SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

METODE ISPRAVLJANJA SLIKE

Kolegij: Obrada slike i računalni vid

Robert Pepić

Osijek, 2022.

Zadatak

Opisati teorijske osnove metoda: *contrast stretching*, *gamma correction*, *histogram equalization*. Napisati Python kod za implementaciju algoritma. Prikazati i pojasniti dobivene rezultate na proizvoljno odabranim slikama.

Uvod

U ovom radu razmotriti će se tri jednostavne metode ispravljanja slika. Opisat će se njihove teorijske osnove te objasniti princip rada njihovi algoritama. Metode će se implementirati korištenjem python programskog jezika te upotrebom *numpy* i *cv2* biblioteka. Prva od njih je *contrast stretching.* Ova metoda koristi se za ispravljanje kontrasta, a to postiže rastezanjem vrijednosti intenziteta slike na neki proizvoljan raspon. Druga metoda koja će se razmotriti je *gamma correction.* Ona se koristi za poboljšanje kvalitete osvjetljenja na slici. Zadnja metoda je *histogram equalization.* Ona se isto kao i prva metoda koristi za poboljšanje kontrasta na slici, no ima više statistički pristup rješavanju problema.

1. Contrast stretching

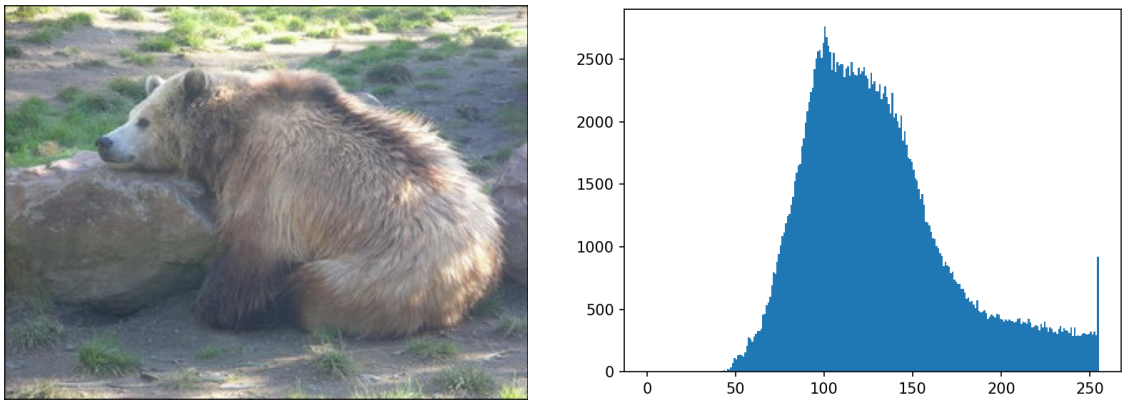
Contrast stretching ili normalizacija jednostavna je metoda ispravljanja slike koja poboljšava kontrast na slici „rastezanjem“ raspona vrijednosti intenziteta koje slika sadrži. Intenziteti koje slika sadrži rastežu se na neki proizvoljni raspon. Taj raspon bi se trebao odabrati na temelju histograma slike koji se želi ispraviti, na način da se smanji utjecaj graničnih vrijednosti. *Contrast stretching* primjenjuje se izravno na sliku, bez ikakvih transformacija, i to na svaki piksel pojedinačno. Ova metoda je linearna, što znači da je nova vrijednost intenziteta piksela linearno ovisna o vrijednosti intenziteta izvornog piksela. Kako je metoda linearna, moguće je napraviti inverz, odnosno ispravljenu sliku transformirati natrag u izvornu. Funkcija koja transformira piksel je slijedeća:

$$O = (I - c)\,\frac{b - a}{d - c} + a$$

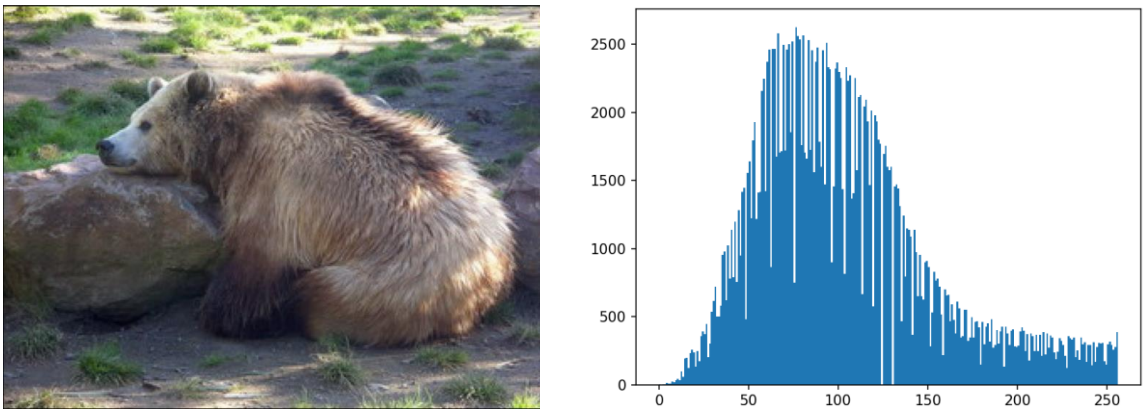
Gdje su:

