SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Hrvoje Ditrih

PhotoGrade – aplikacija za obradu digitalne fotografije

Zagreb, 2020.

Ovaj rad izrađen je na Zavodu za radiokomunikacije pod mentorstvom prof. dr. sc. Sonje Grgić i predan je na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2019./2020.

Sadržaj

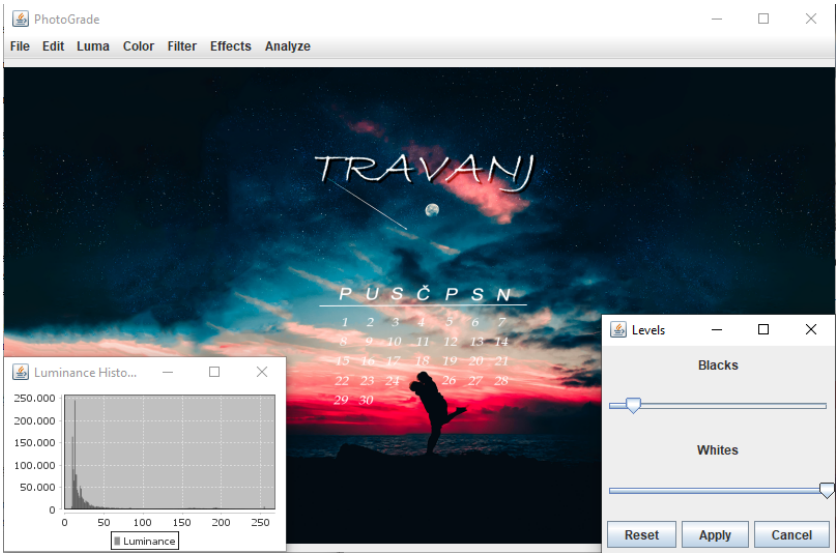
1. Uvod

Još od samih početaka ljudske civilizacije zapaženo je da su ljudi bilježili stvarnost u obliku vizualnih tvorevina - crteža. S napretkom civilizacije i umjetnosti razvijale su se nove tehnike, trendovi i pravci u vizualnom izričaju. Slikarstvo i kiparstvo pored svog umjetničkog značaja služili su kao način bilježenja ljudske stvarnosti, odnosno povijesti. Ljudska naklonost vizualnom izričaju može se povezati s načinom na koji ljudi doživljavaju svijet kroz osjetila. Naime, kroz osjetilo vida čovjek prima mnoštvo informacija iz okoline na koje se neprekinuto oslanja. Izumom fotografije ljudi su dobili alat pomoću kojeg mogu precizno zabilježiti određeni vremenski trenutak. Fotografija je postala nezamjenjiv način evidentiranja povijesti, novi umjetnički pravac te oblik bilježenja vizualnih informacija. Pored korištenja u tehničke, medicinske i poslovne svrhe, fotografijom se služi skoro svaka osoba u modernoj ljudskoj civilizaciji za evidentiranje svoje svakidašnjice pomoću kamera na pametnim telefonima. Pored prosječnog korisnika, profesionalni fotografi specijaliziraju se u korištenju digitalnih fotoaparata, umjetnosti fotografije i korištenju alata za manipulaciju digitalnim fotografijama. Fotografije dobivene iz fotoaparata nisu savršene iz mnogo razloga i postoji potreba za dodatnom obradom fotografije nakon što je ona snimljena. Jedan primjer bio bi povećanje svjetline na fotografiji koja je snimljena pri niskoj svjetlosti (slika 1.1). S druge strane, postoji potreba i za obradom fotografije koje su zadovoljavajuće kvalitete. Takve obrade obično su svrstane pod umjetnički cilj. Primjerice, obrada fotografije koja je dobivena modernim digitalnim fotoaparatom tako da izgleda kao staromodna fotografija dobivena iz starijih fotoaparata (slika 1.2).

Cilj ovog rada jest razmotriti metode obrade digitalne slike te ponuditi alat pomoću kojeg korisnik može obrađivati fotografije po želji. Sukladno cilju, razvijen je alat pod nazivom 'PhotoGrade' (slika 1.3) čija će se funkcionalnost analizirati te usporedno tome objasniti i razmotriti odabrane metode obrade digitalne fotografije.

Slika 1.1: Primjer povećanja svjetline slike

Slika 1.2: Primjer obrađivanja fotografije sepia efektom



Slika 1.3: Primjer korištenja alata PhotoGrade

2. Prostori boja digitalno snimljenih fotografija

U ovom poglavlju dan je općeniti uvid u digitalne slike. Koji su gradivni elementi digitalne slike, kakva je njihova struktura, kako se digitalne slike spremaju na računalu. Objašnjena su dva modela boja koji su korišteni u alatu PhotoGrade te u procesima obrade slika u daljnjim poglavljima.

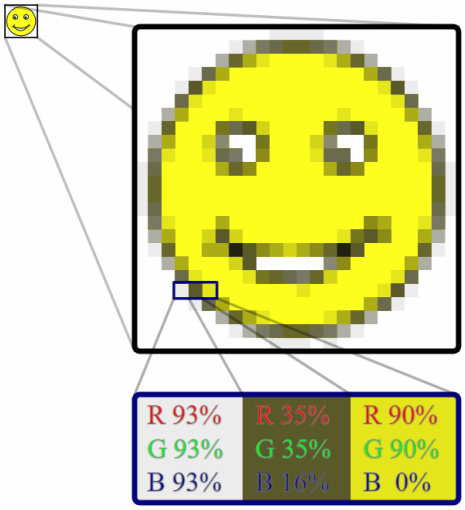
Općenito, digitalna slika je slikovna informacija pohranjena u digitalnom obliku. U širem smislu, digitalna slika može biti vektorska ili rasterska (slika 2.1). Vektorska slika nema definiranu rezoluciju, već su grafičke informacije pohranjene u obliku matematičkih funkcija i operacija koje se mogu iscrtati na bilo kojoj slikovnoj rezoluciji. Digitalne fotografije nikada nisu vektorske stoga ovaj tip digitalne slike više neće biti razmatran u nastavku. Suprotno vektorskoj slici, rasterska slika ima točno definiranu rezoluciju širine w i visine h koja se općenito zapisuje kao $w \times h$. Osnovni element rasterske slike naziva se piksel. Rasterska slika reprezentira se kao dvodimenzionalni niz dimenzija $w \times h$ u kojemu su pohranjene vrijednosti piksela. Svaki piksel u slici predstavlja jedan ton boje.

2.1. RGB model boja

Općenito, model boja je apstraktni matematički model koji opisuje kako se boje mogu reprezentirati kao n-torke [1]. RGB model boja je aditivni model[1] u kojemu se crvena, zelena i plava svjetlost kombiniraju zajedno kako bi se dobila široka paleta boja [2].

Boja u RGB modelu predstavlja se kao trojka (R, G, B) gdje su R,G i B vrijednosti crvene, zelene i plave respektivno. Svaka komponenta trojke ima vrijednost između 0 i maksimalne vrijednosti. Maksimalna vrijednost se može razlikovati

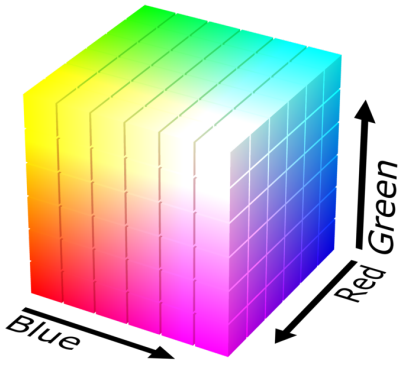
[1] aditivni model boja - konačna boja dobiva se kao suma njenih komponenti



Slika 2.1: Primjer rasterske slike i rastav piksela na RGB komponente

ovisno kakav raspon komponenata se odabere. Najčešće se uzima raspon $[0, 255]$ cijelih brojeva ili ponekad raspon $[0, 1]$ realnih brojeva. Potpuna odsutnost komponente se predstavlja vrijednošću 0, dok se potpuni intenzitet komponente predstavlja maksimalnom vrijednošću. Dakle, trojka $(0, 0, 0)$ predstavlja potpuno crni ton, dok $(255, 255, 255)$ predstavlja potpuno bijeli ton. Crvena boja se predstavlja s $(255, 0, 0, 0)$, zelena s $(0, 255, 0)$ te plava s $(0, 0, 255)$. Ostale boje i nijanse dobivaju se različitim kombinacijama R, G i B komponenata.

RGB model nije apsolutan model boja budući da iste vrijednosti boje na različitim uređajima mogu izgledati drugačije. RGB model se geometrijski može reprezentirati (slika 2.2) u trodimenzionalnom kartezijevom koordinatnom sustavu kao kocka u rasponu $[0, 1]$ po svim osima.



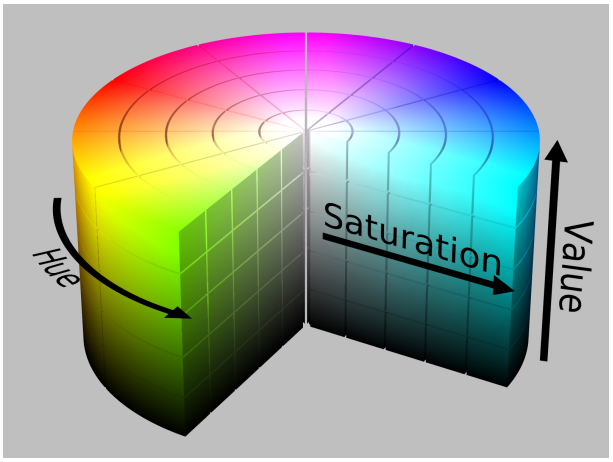
Slika 2.2: Geometrijska reprezentacija RGB modela

2.2. HSV model boja

HSV model je alternativna reprezentacija RGB modela koji je bolje usklađen s načinom na koji ljudi percipiraju boje [2]. U nazivu HSV slovo H predstavlja nijansu boje (engl. *hue*), slovo S predstavlja saturaciju (engl. *saturation*) te slovo V predstavlja vrijednost (engl. *value*). Za razliku od RGB modela koji je predstavljen u kartezijevom koordinatnom sustavu, HSV model predstavlja se u cilindričnom koordinatnom sustavu (slika 2.3). Boja u HSV modelu predstavljena je trojkom (H, S, V) gdje H predstavlja kut zakretanja u cilindričnom sustavu i označava nijansu boje, S predstavlja udaljenost točke od Z osi i označava saturaciju boje te V predstavlja Z koordinatu i označava svjetlinu boje.

Crvena boja počinje kutom 0°, zatim se nastavlja do zelene boje pod kutom 120° i plave boje pod kutom 240°, sve dok se ne vrati opet do crvene boje pod kutom 360°. Saturacija S predstavlja udaljenost boje od svoje luminantne komponente. Dakle, saturacija 0 predstavlja nijansu potpuno sivog tona, a saturacija maksimalne vrijednosti predstavlja potpunu zasićenost boje. Svjetlina V iznosa 0 predstavlja potpuno crnu boju, dok svjetlina maksimalnog iznosa predstavlja najsvjetliju verziju određene nijanse boje i saturacije. Ako je saturacija boje jednaka 0 i svjetlina je maksimalnog iznosa, tada će rezultirana boja u HSV modelu biti potpuno bijela.

Izravna pretvorba RGB modela u HSV model rezultira time da HSV model više neće biti cilindričan, već će biti piramida s heksagonalnom bazom. U ovom radu, za neke metode obrade slike je bilo potrebno napraviti pretvorbu slike iz RGB modela u HSV model te je navedeno kada se radi s HSV modelom. Ako ništa nije navedeno, pretpostavlja se korištenje RGB modela.



Slika 2.3: Geometrijska reprezentacija HSV modela

Pretvorba iz RGB u HSV model radi se prema sljedećim izrazima [3]:

$$V = \max(R, G, B) \tag{2.1}$$

$$S = \begin{cases} \frac{V - \min(R,G,B)}{V} & \text{ako } V \neq 0 \\ 0 & \text{inače} \end{cases} \tag{2.2}$$

$$H = \begin{cases} \frac{60(G-B)}{V-\min(R,G,B)} & \text{ako } V = R \\ \frac{120+60(B-R)}{V-\min(R,G,B)} & \text{ako } V = G \\ \frac{240+60(R-G)}{V-\min(R,G,B)} & \text{ako } V = B \end{cases} \tag{2.3}$$

Vrijednosti se potom skaliraju na raspon $[0, 255]$:

$$V = 255 \cdot V \tag{2.4}$$

$$S = 255 \cdot S \tag{2.5}$$

$$H = \frac{H}{2} \tag{2.6}$$

2.3. Pohrana digitalne slike u radnoj memoriji

Kao što je već navedeno, digitalna slika se u računalu pohranjuje kao dvodimenzionalni niz u kojemu svaki element predstavlja jedan piksel. No, to je samo slučaj kada se apstrahirano govori o digitalnoj slici. Budući da su pikseli predstavljeni nekim modelom boje, najčešće RGB modelu, koji sadrže više komponenti, stvarna pohrana u memorije je češće trodimenzionalni niz u kojemu su vrijednosti $v_{i,j,c}$ gdje je i pozicija po širini, j pozicija po visini te c indeks komponente iz modela boje. Ako se navedeno trodimenzionalno polje rastavi na dvodimenzionalna polja po dubini, takva dvodimenzionalna polja se nazivaju kanali slike. Digitalna slika dimenzija 100×200 predstavljena RGB modelom sadrži tri kanala boje: crveni, zeleni i plavi, gdje svaki kanal ima dimenzije 100×200.

Kako je memorija u računalu ograničena, a vrijednosti se spremaju kao n-bitni binarni brojevi, preciznost pohranjenih vrijednosti nije beskonačna. Tako će broj različitih boja koje digitalna slika može prikazati biti ograničen s duljinom n zapisa binarnog broja u memoriji računala. Ako se za svaku komponentu modela uzima duljina od n bita te se koristi model boja koji ima k komponenti, svaki piksel digitalne slike će zauzimati $n \cdot k$ bita po pikselu.

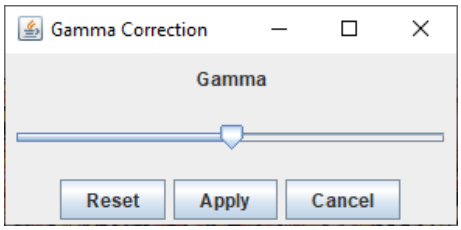
U praksi se često za pohranjivanje digitalnih slika uzima veličina $n = 8$ bita (jedan bajt) po komponenti. Ako se radi o RGB modelu i svaka komponenta zauzima 8 bita, jedan piksel će biti predstavljen s $8 \cdot 3 = 24$ bita po pikselu. Takva slika ima tri kanala boje te svaki piksel može prikazati $2^{8^3} = 256^3 = 16777216$ različitih boja.

Za pohranjivanje crno-bijelih slika (engl. *grayscale*) se koristi samo jedan kanal koji predstavlja svjetlinu sivog tona, gdje je 0 potpuno crna boja, a maksimalna vrijednost potpuno bijela boja. Ako se koristi 8 bita po pikselu, tada svaki piksel može prikazati $2^8 = 256$ različitih nijansi sive.

U praksi se često koristi i RGBA model boja koji koristi još dodatni alfa kanal za određivanje prozirnosti boje. Za prikazivanje visokokvalitetnih fotografija i digitalnih slika koristi se i 16, 24 ili 32 bita po komponenti. U ovome radu slike su pohranjivane kao crno-bijele 8 bitne slike ako je slika sadržavala samo sive tonove odnosno kao 8 bitne RGB slike ako je slika bila u boji.

3. Metode za upravljanje vizualnim značajkama digitalne slike

U ovom poglavlju navedeni su alati za obradu slike u PhotoGrade-u i usporedno s njima objašnjene su same metode obrade koje alati koriste. Kada se odabere neki alat, u većini slučajeva se otvara dodatni prozor (slika 3.1) u kojemu se dodatno namještaju parametri alata. Svaki takav prozor na dnu sadrži gumbove: *Reset*, *Apply* i *Cancel*. Gumb *Reset* vraća parametre alata na njihove inicijalne vrijednosti, gumb *Apply* primjenjuje trajno zadani alat na trenutnu sliku (može se poništiti pomoću *Undo*) te gumb *Cancel* kojim se odustaje od zadane promjene alatom.



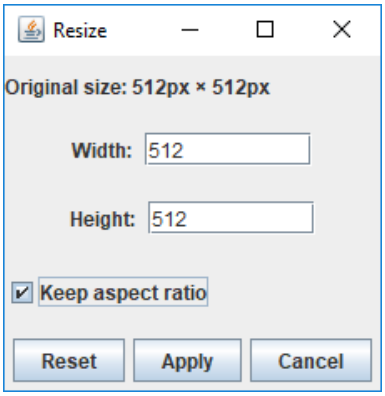
Slika 3.1: Prozor za uglađivanje gamma korekcije

3.1. 2D transformacije

Prvo su navedeni alati koji vrše 2D transformacije nad slikama. U PhotoGrade-u ti alati se nalaze u izborniku 'Edit'.

3.1.1. Promjena dimenzija

Ovom alatu se pristupa u izborniku Edit → Resize. Otvara se prozor (slika 3.2) u kojemu se mogu zadati nova širina i visina slike. Također, postoji opcija *Keep aspect ratio* koja je inicijalno uključena te koja osigurava korisniku da će nakon promjene dimenzija slike zadržati svoj izvorni omjer širine i visine što je u većini slučajeva poželjno.



Slika 3.2: Alat za promjenu dimenzija slike

U nastavku je priložen izvorni kod funkcije (izvorni kod 3.1) koja mijenja dimenzije slike. Promjena dimenzije vrši se OpenCV funkcijom *resize*. Za smanjivanje i povećanje slike koriste se različite interpolacije koje su implementirane u OpenCV biblioteci te neće biti ovdje objašnjene.

```
1  public static Mat resize(Mat mat, int width, int height) {
2          int originalSize = mat.rows() * mat.cols();
3          int newSize = width * height;
4
5          int interpolation = newSize < originalSize ? Imgproc.
               ↪ INTER_AREA : Imgproc.INTER_CUBIC;
6          Mat rmat = new Mat(height, width, mat.type());
7          Imgproc.resize(mat, rmat, rmat.size(), 0, 0, interpolation);
8
9          return rmat;
10   }
```

Izvorni kod 3.1: Promjena dimenzija slike

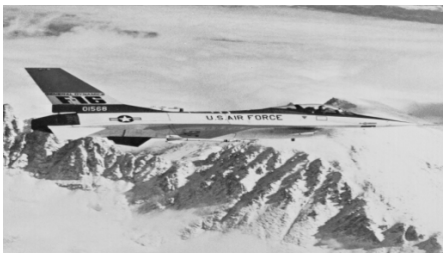
Na slikama 3.3 i 3.4 dan je primjer promjene dimenzija slike. Slika 3.3 pod a) ima dimenzije 512×512 piksela nakon čega joj promijenimo dimenzije na 400×200 piksela. Slika 3.4 pod a) ima omjer širine i visine 1:1 te nakon promjene dimenzija ima omjer 16:9.

(a) Prije

(b) Poslije

Slika 3.3: Primjer promjene dimenzija slike

(a) Prije



(b) Poslije

Slika 3.4: Primjer promjene dimenzija slike

3.1.2. Odsijecanje

Ovom alatu se pristupa u izborniku Edit → Crop. Otvara se prozor u kojem korisnik može odabrati koliko želi odrezati sliku s gornje, donje, lijeve i desne strane slike.

(a) Prije **(b)** Poslije

Slika 3.5: Primjer odsijecanja slike

Ovo je postignuto tako da se uzme podmatrica koja je određena parametrima *xs* i *xe* koji određuju početak i kraj slike po horizontalnoj osi te parametrima *ys* i *ye* koji određuju početak i kraj slike po vertikalnoj osi. U nastavku je priložen kod koji poziva OpenCV funkciju *submat* koja vrši gore navedeni postupak

```
1 public static Mat cropImage(Mat mat, int xs, int ys, int xe, int ye)
2 {
3         Mat rez = mat.submat(ys, ye, xs, xe);
4         return rez;
5  }
```

Izvorni kod 3.2: Odsijecanje slike

3.1.3. Zrcaljenje

Ovom alatu se pristupa u izborniku Edit → Flip pa zatim odabir horizontalnog ili vertikalnog zrcaljena. Ovaj efekt se postiže jednostavnom zamjenom simetrično udaljenih piksela oko središnje horizontalne ili vertikalne osi slike.

(a) Izvorna slika

(b) Horizontalno zrcaljenje

(c) Vertikalno zrcaljenje

Slika 3.6: Primjer zrcaljenja slike

3.1.4. Zakretanje

Ovom alatu se pristupa u izborniku Edit → Rotate pa zatim odabir u smjeru kazaljke na satu ili u smjeru obrnutome od kazaljke na satu. Slika 3.7 prikazuje primjer zakretanja slike.

(a) Izvorna slika

(b) Rotiranje za 90° u smjeru kazaljke na satu

(c) Rotiranje za 90° u smjeru suprotnome od kazaljke na satu

Slika 3.7: Primjer rotiranja slike

Opisat će se općeniti postupak rotacije u 2D prostoru [4]. Budući da se rotacija općenito izvodi oko ishodišta koordinatnog sustava potrebno je osmisliti način rotacije oko proizvoljne točke. U ovom slučaju sliku želimo rotirati oko njezinog središta $S = (S_x, S_y)$. U nastavku ćemo pretpostaviti da se nalazimo u homogenom prostoru.

Prvi korak je translacija slike u ishodište koordinatnog sustava što ćemo učiniti sljedećom matricom:

$$\Psi_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -S_x & -S_y & 1 \end{bmatrix}$$

Zatim se slika rotira oko ishodišta koordinatnog sustava za kut θ sljedećom matricom:

$$\Psi_2 = \begin{bmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Naposljetku sliku treba vratiti u točku S gdje se nalazila prije rotacije i translacije što činimo inverzom matrice Ψ_1:

$$\Psi_3 = \Psi_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ S_x & S_y & 1 \end{bmatrix}$$

Ukupnu matricu transformacije Ψ dobivamo umnoškom sve tri matrice.

$$\Psi = \Psi_1 \cdot \Psi_2 \cdot \Psi_3$$

Treba primijetiti da rotirana slika nema iste dimenzije kao izvorna. Štoviše, slika koja je imala širinu w i visinu h, odnosno dimenzije $w \times h$, sada ima dimenzije $h \times w$.

Da bi se dobila konačna rotirana slika svaki piksel treba transformirati matricom Ψ. Uzima se da se pojedini piksel nalazi u točki $T = \begin{bmatrix} T_x & T_y & 1 \end{bmatrix}$, a zatim se za svaki piksel vrši transformacija $T' = T \cdot \Psi = \begin{bmatrix} T'_x & T'_y & 1 \end{bmatrix}$ nakon čega se ponovno natrag iz homogenog prostora točka T' vraća u radni prostor te se dobiva krajnja pozicija rotiranog piksela na poziciji $T_r = (T'_x, T'_y)$

3.2. Upravljanje luminancijom

3.2.1. LUT

Za upravljanje luminancijom korišten je LUT (*lookup table*) koji je zapravo tablica preslikavanja vrijednosti. U slučaju digitalne slike, vrijednosti koje se preslikavaju su vrijednosti pojedinih piksela. Uzima se da je v vrijednost pojedinog piksela za koji vrijedi $v \in [0, 255]$ budući da se radi o 8-bitnim slikama. LUT se definira kao preslikavanje:

$$v' = L(v) \tag{3.1}$$

Gdje je v' nova vrijednost piksela, a $L(v)$ bilo koja kontinuirana funkcija na intervalu $[0, 255]$. Naravno, ako vrijednost $L(v)$ izađe iz intervala $[0, 255]$ onda vrijedi:

$$v' = \begin{cases} 0 & \text{ako } L(v) < 0 \\ 255 & \text{ako } L(v) > 255 \end{cases} \tag{3.2}$$

Dodatno, ako $L(v)$ nije cijeli broj uzima se najveće cijelo dobivene vrijednosti:

$$v' = \lceil L(v) \rceil \tag{3.3}$$

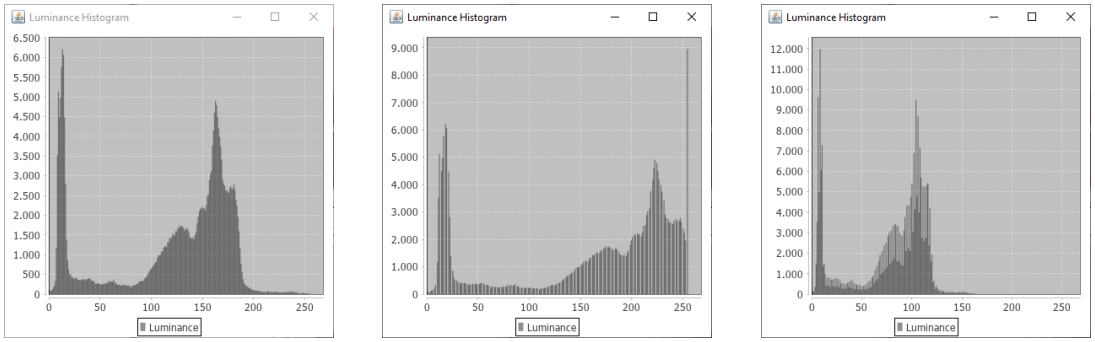
Naravno, navedeno preslikavanje izvodi se za svaki piksel u slici.

3.2.2. Linearno upravljanje svjetlinom i kontrastom

Pojmovi svjetline i kontrasta objasnit će se pod pretpostavkom da su pikseli 8-bitni te da su sivi tonovi (engl. *grayscale*). Što je vrijednost piksela veća, to je on svjetliji, odnosno piksel $v_1 = 200$ je svjetliji od piksela $v_2 = 100$. Pojam kontrasta se može definirati kao razlika svjetlina dvaju piksela. Što je razlika veća, to je kontrast veći. Dakle, kontrast se definira kao $c = \mid v_1 - v_2 \mid$. Sada su neprecizno definirani pojmovi svjetlije i kontrastnije slike budući da nije potrebna gruba definicija, već je dovoljna intuicija. Slika je svjetlija što ima veći broj svjetlijih piksela, a kontrastnija je što ima veću razliku između sjena i svjetlijih tonova.

Alatu za linearno upravljanje svjetlinom i kontrastom pristupa se izbornikom Luma → Linear Brightness & Contrast nakon čega će se pojaviti prozor na kojemu se nalaze dva klizača (engl. *sliders*) pomoću kojih se namještaju parametri linearnog upravljanja svjetline i kontrasta slike.

(a) Izvorna slika **(b)** $\alpha > 1$ **(c)** $\alpha < 1$



(d) Histogram od a) **(e)** Histogram od b) **(f)** Histogram od c)

Slika 3.8: Manipulacija kontrasta linearnim upravljanjem kontrastom

Linearno upravljanje svjetlinom i kontrastom ostvareno je kao LUT [5]:

$$L(\alpha, \beta, v) = \alpha v + \beta \tag{3.4}$$

gdje je α faktor povećanja kontrasta, a β pomak svjetline. Uvijek vrijedi $\alpha > 0$. Za $\alpha > 1$ kontrast se povećava, a za $\alpha < 1$ kontrast se smanjuje. Analizira se slika 3.8 kako bi se na primjeru razumjelo što se događa sa slikom. Vidi se kako su vrijednosti na izvornoj slici centrirane i ravnomjerno raspoređene oko srednjih tonova. Kada povećavamo kontrast (vidi primjer b), vrijednosti se linearno udaljavaju od vrijednosti nula, suprotno tome kada smanjujemo kontrast (vidi primjer c), vrijednosti se linearno približavaju vrijednosti nula. Ovom metodom kontrast utječe i na svjetlinu slike. Subjektivnom procjenom ovo ne daje idealne rezultate koje bismo očekivali od povećanja kontrasta, stoga se u idućem odjeljku razmatra bolja metoda.

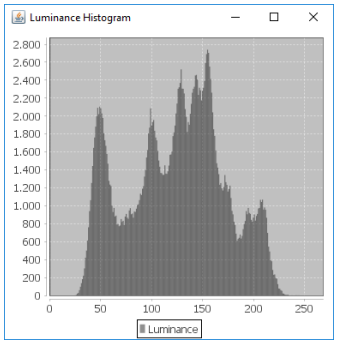
Kada je $\beta > 0$ onda se svjetlina povećava, kada je $\beta < 0$ svjetlina se smanjuje. Na slici 3.9 je analiziran utjecaj pomaka β. Koristi se ista slika kao u prošlom razmatranju. U slučaju kada povećavamo svjetlinu (vidi slučaj b), to se na histogramu odražava kao linearni pomak u desno, dok u slučaju kada smanjujemo

svjetlost (vidi slučaj c) vrijednosti se pomiču linearno u lijevo. Sa subjektivne strane rezultat nije loš, ali postoje slučajevi kada će rezultat izgledati vrlo neprirodno. Funkcija koja bi bila idealna bi bila ona koja bi najviše utjecala na srednje tonove, a manje na sjene i svijetle tonove. Takva metoda će se razmotriti u sljedećem odjeljku.

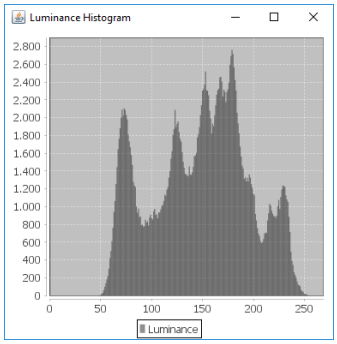
(a) Izvorna slika

(b) $\beta > 0$

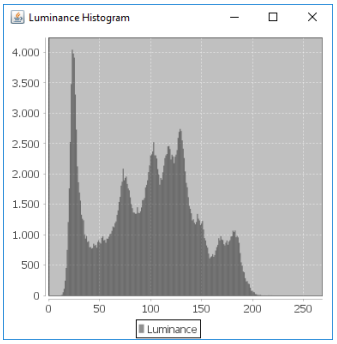
(c) $\beta < 0$



(d) Histogram od a)



(e) Histogram od b)



(f) Histogram od c)

Slika 3.9: Manipulacija svjetline linearnim upravljanjem svjetlinom

3.2.3. Upravljanje svjetlinom i kontrastom kubičnim splajn krivuljama

Splajn krivulje su glatke polinomne krivulje definirane po dijelovima [6]. Jedna od njih je kubična splajn krivulja. Takva krivulja ima dijelove definirane izrazom:

$$P_i(t) = a_i + b_i(t - x_i) + c_i(t - x_i)^2 + d_i(t - x_i)^3 \tag{3.5}$$

Za danih $n + 1$ koordinata $(x_0, y_0), ..., (x_n, y_n)$, pronalazi se n, P_0 i P_k, koji zadovoljavaju za $1 \leq k \leq n - 1$:

$$P_0(x_0) = y_0 \tag{3.6}$$

$$P_{k-1}(x_k) = y_k = P_k(x_k) \tag{3.7}$$

$$P'_{k-1}(x_k) = P'_k(x_k) \tag{3.8}$$

$$P''_{k-1}(x_k) = P''_k(x_k) \tag{3.9}$$

$$P''_0 = P''_{x_n} = 0 \tag{3.10}$$

Kubične splajn krivulje najčešće se nalaze kubičnim splajn interpolatorima. U sklopu projekta je korišten paket Math iz Apache Commons [1] koji sadrži kubični splajn interpolator. Navedene krivulje su korištene kao LUT-ovi kako bi se postigla bolja kontrola kod upravljanja svjetlinom i kontrastom.

Za upravljanje kontrastom korišten je LUT koji je ovisno o parametru α kubična splajn krivulja ili pravac koji prolazi kroz zadane točke. Za $\alpha > 0$, LUT jest kubična splajn krivulja koja redom prolazi kroz točke (x_i, y_i) za $i \in [1, 5]$:

$$(0, 0),\ (64 + \alpha, 64 - \alpha),\ (127, 127),\ (191 - \alpha, 191 + \alpha),\ (255, 255)$$

Odnosno za $\alpha < 0$, točke se interpoliraju linearnom interpolacijom, tj. LUT jest pravac koji prolazi kroz točke:

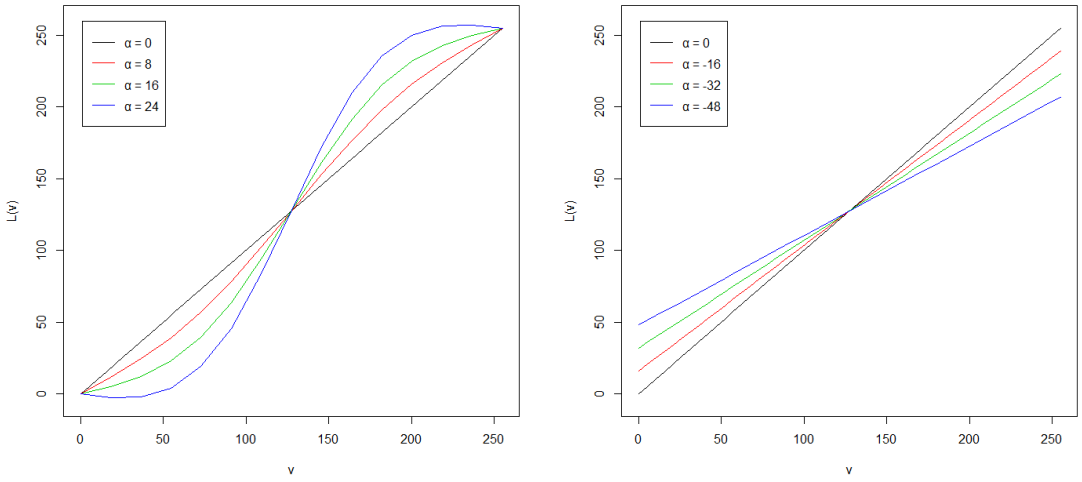
$$(0, 0 - \alpha),\ (255, 255 + \alpha)$$

Ovako definirani LUT upravljanja kontrastom uzima se kao $L_1(\alpha, v)$. Graf koji zornije prikazuje krivulje od $L1(\alpha, v)$ prikazan je na slici 3.10.

LUT koji upravlja svjetlinom jest kubična splajn krivulja koja ovisi o parametru β te koja redom prolazi kroz točke:

$$(0, 0),\ (127 - \beta, 127 + \beta),\ (255, 255)$$

[1] Skupina biblioteka za programski jezik Java

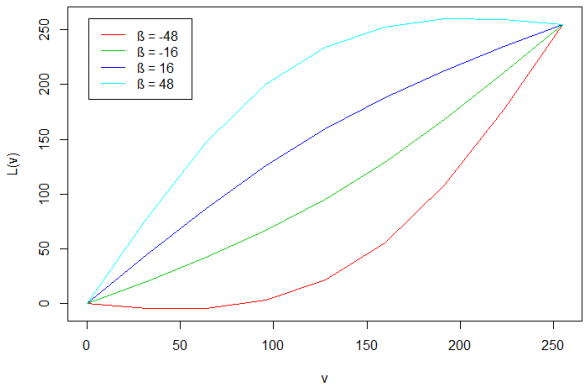


(a) $\alpha > 0$ **(b)** $\alpha < 0$

Slika 3.10: Grafovi koji prikazuju LUT upravljanja kontrastom

Tako definiran LUT uzima se kao $L_2(\beta, v)$ te je zornije prikazan na slici 3.11. Konačno, LUT koji upravlja svjetlinom i kontrastom kubičnim splajn krivuljama jest:

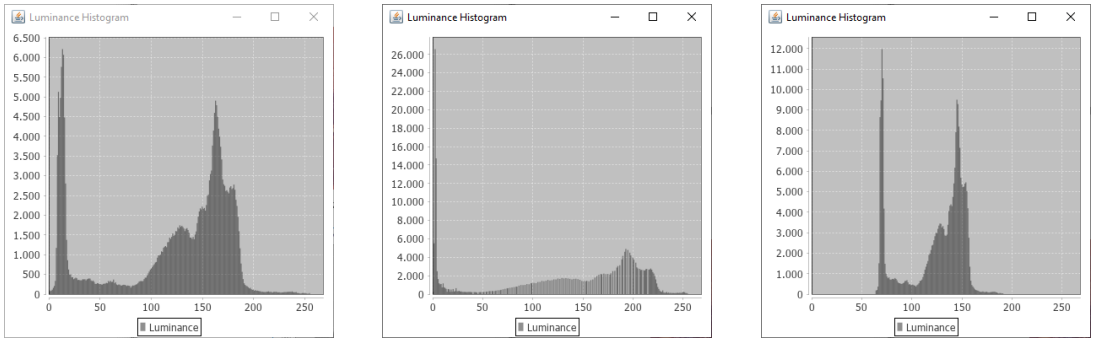
$$L(\alpha, \beta, v) = L_2(\beta, L_1(\alpha, v)) \tag{3.11}$$



Slika 3.11: Graf koji prikazuje LUT upravljanja svjetlinom

Alatu za navedeno upravljanje kontrastom i svjetlinom pristupa se izbornikom File → Smart Brightness & Contrast nakon čega se pojavi identičan prozor za namještanje parametara α i β kao kod linearnog upravljanja svjetlinom i kontrastom (slika 3.13). Na slici 3.12 vidi se da se povećanjem kontrasta (vidi primjer

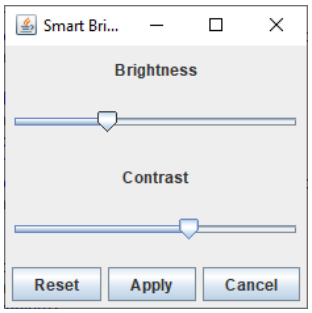
(a) Izvorna slika **(b)** $\alpha > 0$ **(c)** $\alpha < 0$



(d) Histogram od a) **(e)** Histogram od b) **(f)** Histogram od c)

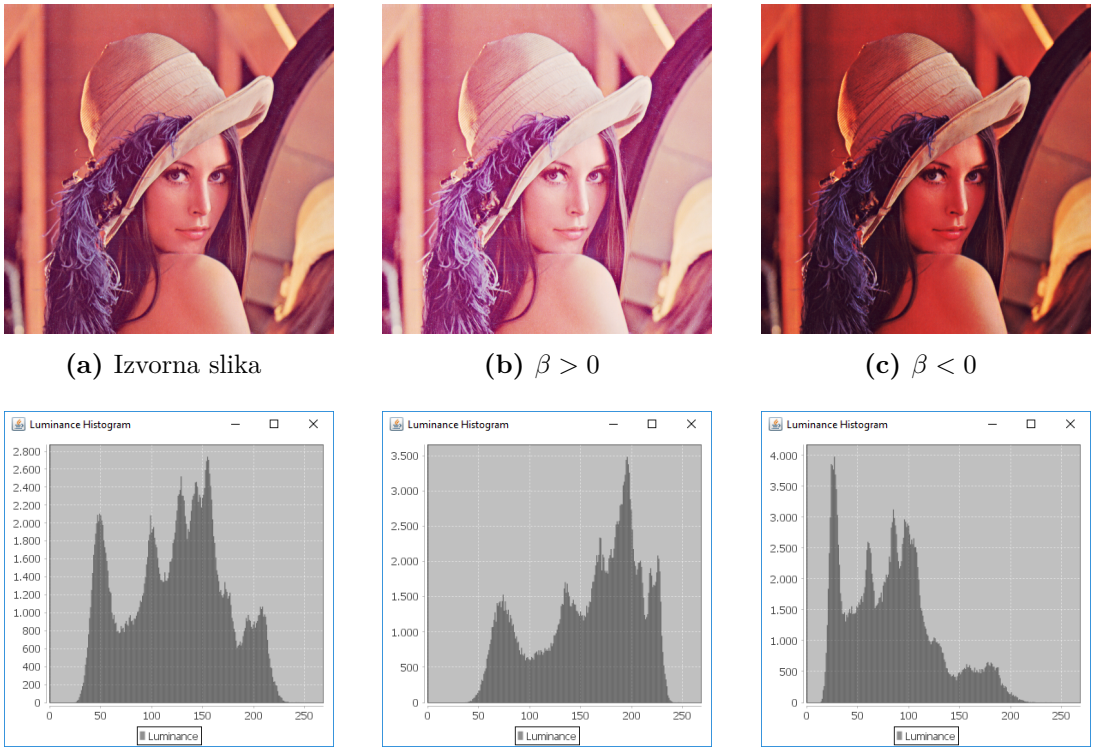
Slika 3.12: Upravljanje kontrastom LUT-om L_1

b) vrijednosti na histogramu razmiču od središta $x = 127$, nasuprot smanjivanju kontrasta (vidi primjer c) kada se vrijednosti na histogramu skupljaju oko središta. Ovime je postignuto prirodnije upravljanje kontrastom za razliku od linearne metode te dodatno, promjena kontrasta više ne utječe toliko na svjetlinu slike.



Slika 3.13: Korištenje alata Smart Brightness & Contrast

Na slici 3.14 analizira se povećanje i smanjenje svjetline navedenom metodom. Iz danih histograma vidi se da se vrijednosti sivih tonova primiču sjenama odnosno svijetlim tonovima ovisno o parametru β.



(a) Izvorna slika **(b)** $\beta > 0$ **(c)** $\beta < 0$

(d) Histogram od a) **(e)** Histogram od b) **(f)** Histogram od c)

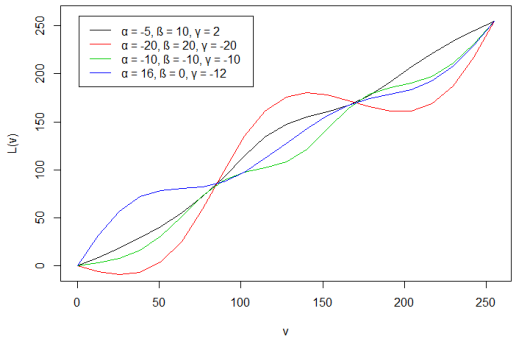
Slika 3.14: Upravljanje svjetlinom LUT-om L_2

3.2.4. Upravljanje sjenama, srednjim i svijetlim tonovima

Ovome alatu pristupa se izbornikom Luma → Shadows & Highlights nakon čega se otvara prozor s tri klizača (engl. *sliders*) pomoću kojih se upravlja sjenama, srednjim i svijetlim tonovima. Princip je sličan onome kod upravljanja svjetlinom pomoću kubičnih splajn krivulja. Naime, kod ove metode svaki parametar kontrolira svjetlinu jednog raspona luminancije. Budući da se radi o 8-bitnim slikama i vrijednostima u rasponu $[0, 255]$, taj raspon je podijeljen na tri jednaka dijela: sjene u rasponu $[0, 85)$, srednji tonovi u rasponu $[85, 170)$ te svijetli tonovi u rasponu od $[170, 255]$. Na temelju tih raspona definira se LUT koji je kubična splajn krivulja te ovisi tri parametra α, β i γ koji upravljaju sjenama, srednjim i svijetlim tonovima respektivno te redom prolazi kroz točke:

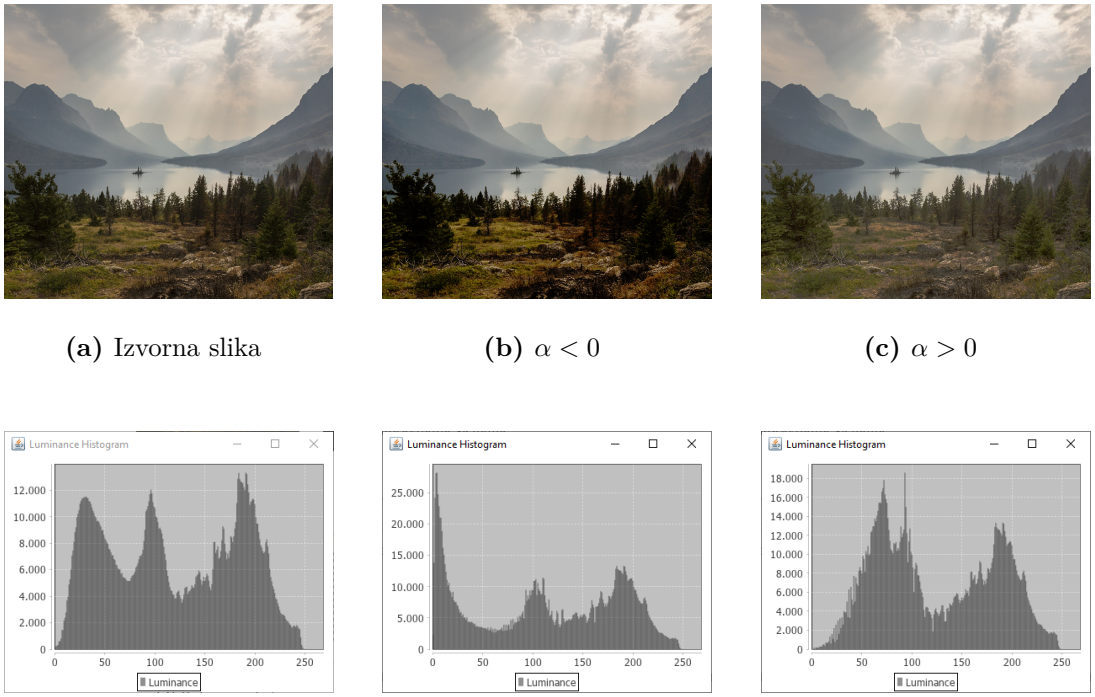
$$(0,0),\ (42-\alpha, 42+\alpha),\ (85{,}85),\ (127-\beta, 127+\beta),$$
$$(170,170),\ (212-\gamma{,}212+\gamma), (255,255)$$

Slika 3.15 zornije prikazuje kako LUT upravlja luminancijom. Na slici 3.16 je primjer upravljanja sjenama. Vidi se kako parametar α povećava i smanjuje



Slika 3.15: Krivulje LUT-a koje upravljaju sjenama, srednjim i svijetlim tonovima

svjetlinu samo dijelovima slike koji se na histogramu nalaze u donjoj trećini, u ovom slučaju to je većinski drveće na donjem dijelu slike.



(a) Izvorna slika **(b)** $\alpha < 0$ **(c)** $\alpha > 0$

(d) Histogram od a) **(e)** Histogram od b) **(f)** Histogram od c)

Slika 3.16: Primjeri upravljanja sjenama

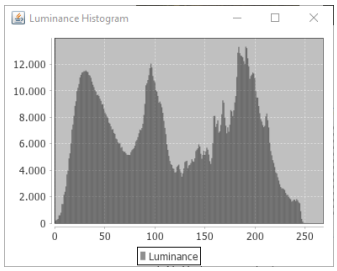
Analogno vrijedi i za primjer na slici 3.17 gdje se upravlja srednjim i svijetlim tonovima. Srednji tonovi na ovom primjeru pripadaju planinama na vertikalnoj sredini slike te je prikazano kako promjena parametra β utječe samo na njih. Promjena svijetlih tonova odnosno parametra γ utječe samo na nebo pri vrhu

slike.

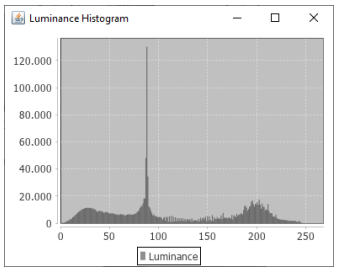
(a) Izvorna slika

(b) $\beta < 0$

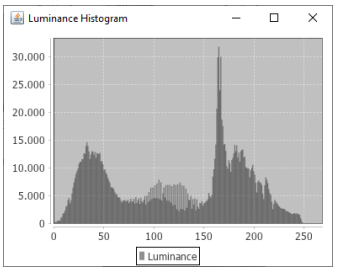
(c) $\beta > 0$



(d) Histogram od a)



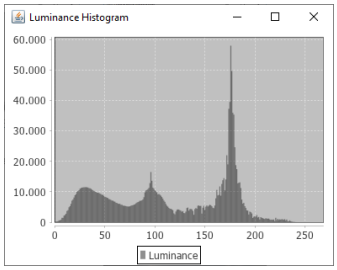
(e) Histogram od b)



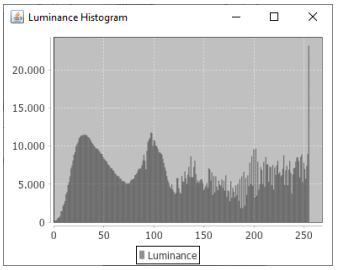
(f) Histogram od c)

(g) $\gamma < 0$

(h) $\gamma > 0$



(i) Histogram od g)



(j) Histogram od g)

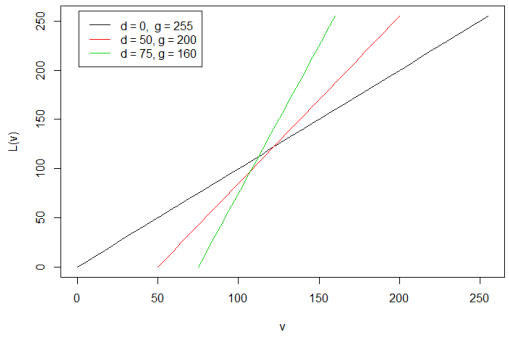
Slika 3.17: Primjeri upravljanja srednjim i svijetlim tonovima

3.2.5. Upravljanje razinama

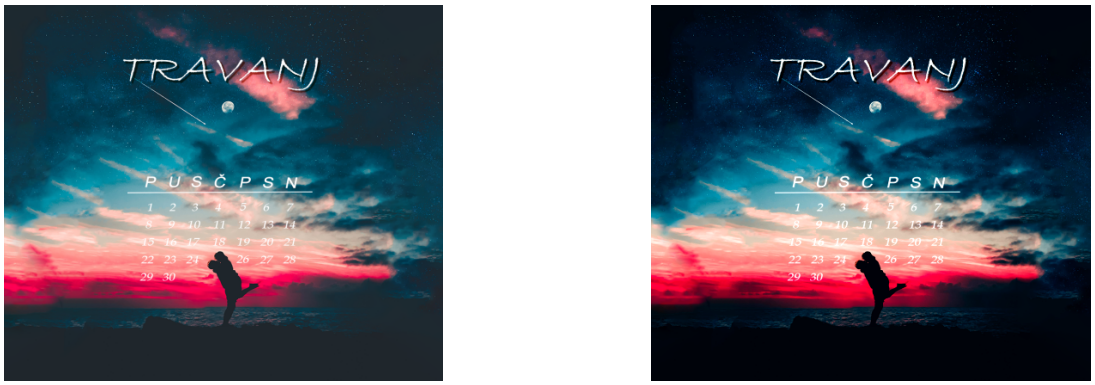
Razine se definiraju pomoću parametara d, g koji su u rasponu $[0, 255]$. Sve vrijednosti manje od g pretvore se u crne piksele te sve vrijednosti koje su veće od g pretvore se u bijele piksele. Uzet je pristup koji će sačuvati prirodnu strukturu slike. Definira se LUT (slika 3.18):

$$L(v) = \frac{255}{g-d}\, v - \frac{255}{g-d}\, d \tag{3.12}$$

Alatu za upravljanje razinama se pristupa izbornikom Luma → Levels. Primjer korištenja alata se nalazi na slici 3.19



Slika 3.18: Graf LUT-a koji upravlja razinama



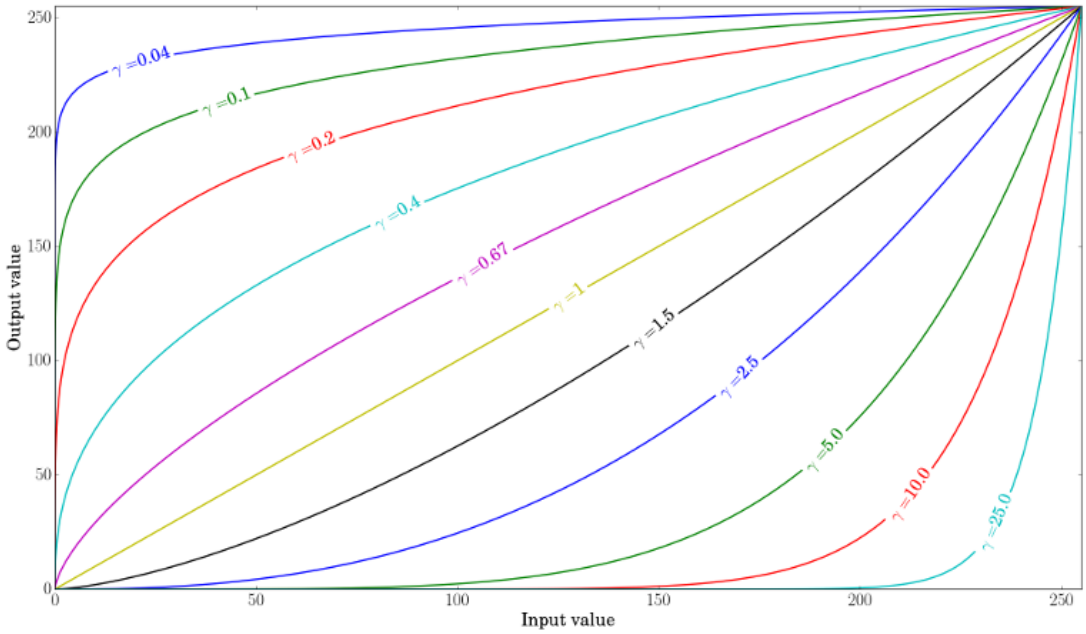
(a) Prije **(b)** Poslije

Slika 3.19: Primjer upravljanja razinama

3.2.6. Gamma korekcija

Gamma korekcija jest primjenjivanje nelinearnog LUT-a (slika 3.20) na vrijednosti čime se postiže nelinearno povećanje ili smanjenje svjetline što je u nekim slučajevima prirodnija transformacija za razliku od linearne [5]. Ovome alatu se pristupa izbornikom Luma → Gamma Correction. Gamma korekcija definira se kao LUT koji ovisi o parametru γ:

$$L(\gamma, v) = \left(\frac{v}{255}\right)^{\gamma} \cdot 255 \tag{3.13}$$



Slika 3.20: Graf za različite γ vrijednosti

Slika 3.21 prikazuje primjer posvjetljivanja tamne slike gamma korekcijom s parametrom $\gamma = 0.4$

Slika 3.21: Primjer povećanja svjetline gamma korekcijom

3.3. Upravljanje zasićenjem i tonom boje

U prošlom potpoglavlju su razmotrene metode obrade digitalne fotografije pomoću kojih se upravlja luminancijom. U ovom poglavlju su razmotrene metode upravljanja zasićenjem i tonom boje.

3.3.1. Alat HSV

Ovome alatu se pristupa izbornikom Color → HSV nakon čega se prikazuje prozor s tri klizača (engl. *sliders*) pomoću kojih se namještaju parametri. Alat HSV stvara linearni pomak u svakom od kanala HSV modela na cijeloj slici. Alat HSV ima tri parametra Δh, Δs i Δv koji određuju pomak u svakom od kanala HSV modela na cijeloj slici. Dakle, za svaki piksel na slici koji ima trojku vrijednosti $v_{HSV} = (h, s, v)$ gdje su h, s i v kut nijanse, saturacija i vrijednost iz HSV modela respektivno, definira se pomak $\Delta v_{HSV} = (\Delta h, \Delta s, \Delta v)$ pomoću kojega se definiraju nove vrijednosti piksela prema izrazu:

$$v'_{HSV} = v_{HSV} + \Delta v_{HSV} = (h + \Delta h, s + \Delta s, v + \Delta v) \tag{3.14}$$

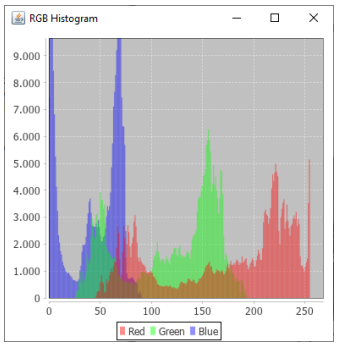
gdje je v'_{HSV} nova trojka piksela. Ovaj postupak se provodi za svaki piksel na slici.

Na slici 3.22 prikazan je primjer upravljanja nijansom boje alatom HSV. Na histogramu se očituje kao redistribucija svakog kanala boja. Također, vidi se pikselizacija prijelaza boja na slici b) što jest nedostatak ovo postupka. Nadalje, na istoj slici 3.22 je prikazano i upravljanje saturacijom boje. Smanjivanje saturacije na histogramu se očituje kao približavanje kanala dominantnom kanalu (u ovom slučaju je to kanal crvene boje), a povećanje saturacije ima suprotan učinak na kanale.

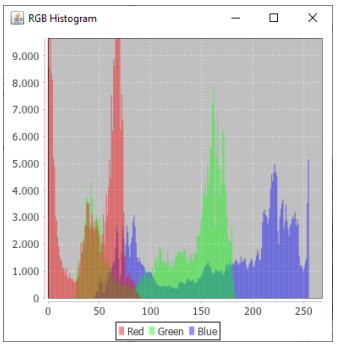
Djelovanje parametra Δv se neće posebno analizirati zato što ima identičan utjecaj na sliku kao metoda linearnog upravljanja svjetlinom koja je razmotrena u prošlom potpoglavlju.

(a) Orginalna slika

(b) $\Delta h > 0$



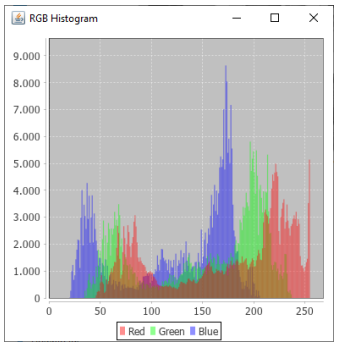
(c) Histogram od a)



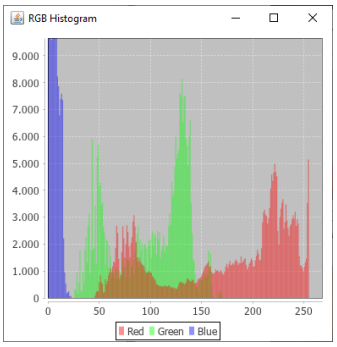
(d) Histogram od b)

(e) $\Delta s < 0$

(f) $\Delta s > 0$



(g) Histogram od e)



(h) Histogram od f)

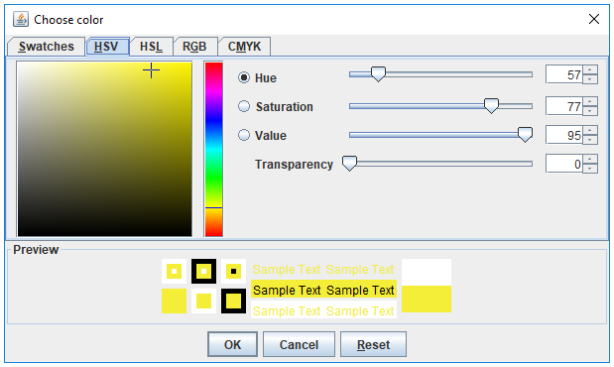
Slika 3.22: Upravljanje nijansom i saturacijom boje

3.3.2. Bojanje slike u zadanu boju

Ovome alatu pristupa se izbornikom Color → Tint nakon čega se otvara prozor u kojemu korisnik odabire boju (slika 3.23). Nakon odabrane boje, uzimaju se vrijednosti h_t, s_t koje predstavljaju kut nijanse i saturaciju boje iz HSV modela respektivno. Tada, za svaki piksel slike, koji ima trojku vrijednosti $v_{HSV} = (h, s, v)$ gdje su h, s i v kut nijanse, saturacija i vrijednost iz HSV modela respektivno, definira se transformacija:

$$v'_{HSV} = (h_t, s_t, v) \tag{3.15}$$

gdje je v'_{HSV} nova trojka piksela. Dakle, za svaki piksel se postavlja nijansa boje i saturacija od boje koju je korisnik odabrao. Na slici 3.24 se nalazi primjer bojanja slike u ljubičastu nijansu.



Slika 3.23: Odabir boje

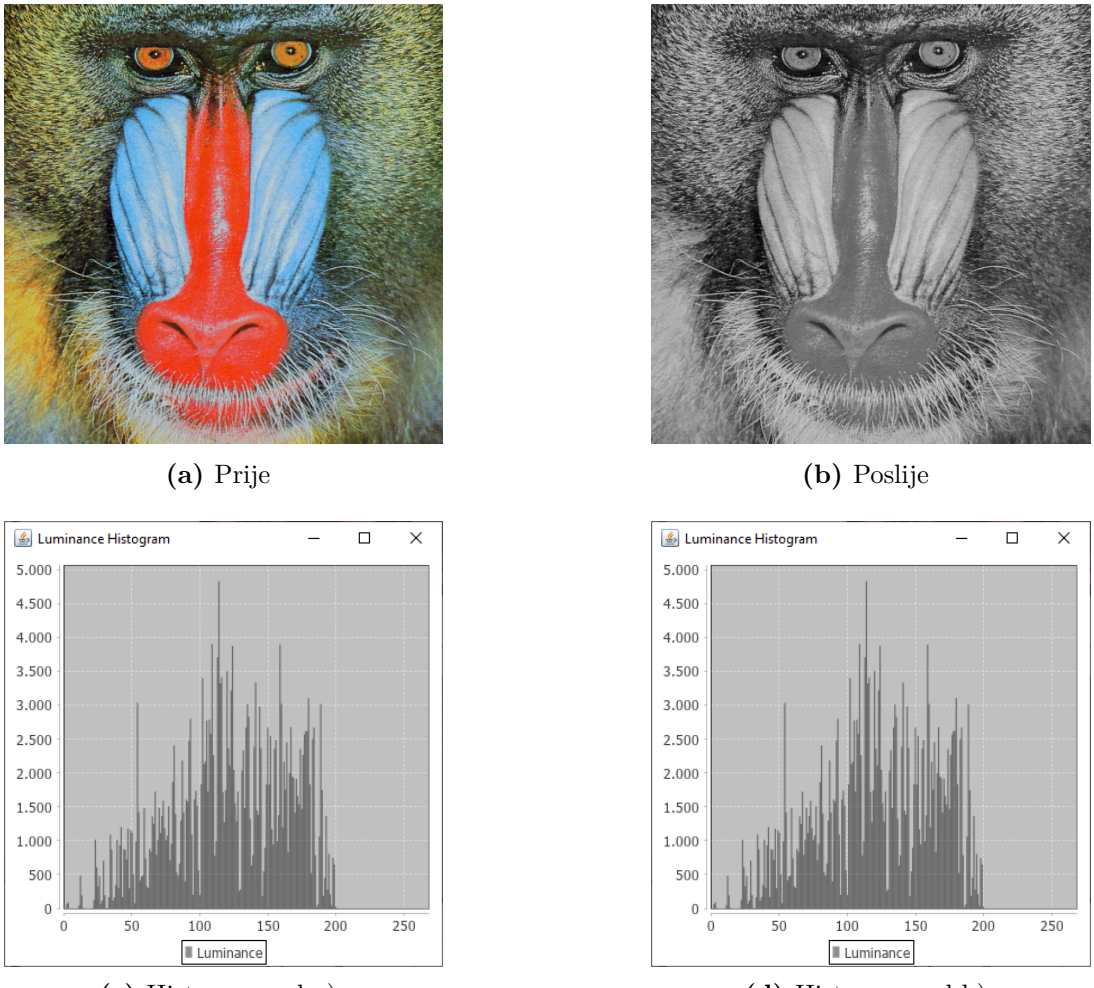
Slika 3.24: Primjer bojanja slike

3.3.3. Pretvorba u sive tonove

Ovdje se razmatra pretvorba slike u boji u crno-bijelu sliku (engl. *grayscale*). Alatu za pretvorbu u crno-bijelu sliku se pristupa izbornikom Color → to Grayscale. Pri pretvorbi RGB piksela u crno-bijeli piksel važno je zadržati luminantnu vrijednost piksela [7]. Za svaki model boje to se radi na drugačiji način ovisno o zastupljenosti boja po kanalima. Budući da se ovdje radi pretvorba iz RGB modela pomoću biblioteke OpenCV, određivanje približne luminantne vrijednosti piksela Y' dan je izrazom [3]:

$$Y' = 0.229 \cdot R + 0.587 \cdot G + 0.114 \cdot B \tag{3.16}$$

gdje su R, G i B vrijednosti crvene, zelene i plave boje iz RGB modela.



(a) Prije

(b) Poslije

(c) Histogram od a)

(d) Histogram od b)

Slika 3.25: Primjer pretvorbe slike u boji u crno-bijelu sliku

Na slici 3.25 je dan primjer pretvorbe. Histogrami od slika a) i b) su identični.

3.4. Filteri

U ovom poglavlju razmotrena je uporaba filtera na digitalne fotografije kroz alat PhotoGrade. Filteri se obično koriste za povećavanje i smanjivanje šuma. U alatu PhotoGrade implementirane su metode filtriranja koje rezultiraju u izoštravanju slike ili zamućenju slike.

3.4.1. Zamućenje

Za postizanje efekta zamućenja korištena je metoda Gaussovog uglađivanja (engl. *Gaussian blur*). Tipično se koristi za smanjenje šuma ili detalja u slikama [2]. Metoda se zasniva na primjeni funkcije Gaussove distribucije na sliku. Gaussova distribucija u 2D prostoru dana je izrazom:

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.17}$$

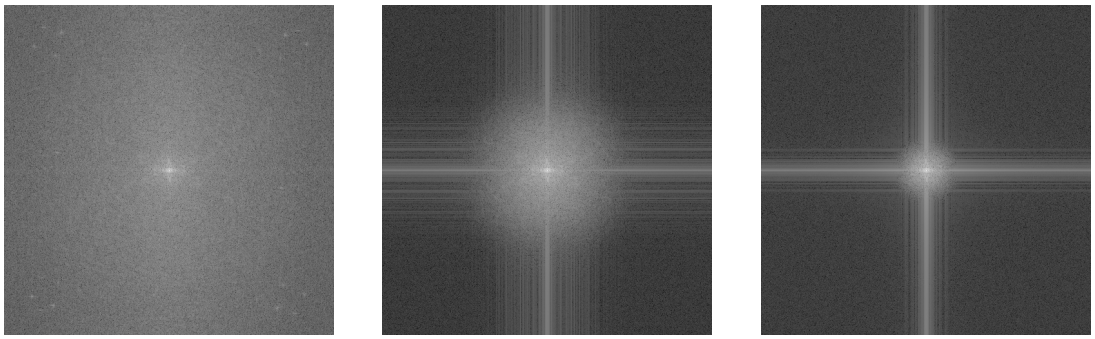
gdje su x i y udaljenosti po apscisi i ordinati, a σ standardna devijacija Gaussove distribucije. Na temelju vrijednosti distribucije konstruira se kvadratna matrica reda $2n-1$, gdje je $n \in \mathbb{N}_0$, koja predstavlja jezgru (engl. *kernel*) za konvoluciju sa slikom. 2D konvolucija se provodi tako da se središte jezgrene matrice poravna s elementom nad kojim se provodi transformacija. Poravnate vrijednosti iz jezgre i elementi iz niza se pomnože te međusobno sumiraju. Rezultat je konvolucija za element nad kojim se provodi transformacija. Takav postupak se provodi za svaki element u dvodimenzionalnom nizu, odnosno za svaki piksel na slici. Primjer jezgrene matrice reda 5 za Gaussovo uglađivanje [8]:

$$\frac{1}{256} \cdot \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \tag{3.18}$$

Gaussovo uglađivanje djeluje kao niskopropusni filtar što je vidljivo na FFT grafovima (slika 3.26) koji prikazuju prirodni logaritam modula frekvencija Fourierove transformacije.

Alatu za zamućivanje slike u PhotoGrade-u pristupa se izbornikom Filter → Blur nakon čega se otvara prozor u kojemu se pomoću klizača (engl. *slider*) namješta red Gaussove jezgrene matrice.

(a) Izvorna slika **(b)** Jezgra reda 9 **(c)** Jezgra reda 29



(d) FFT graf od a) **(e)** FFT graf od b) **(f)** FFT graf od c)

Slika 3.26: Primjer zamućivanja slike metodom Gaussovog uglađivanja

3.4.2. Izoštravanje

Za izoštravanje slike korištena je metoda unsharp mask čiji se princip nadovezuje na metodu Gaussovim uglađivanjem. Slika koju izoštravamo reprezentirana je matricom $\boldsymbol{O}$ i uglađena Gaussovom metodom čiji je rezultat označen matricom $\boldsymbol{B}$. Slika dobivena metodom unsharp mask reprezentirana matricom $\boldsymbol{U}$ izvedena je izrazom [9]:

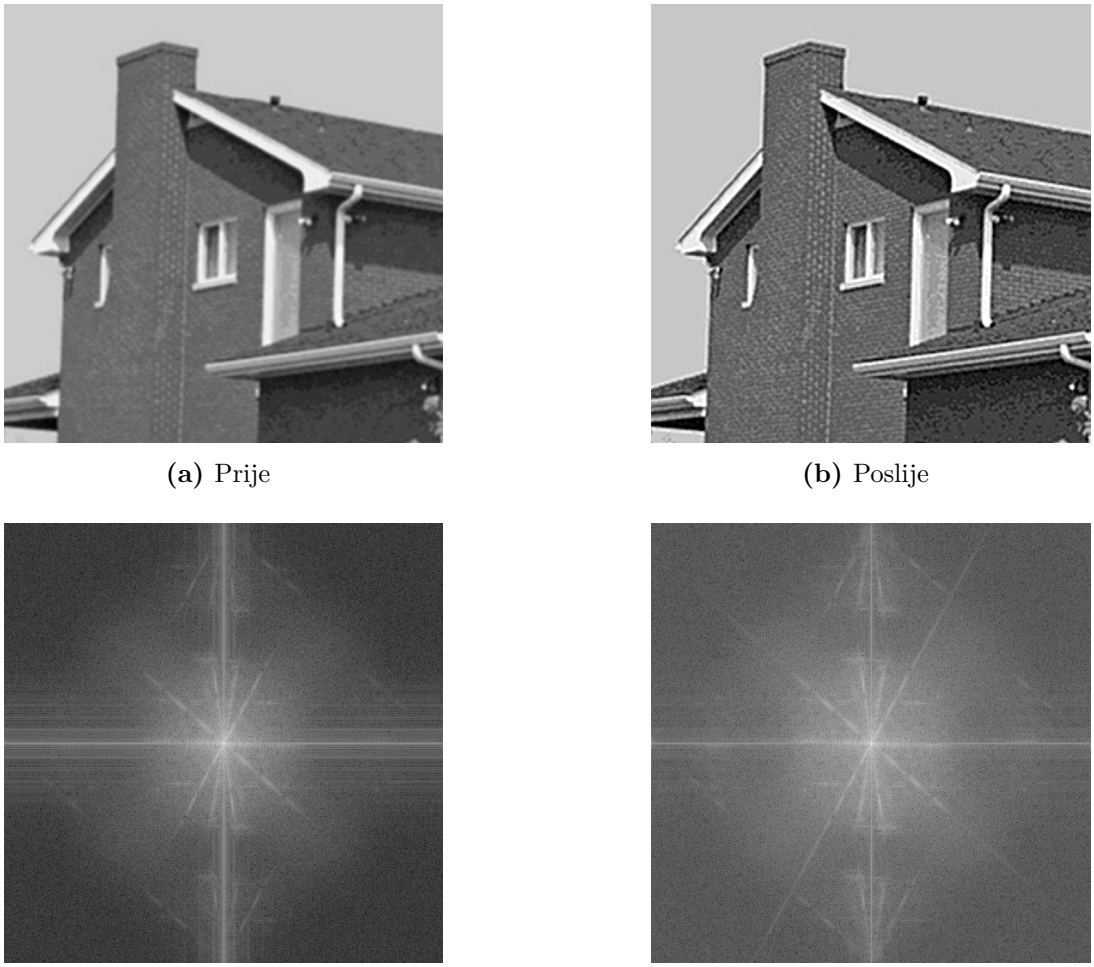
$$\boldsymbol{U} = \boldsymbol{O} + \alpha \cdot (\boldsymbol{O} - \boldsymbol{B}) \tag{3.19}$$

gdje je α koeficijent koji označava jačinu izoštravanja. Vrijedi $\alpha > 0$, $\alpha \in \mathbb{R}$. Na ovaj način se postiglo pojačavanje srednjih i visokih frekvencija. Naime, $\boldsymbol{O}$ u prosjeku sadržava najviše srednjih frekvencija dok $\boldsymbol{B}$ sadržava niske frekvencije. Njihovom razlikom $\boldsymbol{O} - \boldsymbol{B}$ dobivamo srednje i visoke frekvencije koje su pojačane koeficijentom α. Zatim na izvornu sliku dodajemo ovako dobivenu razliku. Budući da metoda unsharp mask pojačava visoke frekvencije rezultat će biti i povećanje šuma u slici.

U PhotoGrade-u unsharp mask alat pokreće se izbornikom Filter → Sharpen nakon čega se otvara prozor s dva klizača (engl. *slider*) pomoću kojih se

namještaju parametri α te red jezgre Gaussovog uglađivanja.

Slika 3.27 prikazuje primjer izoštravanja slike. FFT grafovi ukazuju da je metoda unsharp mask visokopropusni filtar. Djelovanje izoštravanja jasno se vidi na detaljima kuće.



(a) