

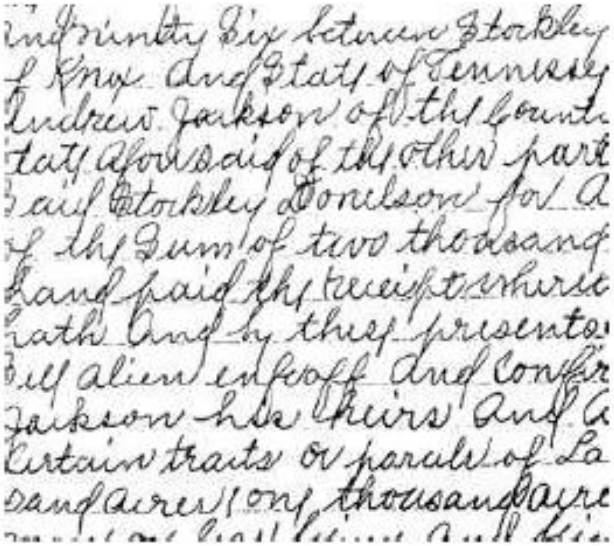
- Problem: nismo uspeli da uklonimo tekst sa slike

Segmentacija pomoću **lokalnog** praga





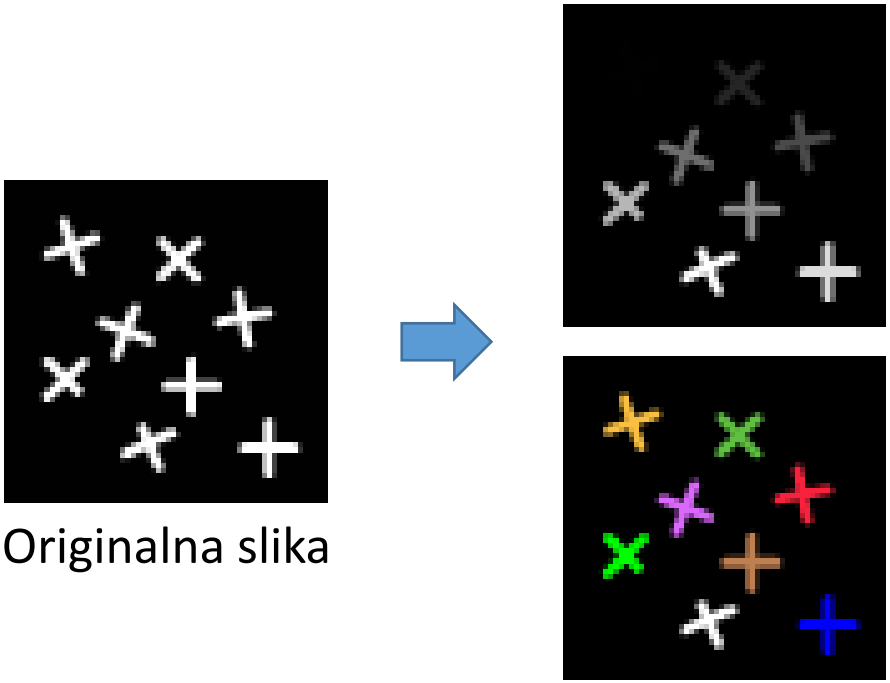
Globalni prag



Lokalni prag

Označavanje povezanih regiona

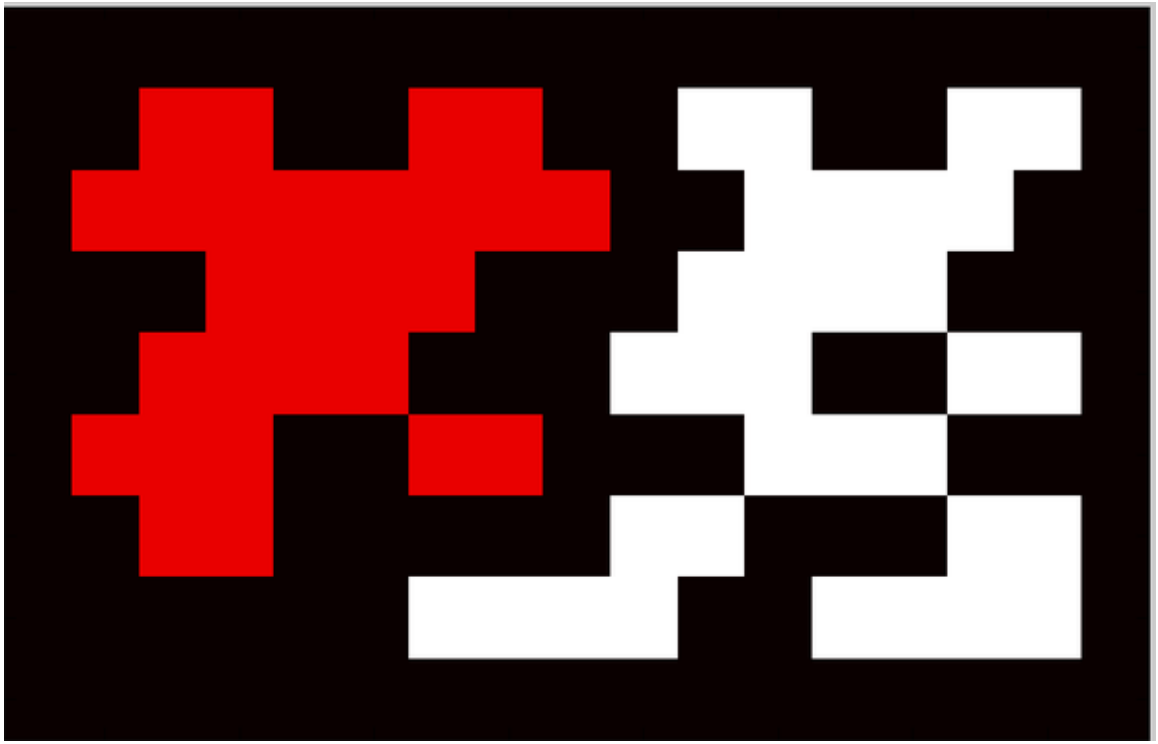
- Želimo da grupišemo piksele na slici u povezane regione (komponente)
- Kada smo otkrili sve grupe piksela koje postoje, svaki piksel ćemo anotirati *grayscale* nivoom (ili bojom) zavisno od komponente kojoj pripada



Originalna slika

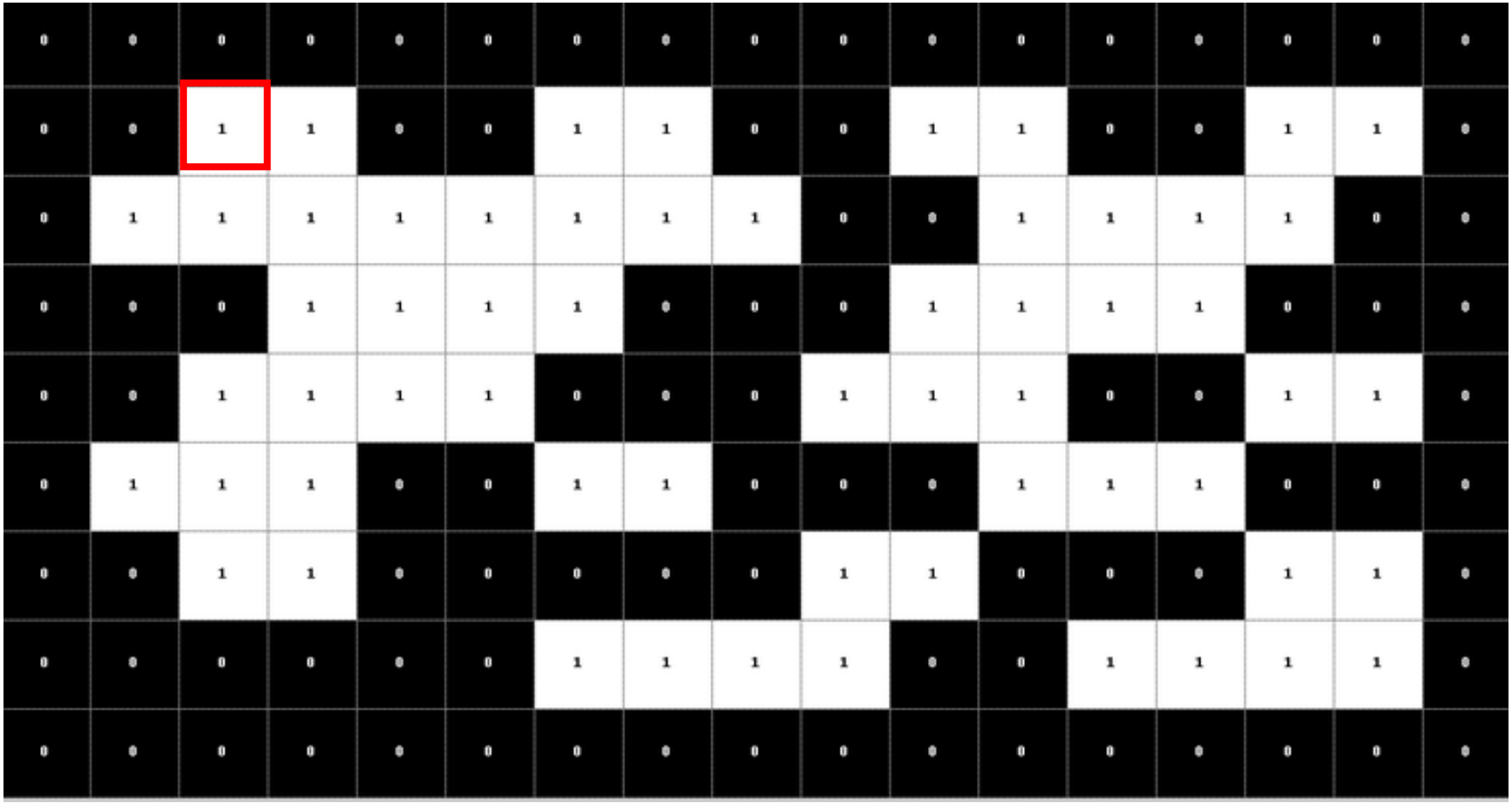
Označavanje povezanih regiona

- Ulaz u algoritam:
  - **Binarna** slika
  - Definicija susedstva (u ovom primeru 8-*connectivity*)
- Cilj:



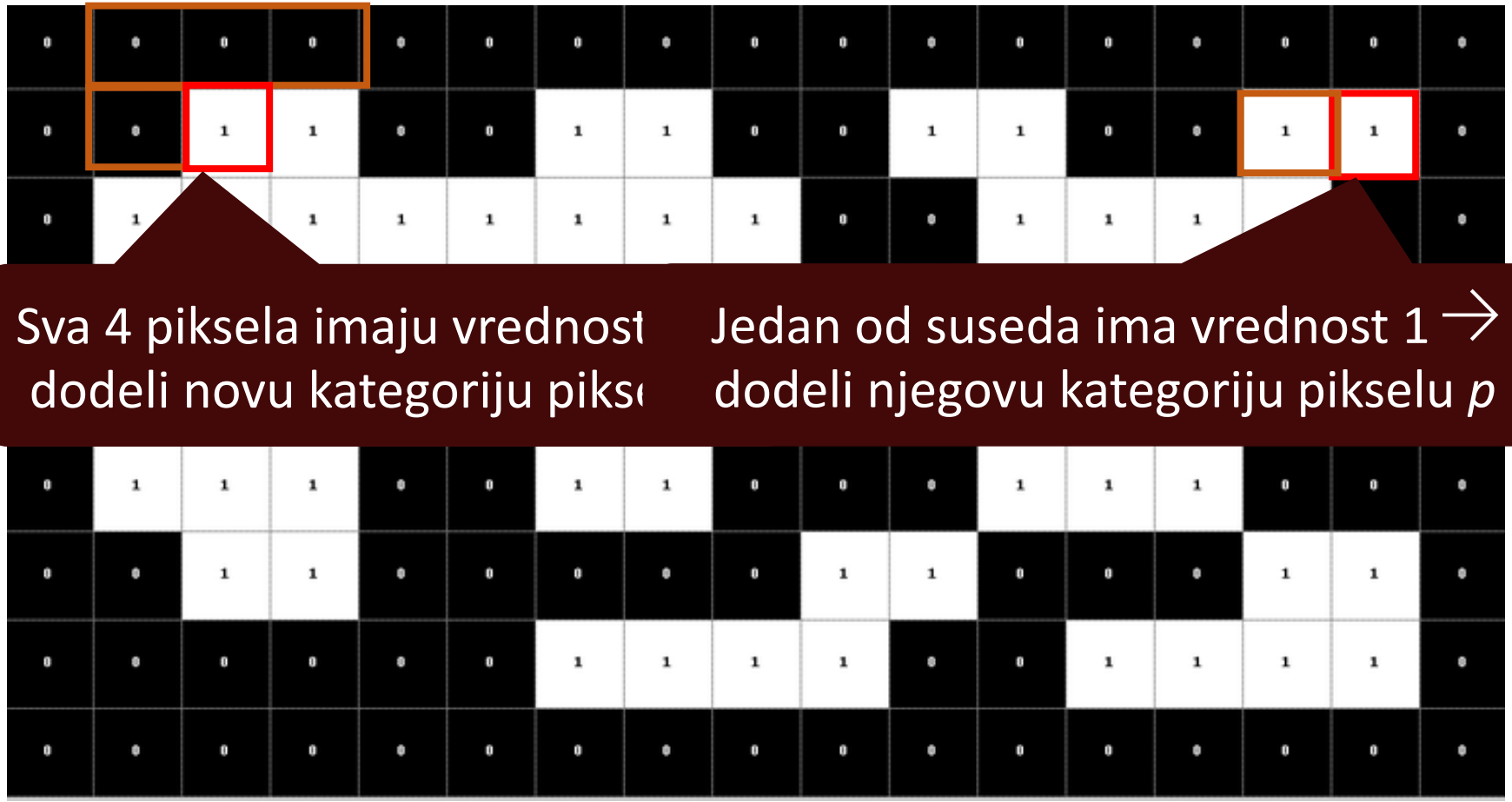
Označavanje povezanih regiona

- Skeniramo sliku piksel-po-piksel sve dok ne dođemo do piksela sa vrednošću 1



Označavanje povezanih regiona

- Razmotrimo 4 suseda ovog piksela koje smo već razmotrili

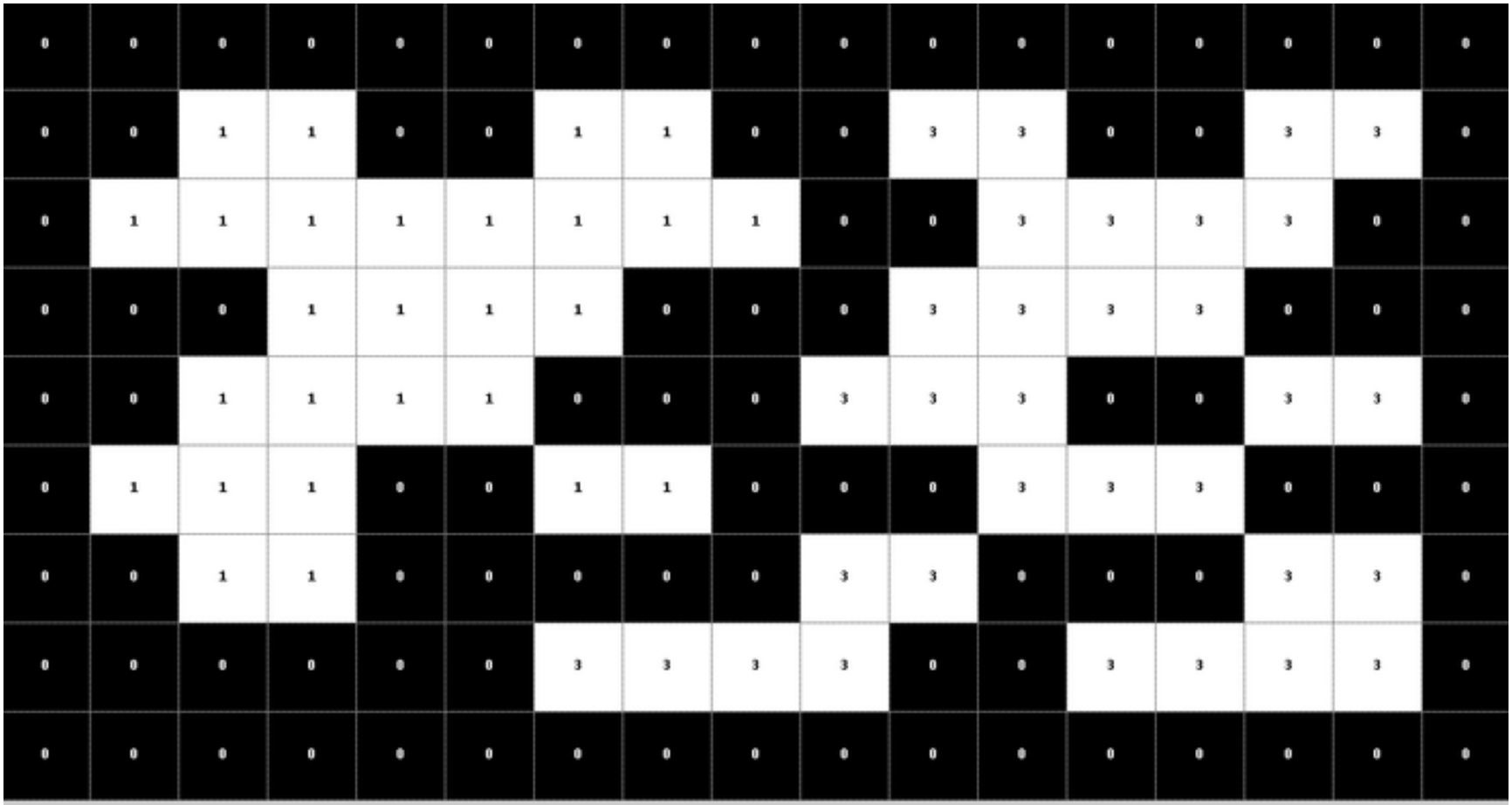


Označavanje povezanih regiona

- Razmotrimo 4 suseda ovog piksela koje smo već razmotrili

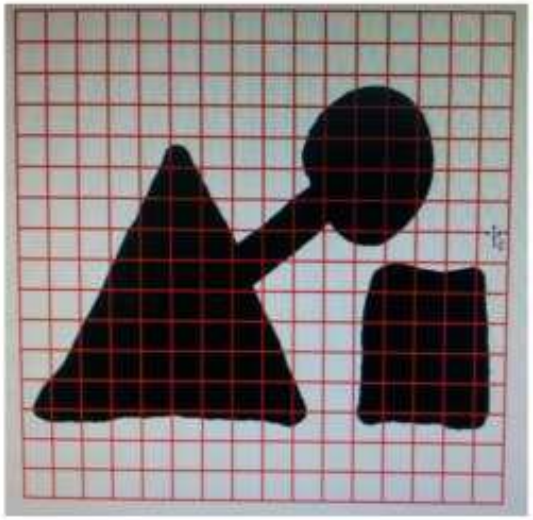
Označavanje povezanih regiona

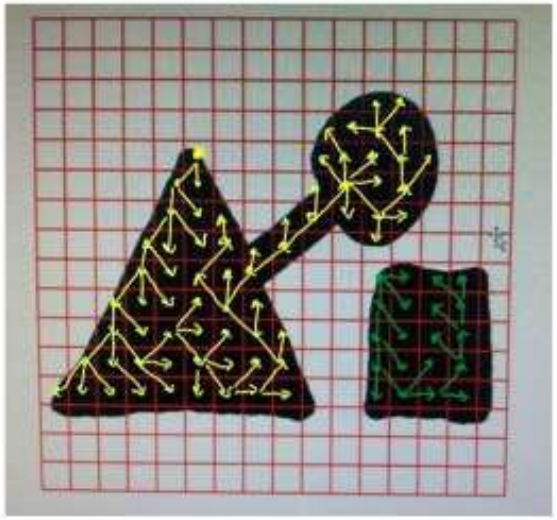
- Po završetku skeniranja piksela, dodeli jedinstvenu kategoriju svim ekvivalentnim kategorijama



Druga varijanta algoritma

- Krenemo od prvog obojenog piksela na koga naiđemo
- Zatim pretražujemo okolne piksele koji su iste boje
  - Svaki piksel koji posetimo dodamo u **listu posećenih piksela** kako ga ne bismo obilazili dva puta
  - Sve njemu susedne piksele iste boje stavljamo u **listu otvorenih ivica**
  - U svakoj iteraciji algoritma isprobaćemo po jedan piksel iz **liste otvorenih ivica**
- Možemo vršiti *breadth birst search* ili *depth first search* pretragu



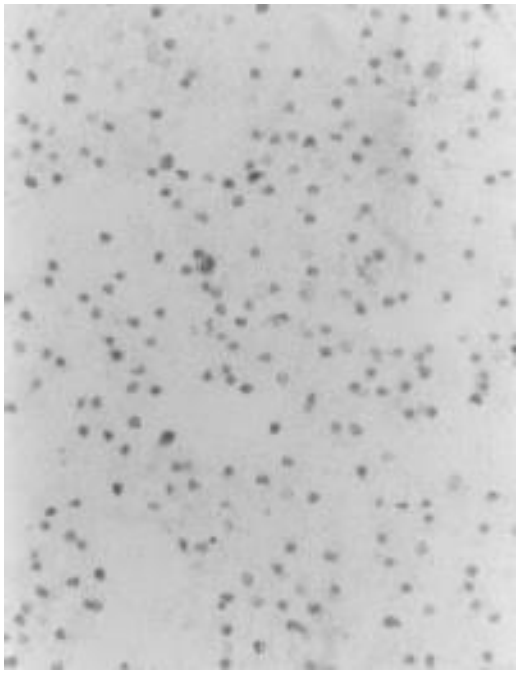


Primena pronalaženja povezanih regiona

- Ekstrakcija i obeležavanje različitih povezanih komponenti na slici se nalazi u centru mnogih metoda za automatsku analizu slike

- Na primer, ovim postupkom možemo:
  - Prebrojati objekte na realnim slikama
  - Odrediti gde se koji objekat nalazi
  - Odrediti koji pikseli pripadaju objektu...

Prebrojavanje objekata

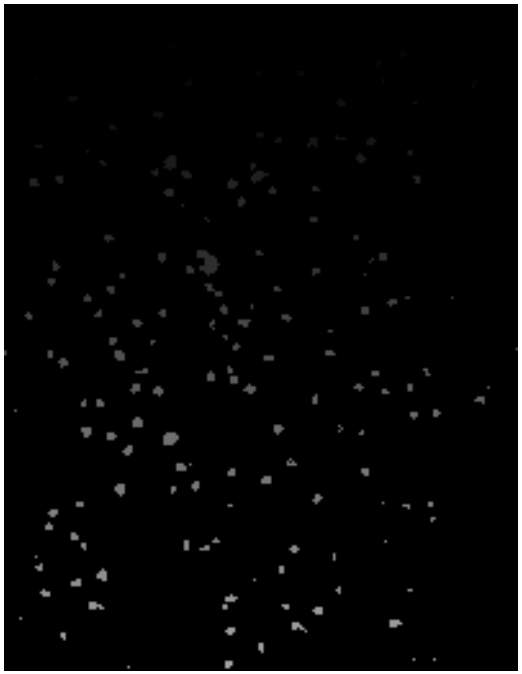


Originalna slika

Ćelije nervnog tkiva miševa: zdrave ćelije su srednjeg intenziteta sive boje, dok su mrtve ćelije guste i crne. Cilj je da se prebroje mrtve ćelije

Binarna slika

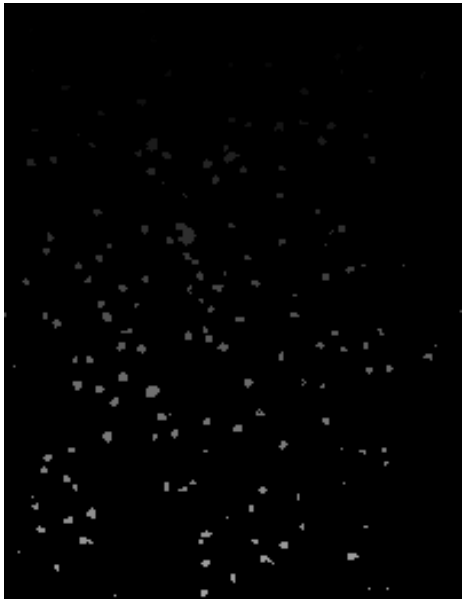
Primenjen je prag oko 150. Bele tačke odgovaraju mrtvim ćelijama sa originalne slike



Rezultati (obojeno nijansama sive)

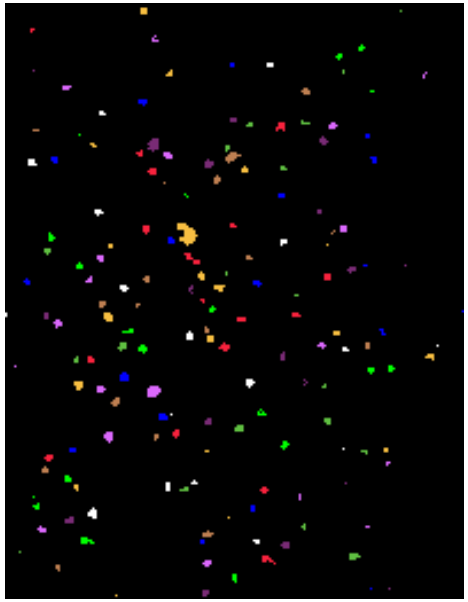
Algoritam je pronašao 163 komponente
Ovo je blisko originalnoj slici, ali ponegde je više različitih ćelija spojeno u istu komponentu

Prezentacija rezultata



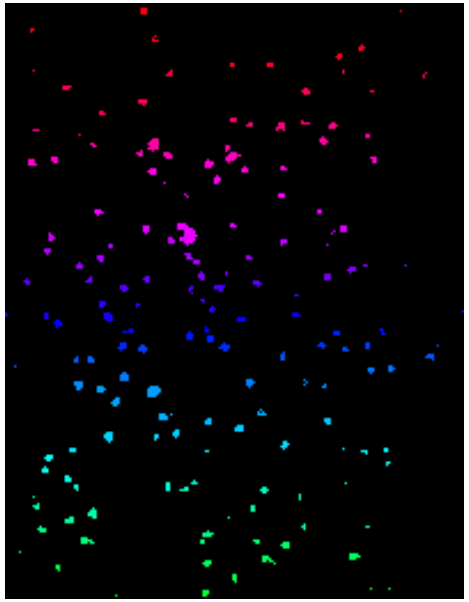
Rezultati (nijanse sive)

Loša vidljivost...
Možemo predstaviti bojama, ali teško je pronaći 163 lako raspoznatljive boje



8 boja koje je lako razlikovati

Ali gubimo informacije jer smo 163 kategorije mapirali na 8



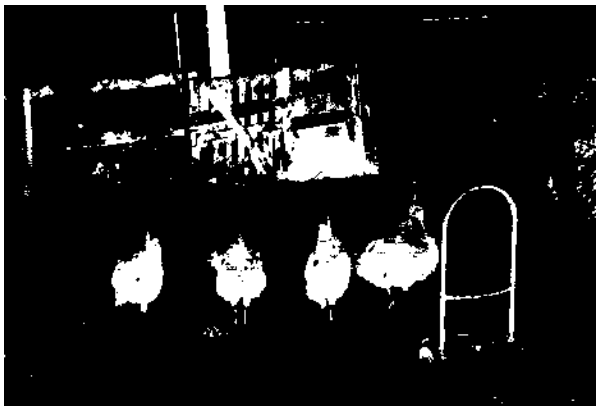
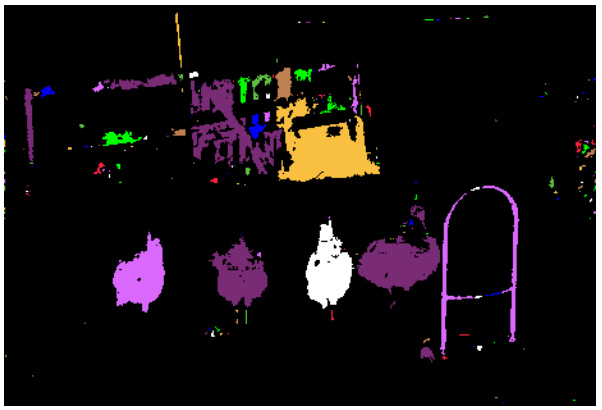
Svakom intenzitetu sive je dodeljena druga boja

Ne gubimo informacije, ali su mnoge nijanse slične pa je teško videti granice bliskih komponenti

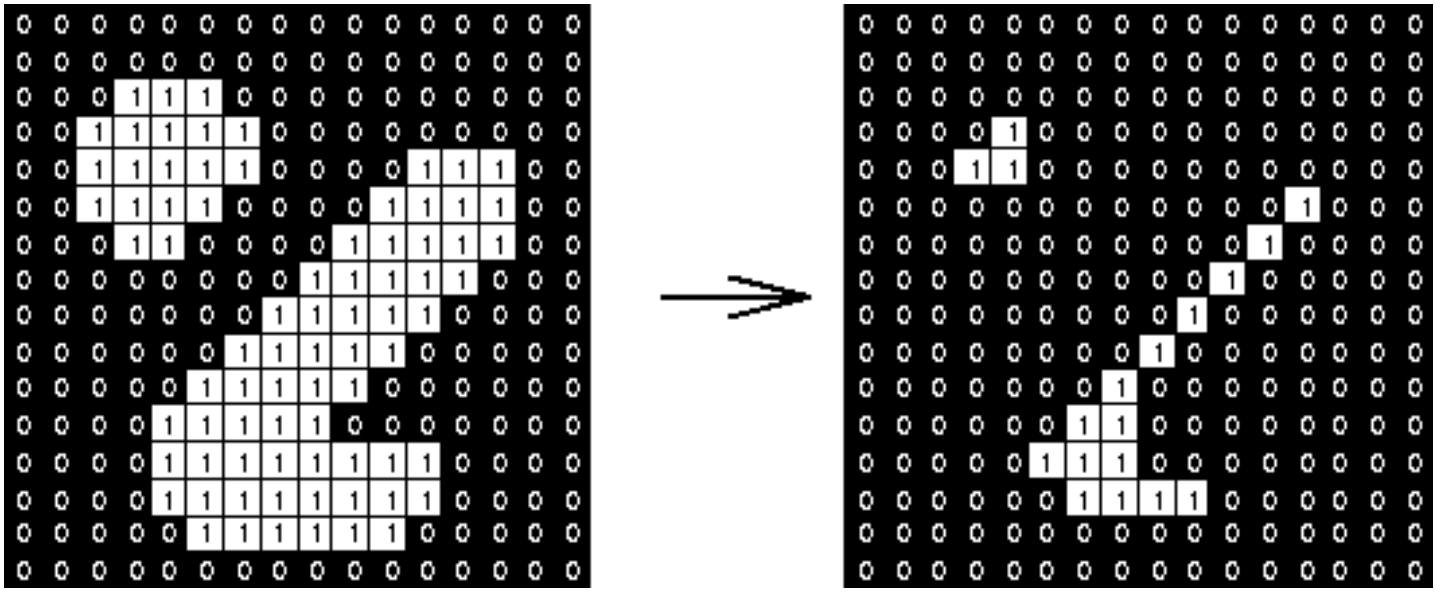
- Sofisticianije tehnike kombinuju ove dve tehnike – osiguraće se da bliske komponente uvek imaju boje koje se lako razlikuju

Prebrojavanje objekata

- Veći problem imamo kod prebrojavanja ćurki na sledećoj slici:

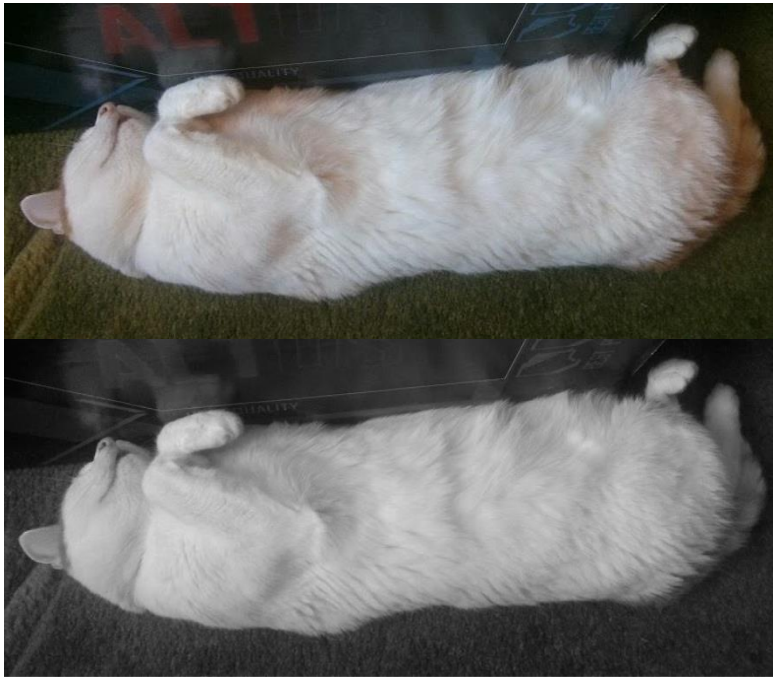
- Iako je svaka ćurka predstavljena posebnom komponentom, pronašli smo ukupno 196 komponenti od kojih nisu ćurke
- Ova dva primera su pokazala da je označavanje povezanih komponenti lak deo procesa automatske analize
- Glavni zadatak je da se dobije dobra binarna slika koja razdvaja objekte od interesa od pozadine

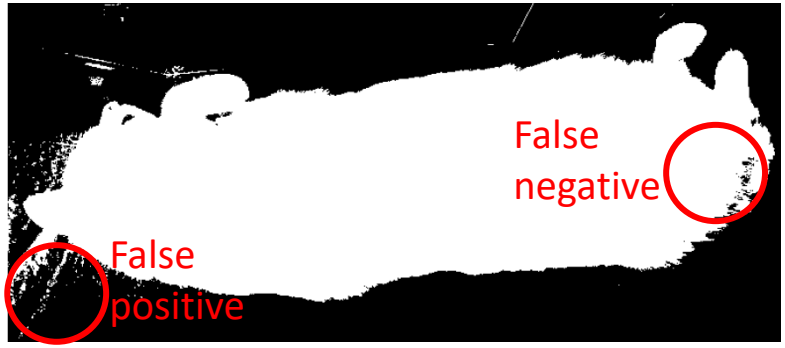


Morfološka obrada slike

Morfološka obrada slike – motivacija

- Segmentacija pomoću praga
  - Dobra za pronalaženje objekata koji su svetliji ili tamniji od pozadine
  - Izlaz su binarne slike (crni pikseli – pozadina, beli – objekat od interesa)
- Ali, gotovo uvek će rezultujuće binarne slike imati neke probleme u vidu crnog ili belog šuma

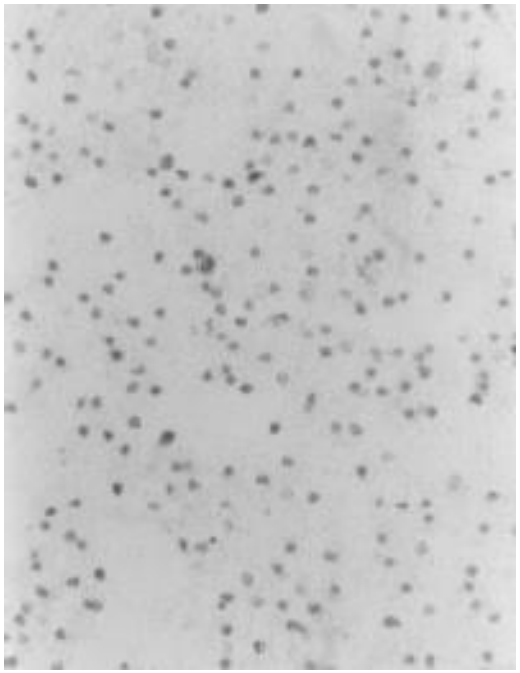




Morfološka obrada slike – motivacija

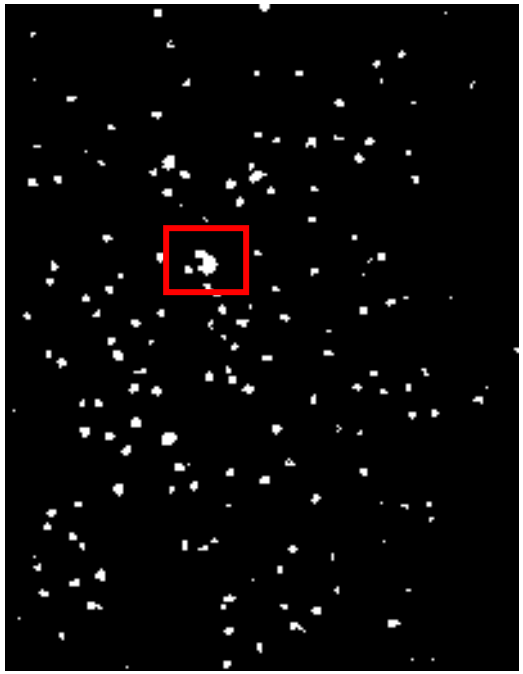
Slova lošeg kvaliteta – njihova obrada pre prosleđivanja neuronskoj mreži bi možda pobojšala njene preformanse

Morfološka obrada slike – motivacija



Originalna slika

Ćelije nervnog tkiva miševa: zdrave ćelije su srednjeg intenziteta sive boje, dok su mrtve ćelije guste i crne. Cilj je da se prebroje mrtve ćelije



Binarna slika

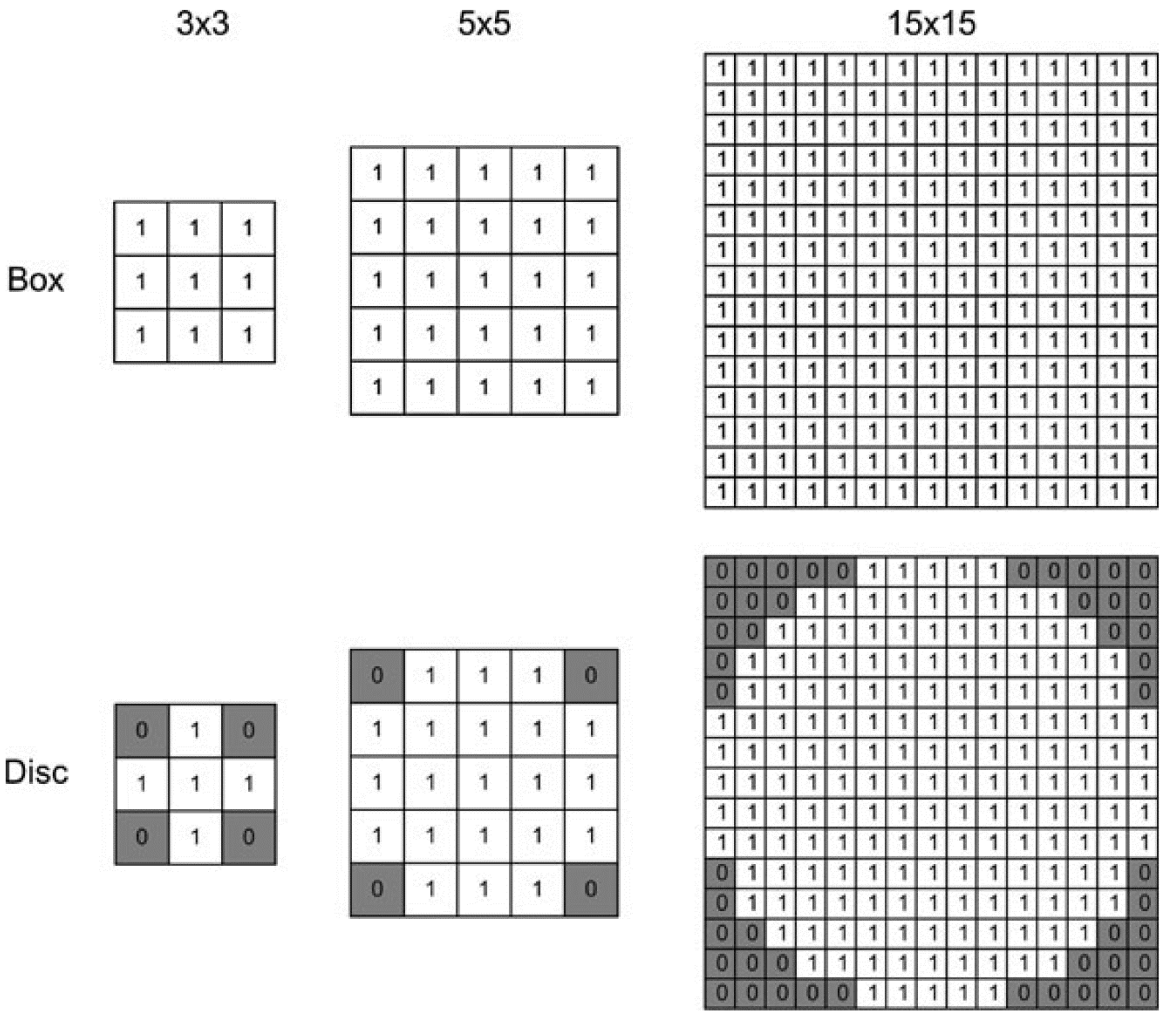
Primenjen je prag oko 150. Bele tačke odgovaraju mrtvim ćelijama sa originalne slike

- Više objekata spojenih u jedan zbog belog šuma
- Može da predstavlja problem za prebrojavanje objekata

Morfološka obrada slike

- Ulaz:
  - Binarna slika
  - Strukturni element
- Izlaz:
  - Binarna slika iste veličine kao ulazna

- Strukturni element
  - Matrica sastavljenu od 0 i 1
  - Definiše okolinu piksela



Morfološke operacije

- Skup operacija za obradu slika koje se baziraju na oblicima
- Vrednost svakog piksela izlaza će biti bazirana na poređenju odgovarajućeg piksela ulazne slike sa svojom okolinom
- Odabirom veličine i oblika susedstva, možemo kreirati morforloške operacije osetljive na određene oblike na ulaznoj slici

Morfološke operacije

- Osnovne binarne operacije su:
  - Dilacija – dodaje piksele na ivicu objekata na slici
  - Erozija – uklanja piskele sa ivica objekata na slici

- Njihovom kombinacijom možemo konstruisati operatore za:
  - Otvaranje
  - Zatvaranje
  - Detekciju ivica

Dilacija (*Dilation*)

$$A \oplus B = \left\{ z | (\hat{B})_z \cap A \neq \varnothing \right\}$$

- Logička operacija OR između elemenata neke okoline
  - Dovoljno je da BAR jedna tačka iz okolike bude 1 pa da tačka u sredini bude 1
- Količina piksela sa vrednošću 1 se povećava i rupe objektu (manje od strukturnog elementa) se popunjavaju (uklanja *false negatives*)

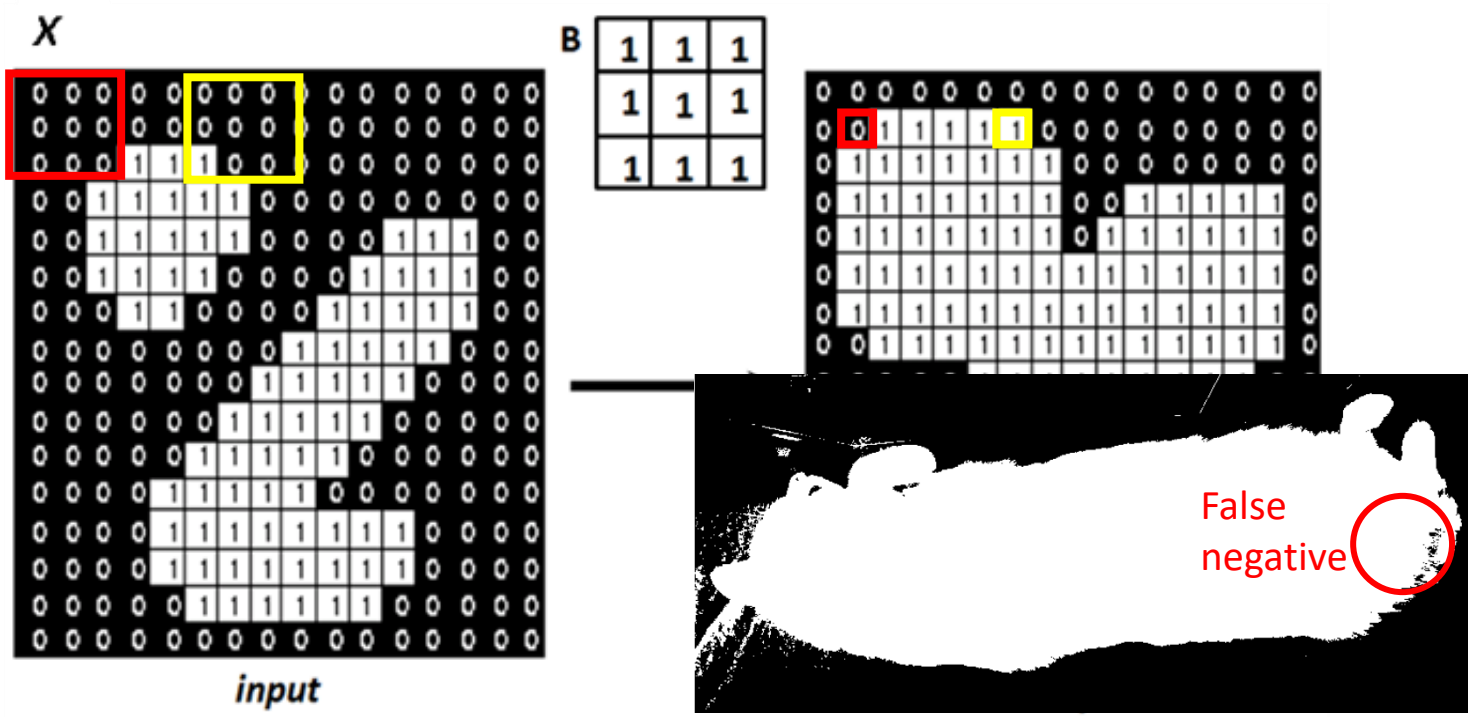
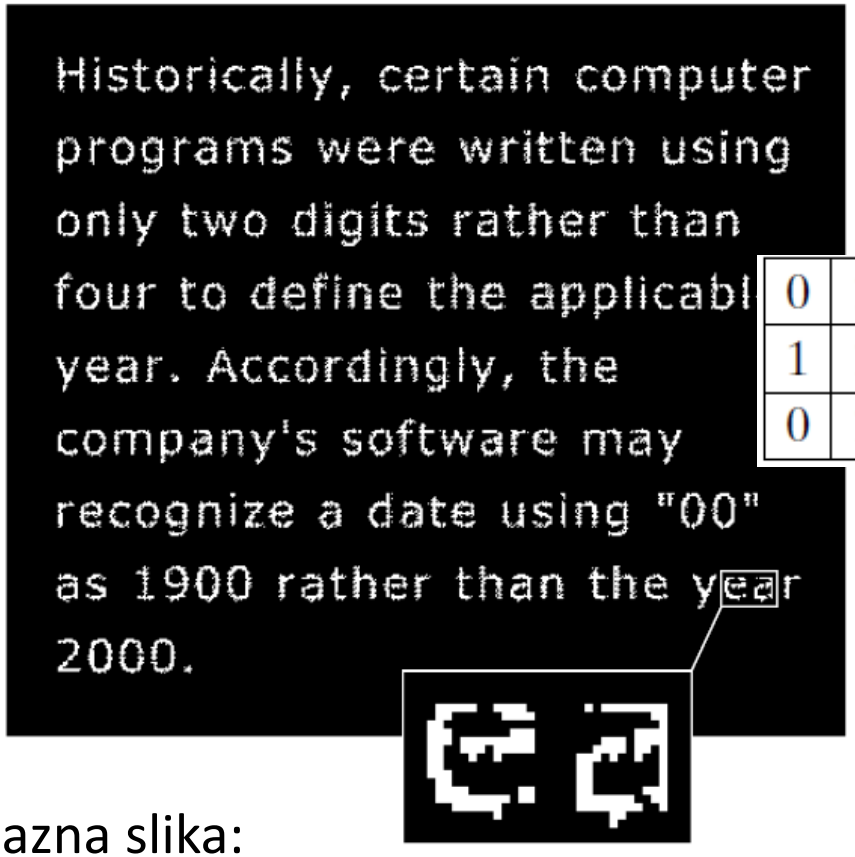
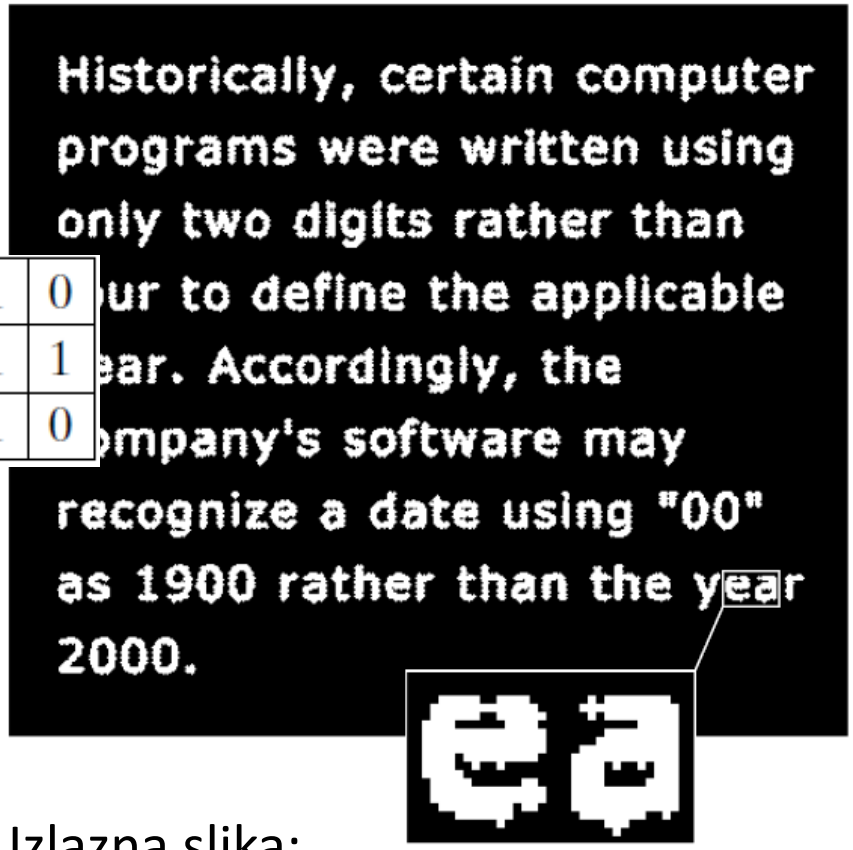


Figure 1. Effect of dilation using a 3X3 square structural element B.

Dilacija - primer



0	1	0
1	1	1
0	1	0



Ulazna slika:

- Loš kvalitet slova
- Znamo da je maksimalna veličina prekida 1-2 piksela – ne treba nam veliki strukturni element

Izlazna slika:

- Rupe su popunjene
- Možemo primeniti algoritam za pronalaženje povezanih komponenti za detekciju slova

Erozija (*Erosion*)

$$A \ominus B = \{z | (B)_z \cap A^c = \varnothing\}$$

- Logička operacija AND između elemenata neke okoline
  - Piksel izlazne slike će biti 1 ukoliko je svaki piksel u okolini tog piksela na ulaznoj slici ima vrednost 1. U suprotnom, biće 0
- Količina piksela sa vrednošću 1 se smanjuje i veličina objekta se smanjuje (uklanja *false positives*)

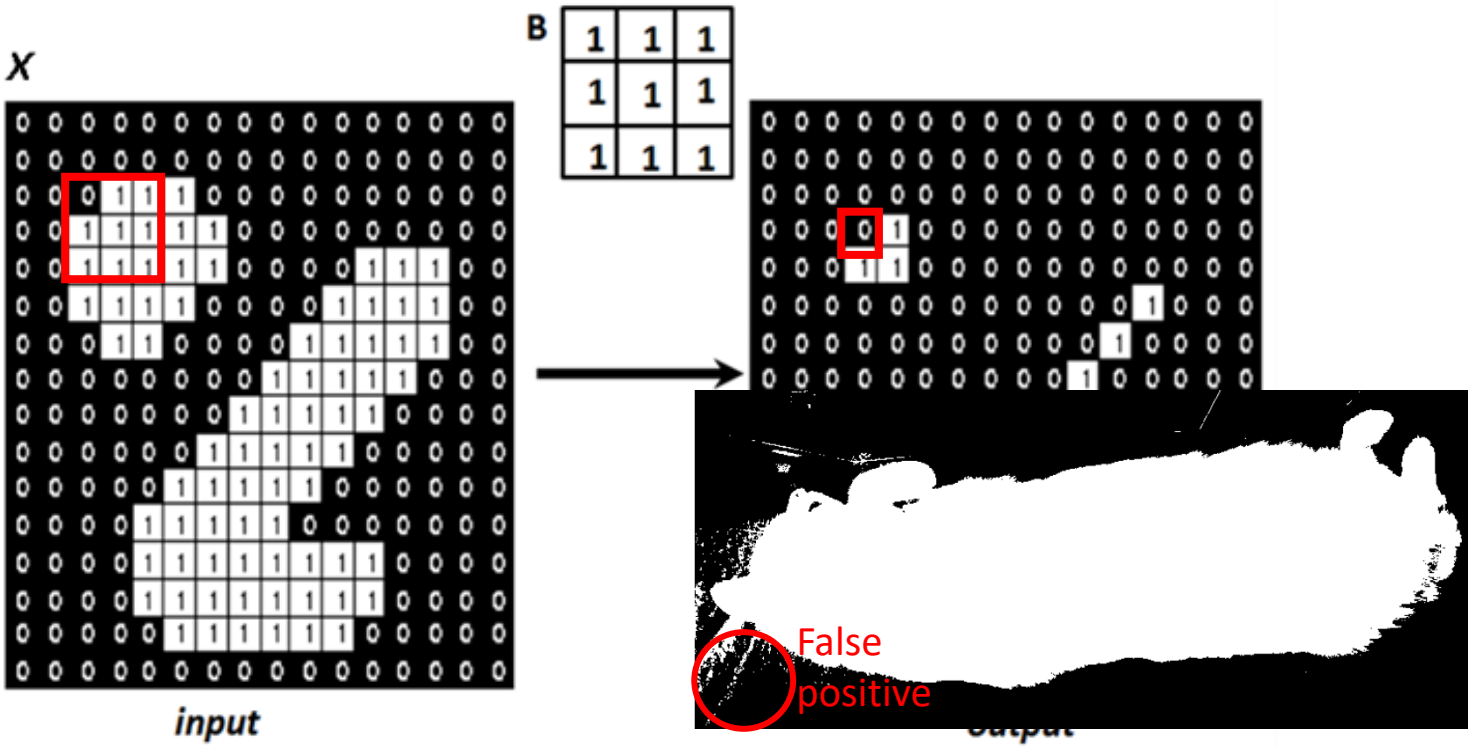
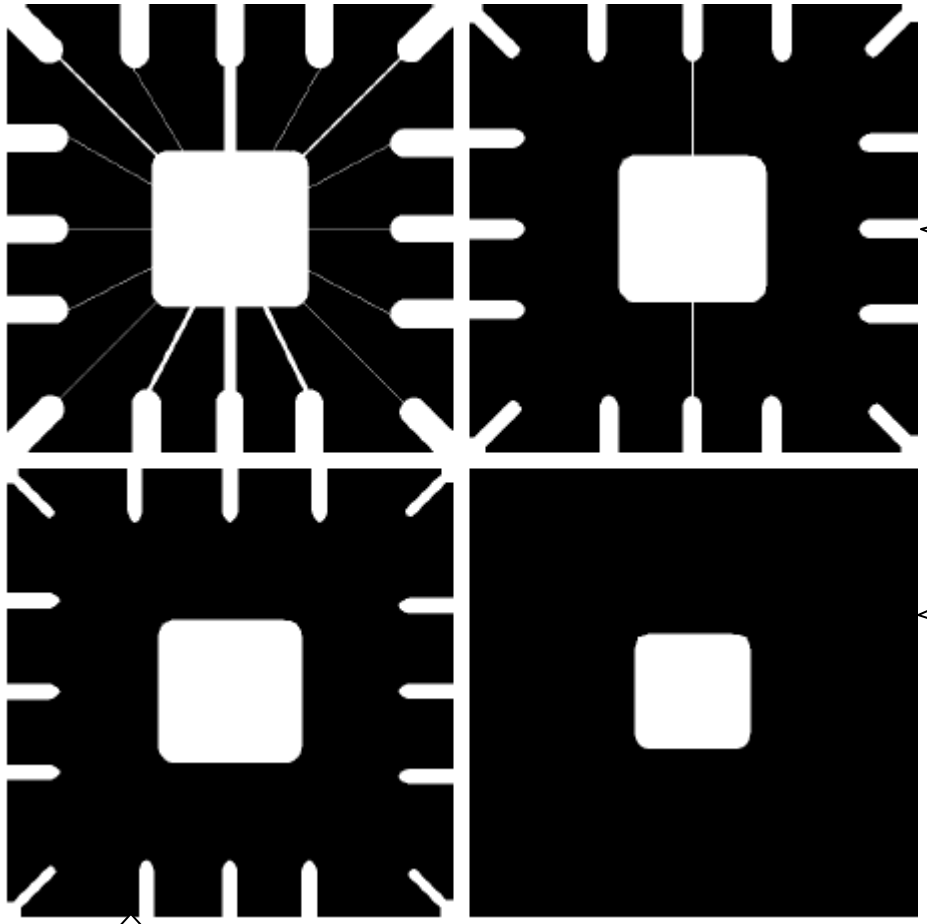


Figure 2. Effect of erosion using a 3X3 square structural element B.

Erozija – primer



15×15

Želimo da uklonimo tanke linije sa slike

11×11

(vertikalne linije nisu uklonjene jer su šire od 11 piksela)

45×45

(dalje uvećavanje strukturnog elementa uklanja veće komponente)

Šta bismo želeli?

- Šta možemo:
  - Dilacija – možemo da premostimo neželjen procepe ali uvećavamo objekte
  - Erozija – uklanjamo neželjene tanke linije ali smanjujemo objekte

- Ali, želeli bismo približno iste veličine objekata na slici uz uklanjanje sitnih problema
  - Otvaranje – prvo erozija pa onda dilacija
  - Zatvaranje – prvo dilacija pa onda erozija

Otvaranje

$$A \circ B = (A \ominus B) \oplus B$$

- Koristimo *isti strukturni element* da izvršimo prvo eroziju pa onda dilaciju
- Efekat: Izglađivanje (*smoothing*) ivica objekta: uklanjanje tankih mostova i eliminisanje tankih struktura

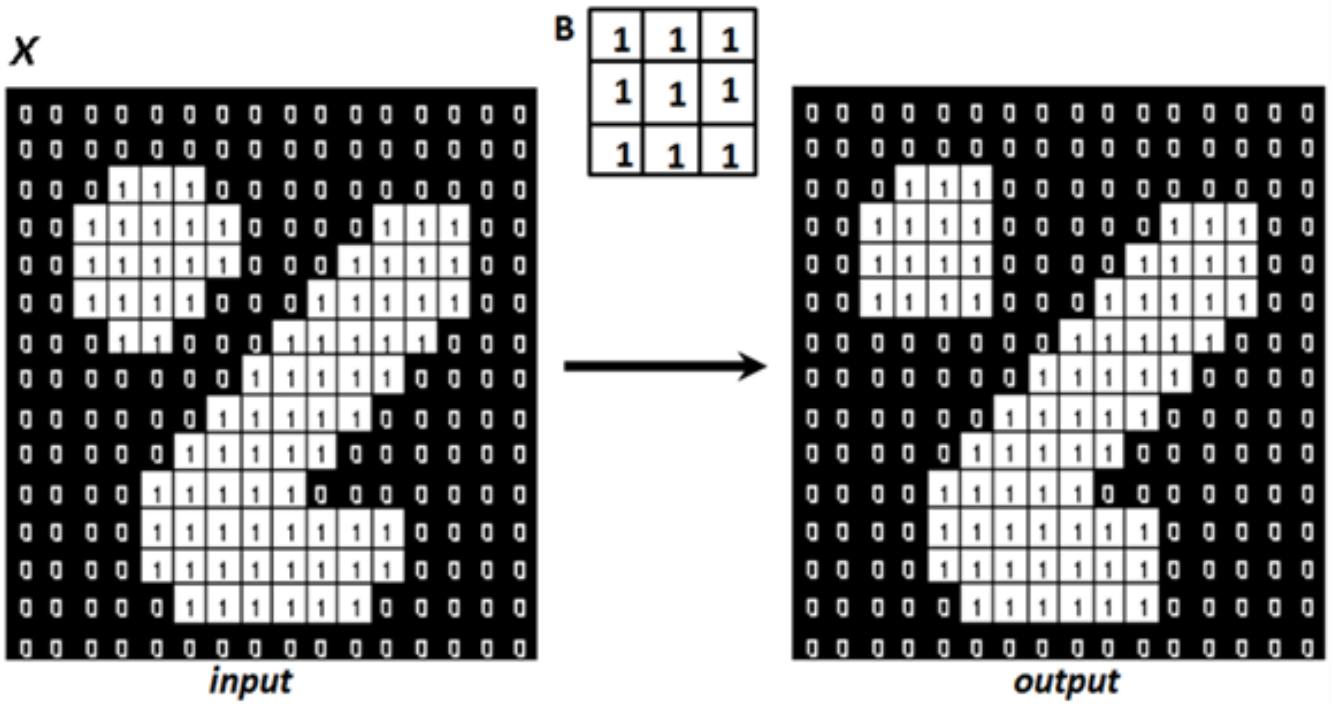
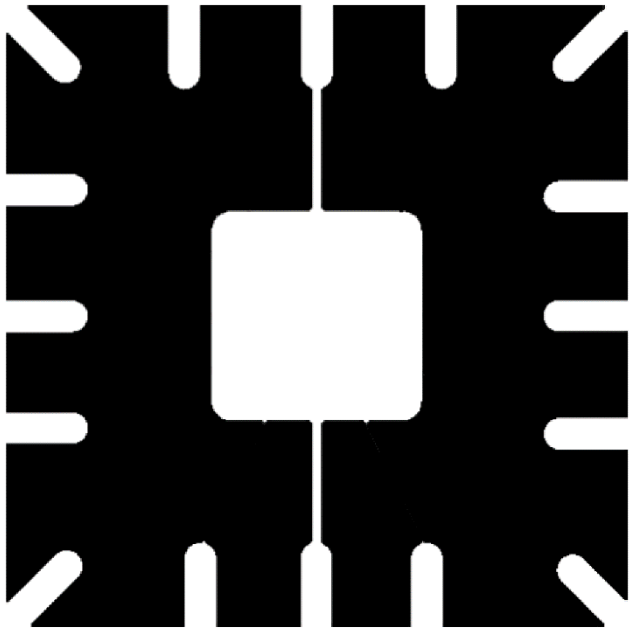


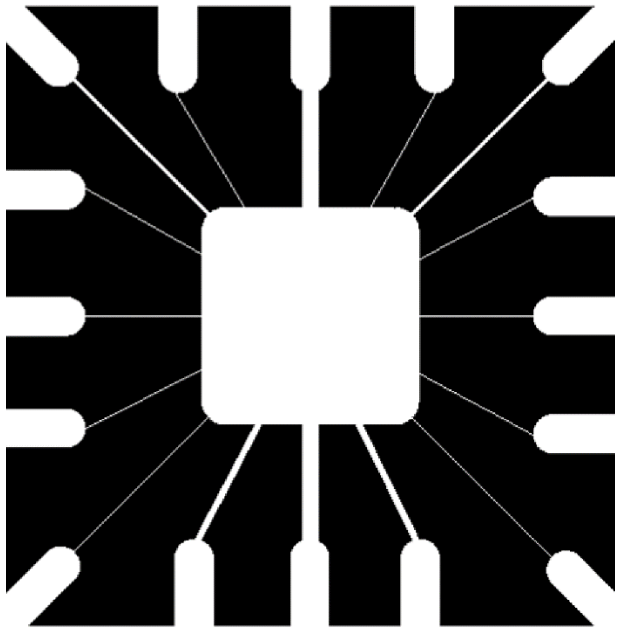
Figure 3. Effect of opening using a 3X3 square structural element B.

Erozija i Otvaranje

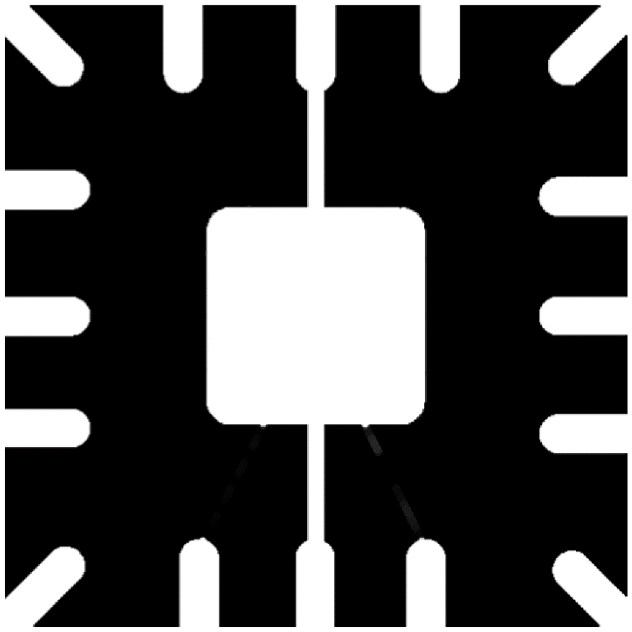


Erozija

Linije tanje od strukturnog elementa su uklonjene, ali su objekti na slici smanjeni



Originalna slika



Otvaranje

Linije tanje od strukturnog elementa su uklonjene, a veličina objekata sačuvana

Zatvaranje (*Closing*)

$$A \bullet B = (A \oplus B) \ominus B$$

- *Koristimo isti strukturni element* da izvršimo prvo dilaciju pa onda eroziju
- Efekti: Izglađivanje objekata dodavanjem piksela: smanjuje rupe u objektima manje od veličine strukturnog elementa, povezuje strukture odvojene malim prekidima,...

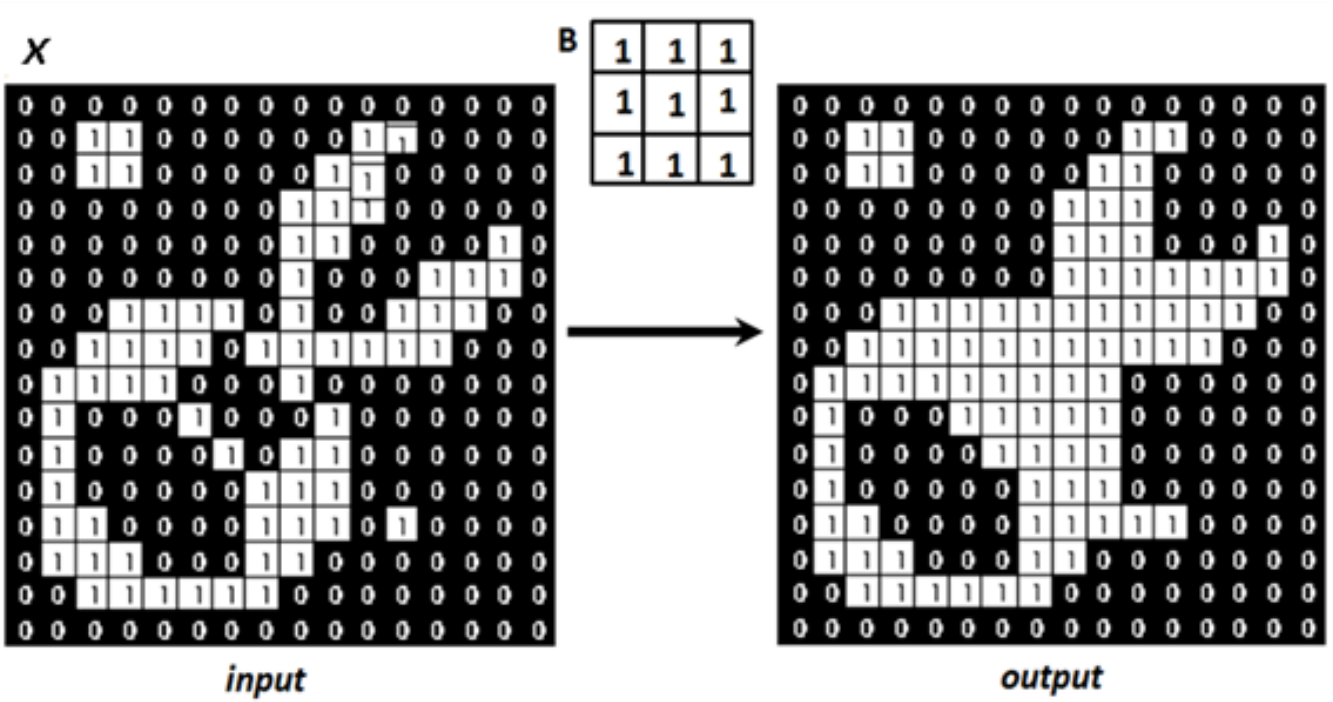
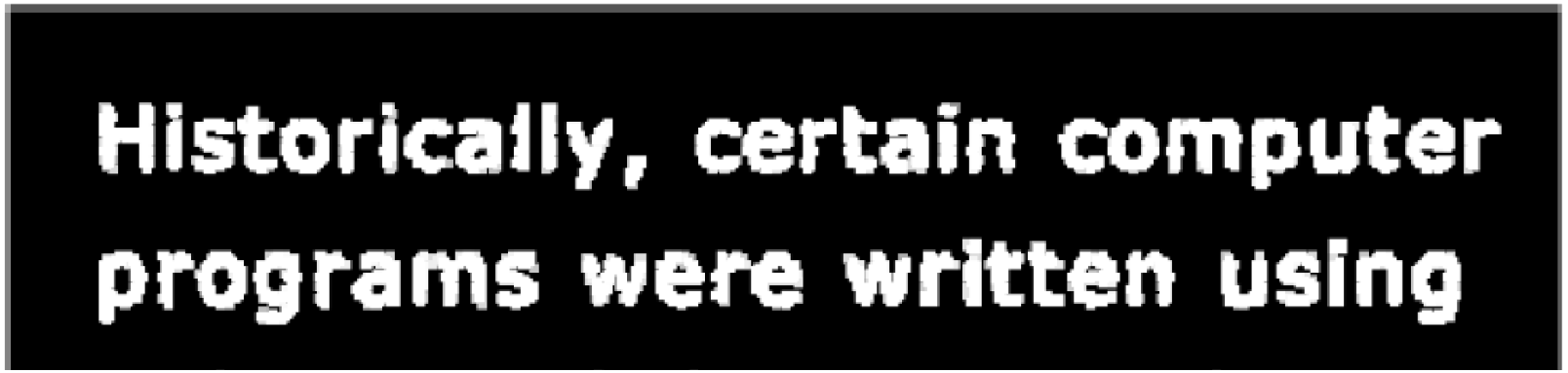
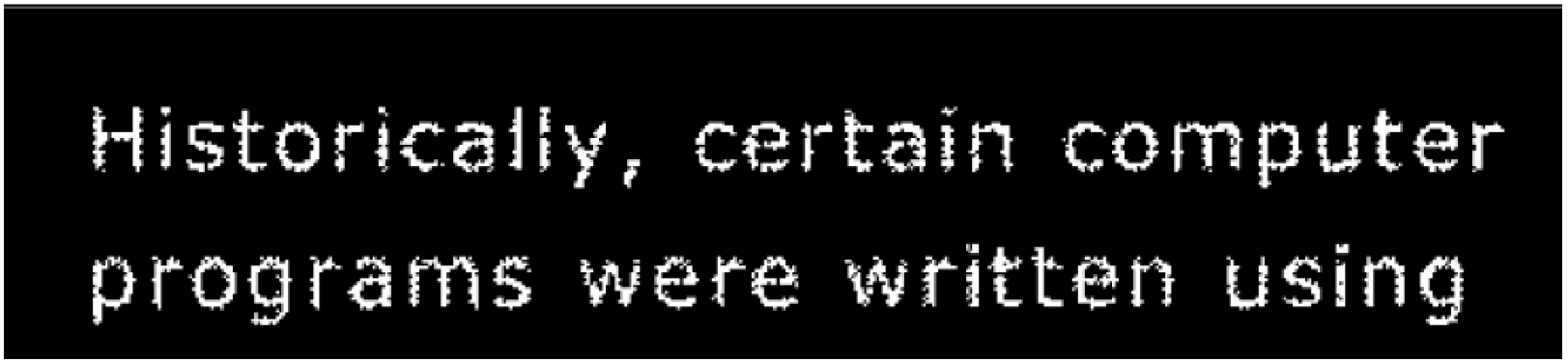


Figure 4. Effect of closing using a 3X3 square structural element B.

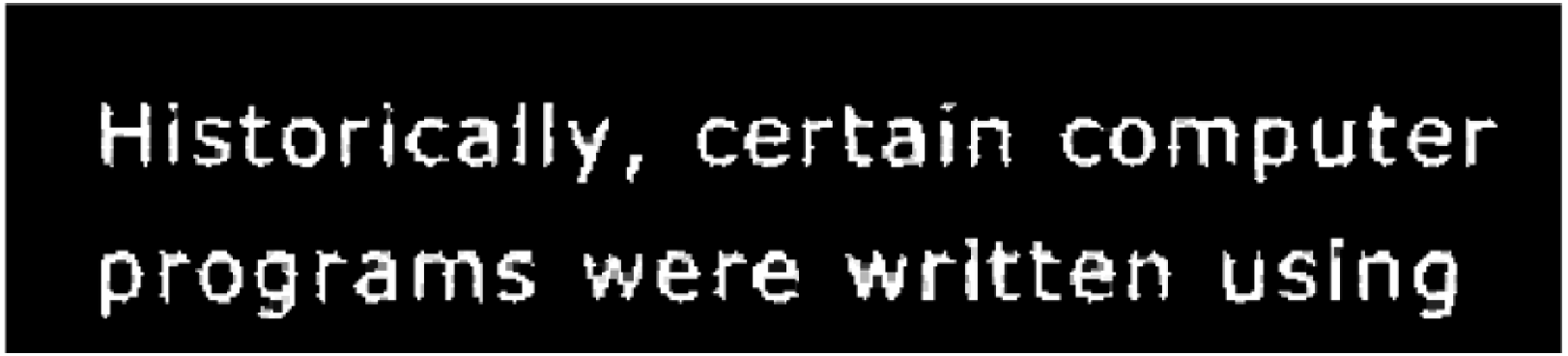
Dilacija i zatvaranje



Dilacija



Originalna slika



Zatvaranje

Zatvaranje

- Morfološkom operacijom zatvaranja možemo da izbacimo tekst sa slike

Primer primene

a b
d c
e f

FIGURE 9.11
(a) Noisy image.
(b) Structuring element.
(c) Eroded image.
(d) Opening of A.
(e) Dilation of the opening.
(f) Closing of the opening.
(Original image courtesy of the National Institute of Standards and Technology.)

Uticaj strukturnog elementa

- I kod otvaranja i kod zatvaranja, jačina izglađivanja zavisi od veličine strukturnog elementa
  - Veličinu treba odrediti u skladu sa veličinom objekata od interesa
  - Za velike objekte, veći strukturni element je preferiran, dok je za manje neophodan manji radi očuvanja finih strukturnih detalja objekta

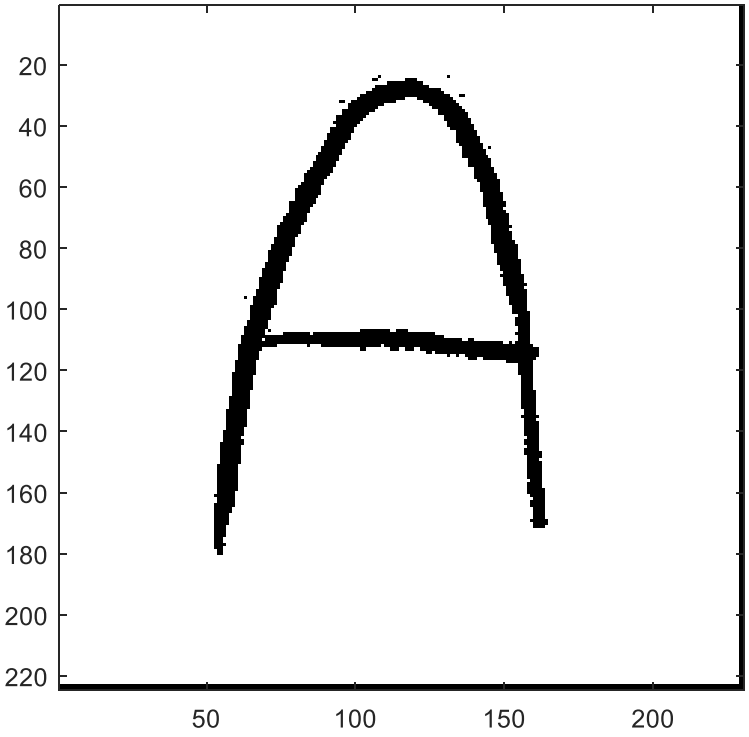
- Poželjan oblik strukturnog elementa takođe zavisi od geometrijskih oblika objekata na slici
  - Na medicinskim slikama obično postoji veoma malo pravih linija i objekata, pa se preferira oblik diska
  - Na satelitskim snimcima imamo objekte poput zgrada, puteva,... Zato bi bilo dobro da je strukturni element u obliku pravougaonika (odgovarajuće orijentacije)

Detekcija ivica

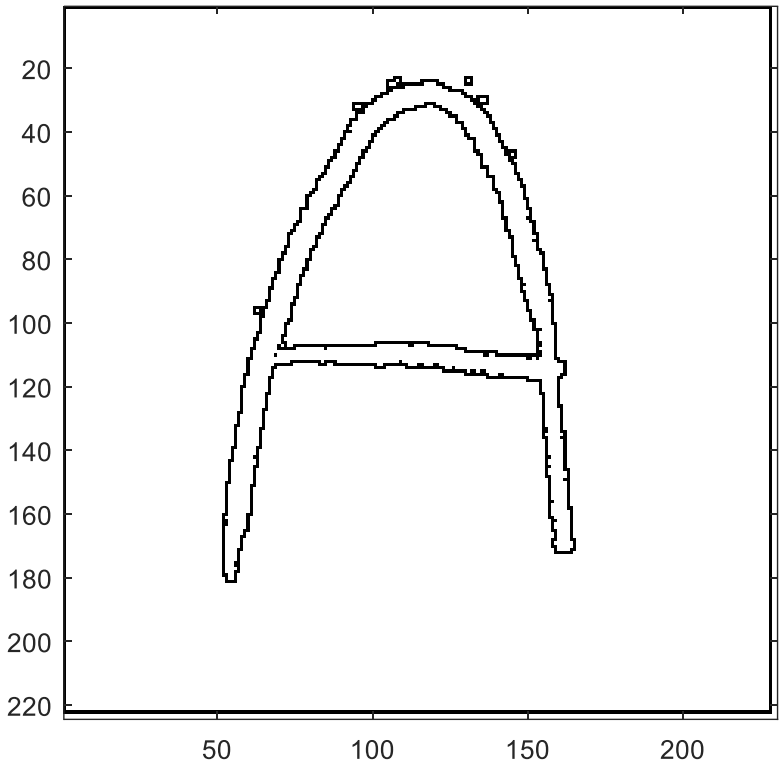
- Ima mnogo kombinacija dilacije i erozije
- Možemo ih iskoristiti da npr. pronađemo ivice objekta

Detekcija ivica

$f =$

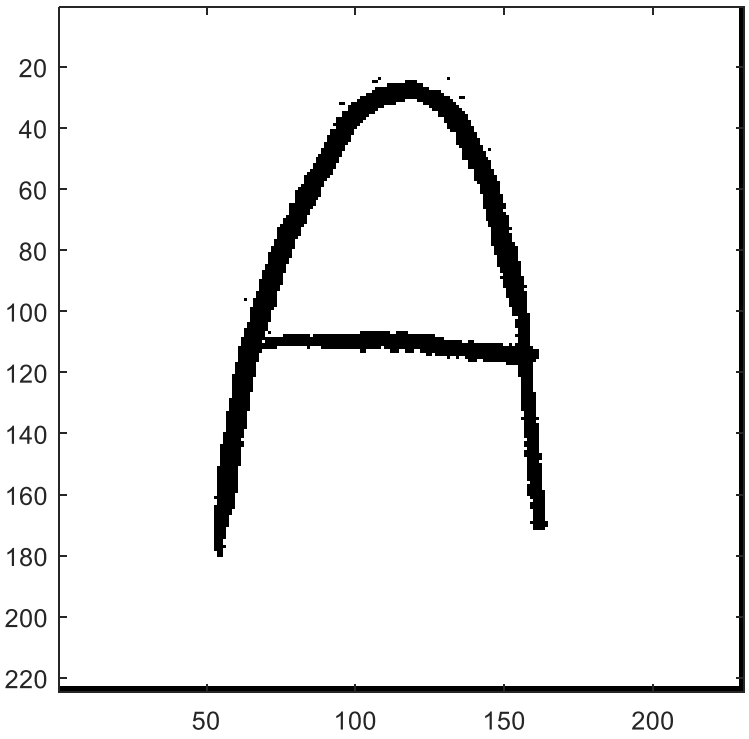


Detekcija ivica$_1$ = Dilacija $-f$

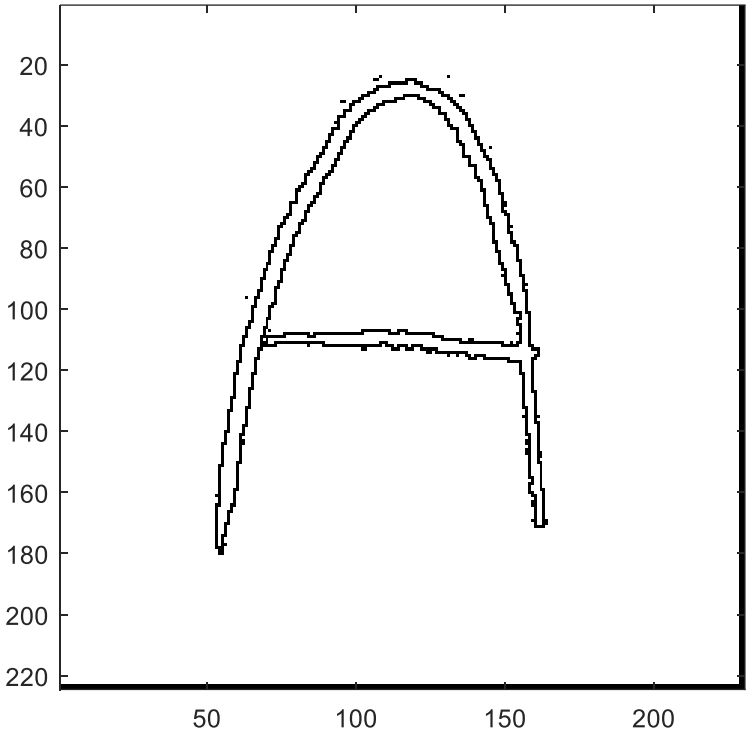


Detekcija ivica

$f =$



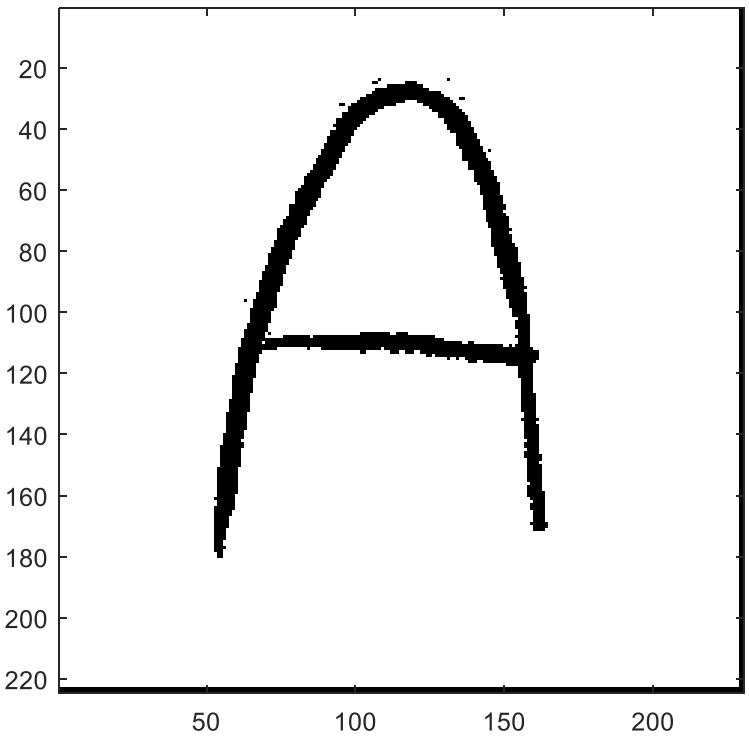
Detekcija ivica$_2$ = f − Erozija

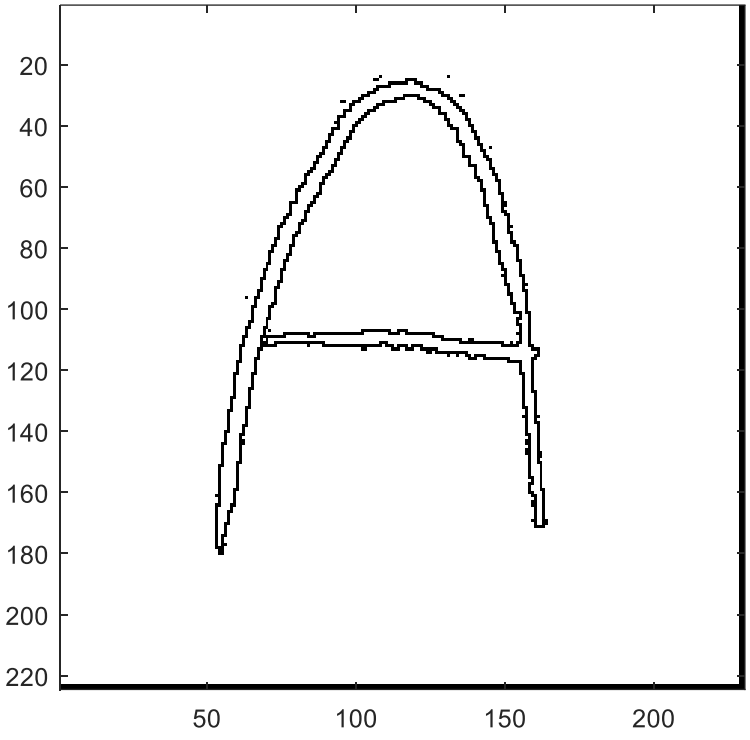


Detekcija ivica

Detekcija ivica$_3$ = Dilacija − Erozija

$f =$

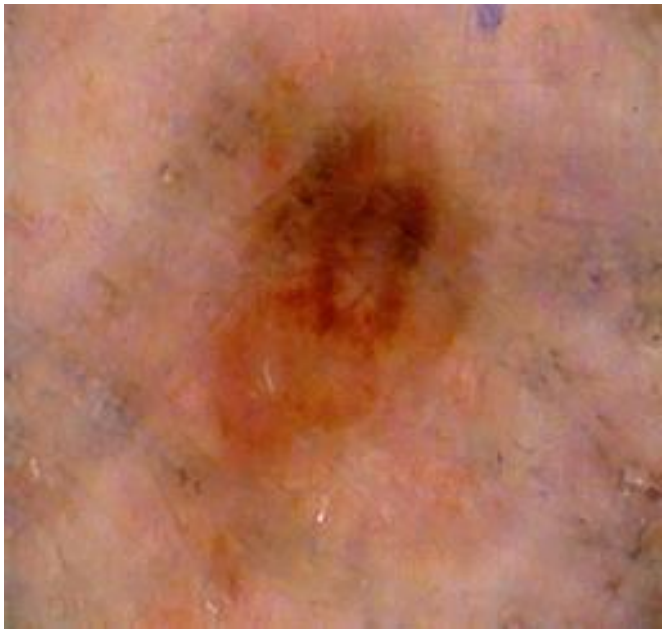




Proširenje na *grayscale* slike

- Dilacija – maksimum piksela susedstva
- Erozija – minimum piksela susedstva

Primena – *DullRazor* tehnika



- Klasifikacija mladeža na benigne i melanome
- *DullRazor* – kombinacija dilacije, erozije, oduzimanja slika, uklanjanja šuma, primene maske,...

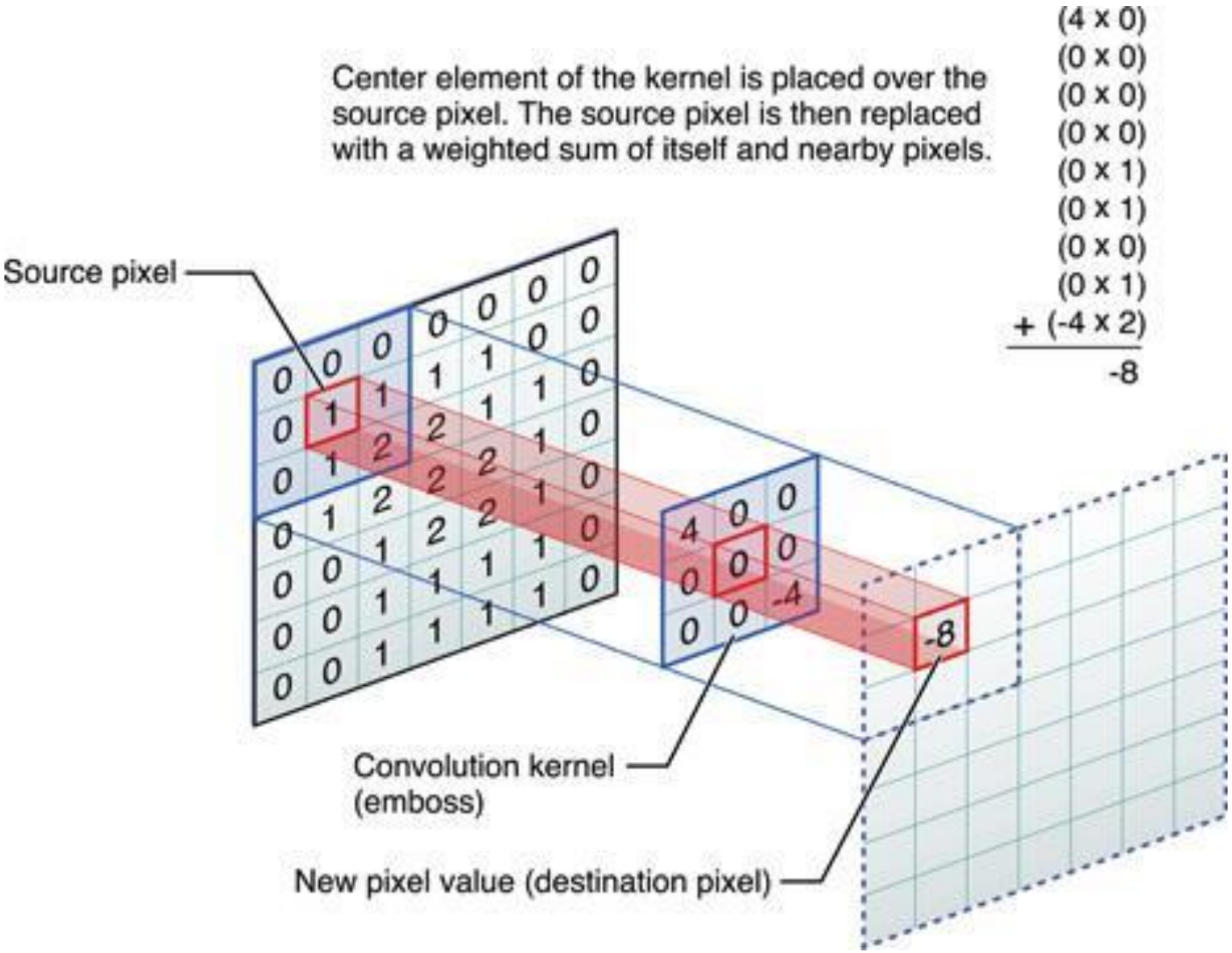
- Lee, Tim, et al. "Dullrazor®: A software approach to hair removal from images." *Computers in biology and medicine* 27.6 (1997): 533-543
- Kiani, Kimia, and Ahmad R. Sharafat. "E-shaver: An improved DullRazor® for digitally removing dark and light-colored hairs in dermoscopic images." *Computers in biology and medicine* 41.3 (2011): 139-145.

Konvolucija

Konvolucija

- Konvolucija je matematička operacija definisana nad dve funkcije koja proizvodi treću funkciju
- Pošto smo definisali slike kao funkcije, onda je možemo primeniti na slikama
- Konvolucija je važna operacija za *signal processing*
  - Može se primeniti za dva signala (1D) ili dve slike (2D)
  - Razmišljajte o jednom signalu kao ulaznom, a drugom (kernelu) kao o filteru koji se primenjuje da bismo dobili izlaz

Konvolucija



Konvolucija

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

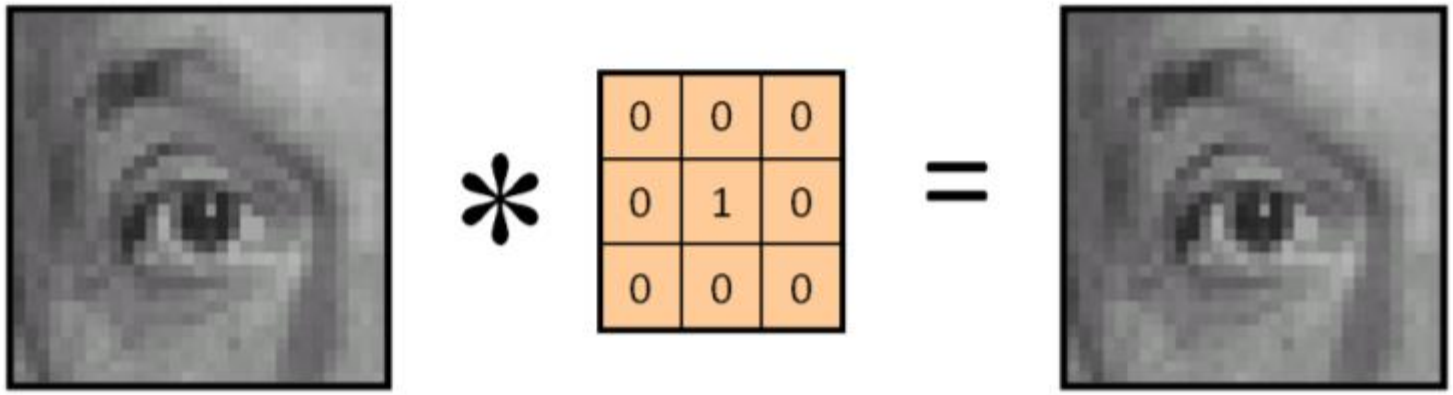
Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

Konvolucija

- „Prevlačimo“ kernel preko svakog piksela na slici
- Vrednost piksela na izlaznoj slici dobijamo tako što množimo odgovarajuće vrednosti ulazne slike i kernela, a zatim sve umnoške saberemo
- Npr. sledeći kernel ništa ne menja u originalnoj slici:



- Često se ova metoda zove *weighted sliding window*
- Ulaznu sliku možemo padovati nulama kako bi izlazna bila iste veličine

Konvolucija

- *Mean filter (low-pass filter)*

Original

$$* \quad \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$

Blur (with a mean filter)

- *Shift left*

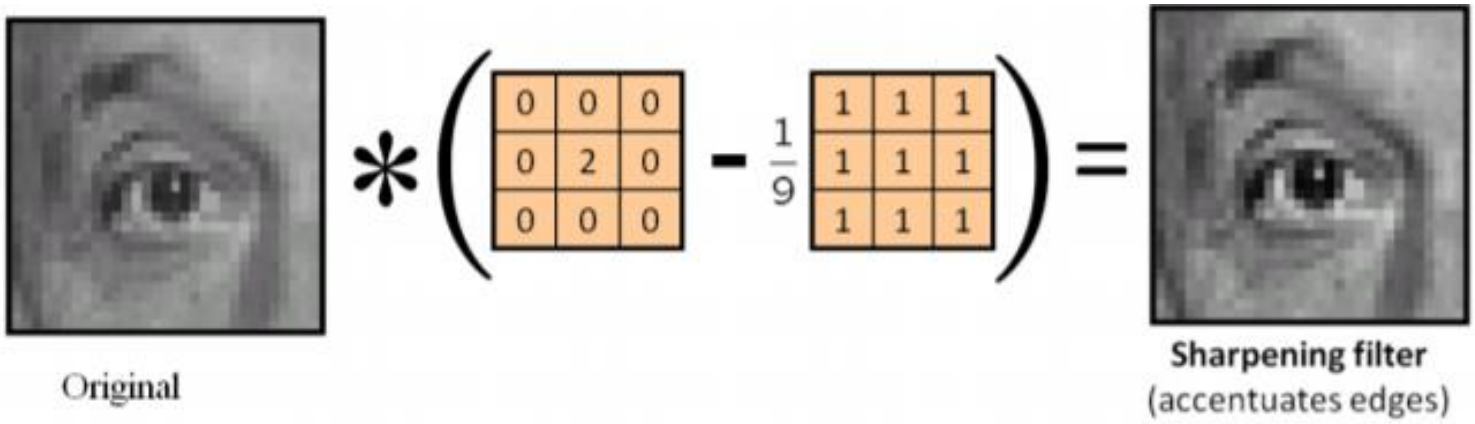
Original

$$* \quad \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$

Shifted left
By 1 pixel

Konvolucija

- *Sharpening (high-pass filter)*



Original

Sharpening filter
(accentuates edges)

https://en.wikipedia.org/wiki/Kernel_(image_processing)