- I – Stara vrijednost piksela
- O – Nova vrijednost piksela
- a – najmanja vrijednost intenziteta piksela izvorne slike
- b – najveća vrijednost intenziteta piksela izvorne slike
- c – najmanja vrijednost intenziteta piksela nove slike
- d – najveća vrijednost intenziteta piksela nove slike

Pomoću ove funkcije potrebno je transformirati svaki piksel slike. Ukoliko je u pitanju *grayscale* slika ovim procesom dobiti ćemo traženi rezultat. Ukoliko se pak radi o slici u boji, koja ima tri kanala (RGB) istu je prvo potrebno podijeliti po kanalima te na svakom kanalu zasebno izvršiti transformaciju. Pritom je potrebno koristiti isti raspon rastezanja kako bi se očuvali ispravni omjeri boja. Nakon transformacije kanale je potrebno ponovno spojiti i kao rezultat dobit će se ispravljena slika. Korištenjem *numpy* i *cv2* biblioteka u pythonu se lako može implementirati cijeli ovaj postupak. Izvorni kod implementacije nalazi se u dokumentaciji, a na slikama ispod mogu se vidjeti rezultati ispravljanja.



Slika 1.1 Originalna slika i njen histogram



Slika 1.2 Ispravljena slika i njen histogram

Kao što se može i vidjeti iz rezultata slika izgleda puno bolje nakon transformacije, a na histogramu se može primijetiti kako su se vrijednosti intenziteta „restegnule“ duž raspona od 0 do 255.

2. Gamma correction

Ljudsko oko percipira boju i osvjetljenje drugačije od senzora digitalnih kamera. Kada dvostruko veći broj fotona pogodi senzor digitalne kamere, on prima dvostruko veći signal (linearni odnos). Međutim, ljudske oči ne funkcioniraju tako. Umjesto toga, dvostruku količinu svjetlosti doživljavamo samo malo svjetlije (nelinearan odnos). Nadalje, naše oči su također puno osjetljivije na promjene tamnih tonova od svjetlijih tonova (također nelinearni odnos). *Gamma correction* metoda je ispravljanja slike koja se koristi kako bi se nadoknadila ova razlika između načina na koji ljudsko oko percipira boju i osvjetljenje i načina na koji to radi

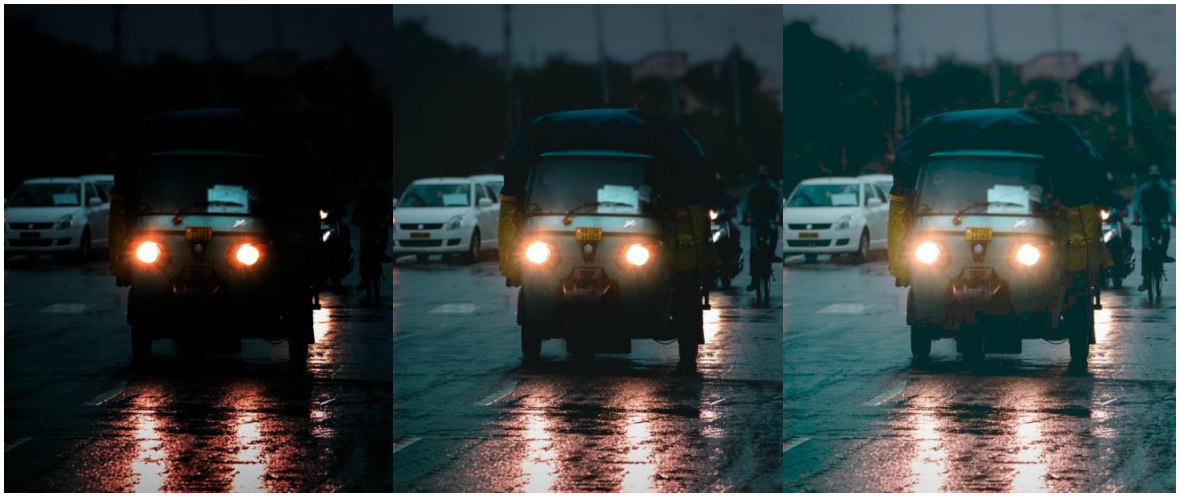
digitalna kamera. Metoda radi na slijedeći način. Prvo, intenzitet svakog piksela slike skalira se sa raspona [0, 255] na raspon [0, 1]. Nakon toga dobivena vrijednost potencira se na vrijednost 1/γ te se rezultat skalira natrag na raspon [0, 255]. Ovaj postupak može se opisati jednadžbom:

$$O = \left(\frac{I}{255}\right)^{\frac{1}{\gamma}} * 255$$

Gdje su:

- I – izvorna vrijednost piksela
- O – vrijednost piksela nakon ispravljanja slike
- γ – koeficijent gama koja kontrolira svjetlinu slike

Za vrijednosti γ < 1 slika će biti tamnija, dok će za vrijednosti γ > 1 slika biti svjetlija. Postavljanje vrijednosti γ na 1 nema učinka na sliku. Kao i prošla metoda ispravljanja, *Gamma Correction* implementirat će se pomoću *numpy* i *cv2* python biblioteka. Izvorni kod se kao i u prošlom primjeru nalazi u dokumentaciji. Na slikama ispod mogu se vidjeti rezultati implementiranog algoritma za vrijednosti γ = 0.5, γ = 1 i γ = 3.



Slika 2.1 Rezultati primijene algoritma na slike za vrijednosti γ = 0.5, γ = 1 i γ = 3.

Kao što se može vidjeti iz rezultata, slika na kojoj je primijenjen algoritam s parametrom γ = 0.5 postala je mračnija i neki detalji više nisu uočljivi, dok je slika na kojoj je primijenjen algoritam s parametrom γ = 3 postala svjetlija i detalji kao što su kuće i stupovi u pozadini postali su lakše uočljivi. Slika ispravljena s parametrom γ = 1 ekvivalentna je originalnoj.

3. Histogram equalization

Histogram Equalization je statistički pristup rješavanju problema ravnomjernog raspodjeljivanja vrijednosti intenziteta. Koristi se za poboljšanje kontrasta slike. Metoda se sastoji od nekoliko koraka. Prvo, za sliku koju želimo obraditi potrebno je izračunati vjerojatnost za svaku moguću razinu intenziteta piksela. Vjerojatnost pojedine razine dobije se tako da se ukupan broj piksela s tom razinom podjeli s ukupnim brojem piksela na slici:

$$p(x = i) = \frac{n_i}{n}, \;\; 0 \leq i < L$$

Gdje je L ukupan broj mogućih razina intenziteta. Za običnu *grayscale* sliku to bi bilo 256 razina. Nakon što se izračunaju vjerojatnosti za sve razine intenziteta, računa se kumulativna suma vjerojatnosti:

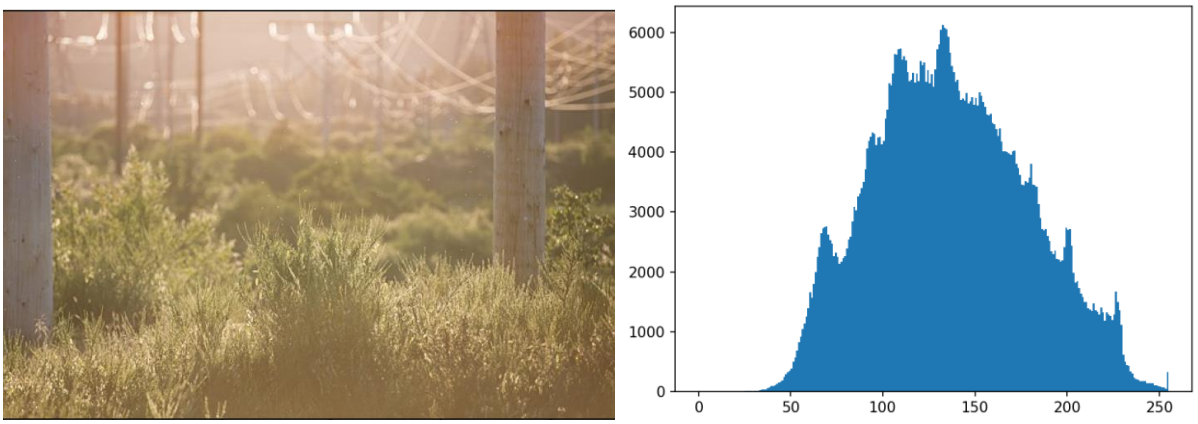
$$cdf(i) = \sum_{j=0}^{i} p(x = j)$$

Zatim se na temelju kumulativne sume vjerojatnosti stare vrijednosti intenziteta transformiraju u nove. Transformacija se vrši tako da se uzme vrijednost kumulativne sume vjerojatnosti pojedine razine te da se ona pomnoži s najvećim intenzitetom kojeg slika podržava (L-1, za *grayscale* sliku to je 255). Konačno, dobivene brojeve potrebno je zaokružiti na cijele. Kao rezultat dobiju se nove vrijednosti intenziteta razina. Na tablici ispod prikazane je postupak za *grayscale* sliku od n = 400 piksela.

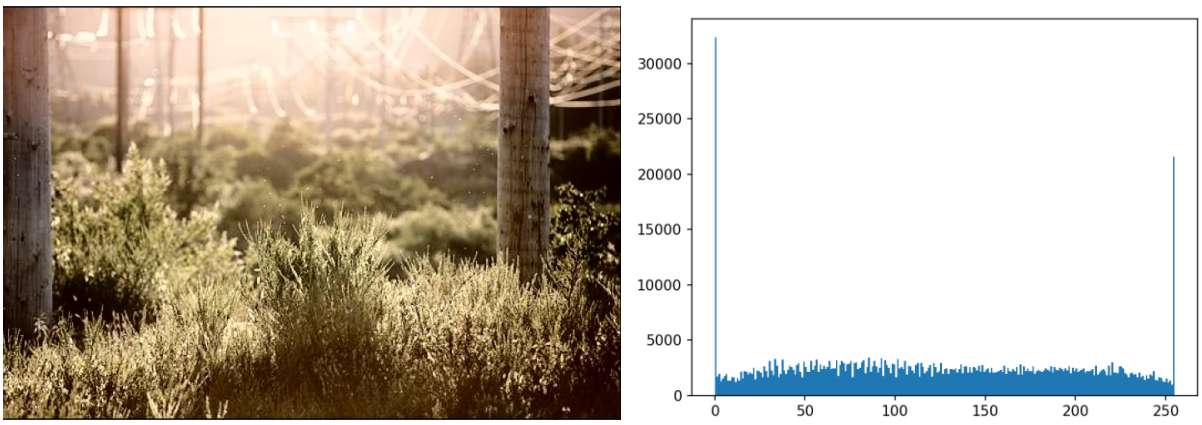
i (razina)	**n_i**	**p(x = i)**	**cdf(i)**	**cdf(i) * 255**	**Rezultat**
0	10	0.025	0.025	6.375	6
1	12	0.03	0.055	14.025	14
2	20	0.05	0.105	26,775	27
…	…	…	…	…	…
253	10	0.025	0,9625	245.4375	245
254	20	0.05	0.9875	251.8125	252
255	5	0.0125	1	255	255

Tablica 3.1 Koraci histogram equalization algoritma

Cijeli ovaj postupak lagano se implementira u pythonu korištenjem *cv2* biblioteke. Ona u sebi već ima definiranu funkciju *equalizeHist( )* kojoj se kao parametar jednostavno preda slika koju se želi ispraviti. Međutim, ona će ispravno raditi samo na *grayscale* slikama. Ukoliko se želi ispraviti sliku u boji (RGB) ne može se, kao kod *contrast stretching* metode samo podijeliti sliku na tri kanala i na svakom kanalu primijeniti metodu. Omjeri boja će se narušiti i neće se dobiti ciljani rezultat. Kako bi se riješio ovaj problem sliku je prvo potrebno pretvoriti u neki format koji ima odvojeni kanal za osvijetljenost, primjerice YCbCr. Zatim se na kanal koji predstavlja osvijetljenost (u ovom slučaju Y) primjeni *histogram equalization* metoda. Nakon toga sliku je potrebno pretvoriti nazad u izvorni oblik i kao rezultat dobije se slika s ispravljenim kontrastom. Na slikama ispod mogu se vidjeti rezultati primjene algoritma.



Slika 3.1 Originalna slika i njen histogram



Slika 3.2 Ispravljena slika i njen histogram

Može se vidjeti kako ispravljena slika ima puno bolji kontrast u odnosu na originalnu sliku. Učinak algoritma na sliku može se također vidjeti i u razlici između histograma originalne i ispravljene slike. Na histogramu ispravljene slike vrijednosti intenziteta su se ravnomjerno raspodijelile, odnosno ujednačile su se.

Zaključak

Ovim radom pojašnjene su tri metode ispravljanja slike. Objašnjene su njihove teorijske osnove i principi rada njihovih algoritama. Isti su implementirani korištenjem python programskog jezika te *numpy* i *cv2* biblioteka. Njihov rad demonstriran je na proizvoljno odabranim slikama te su prokomentirani dobiveni rezultati. Pokazano je kako i najjednostavnije metode ispravljanja mogu imati velik učinak na kvalitetu slike.

LITERATURA

Contrast Stretching

https://homepages.inf.ed.ac.uk/rbf/HIPR2/stretch.htm

https://pythontic.com/image-processing/pillow/contrast%20stretching

Gamma Correction

https://www.pyimagesearch.com/2015/10/05/opencv-gamma-correction/

https://lindevs.com/apply-gamma-correction-to-an-image-using-opencv/

Histogram equalization

https://en.wikipedia.org/wiki/Histogram_equalization

https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html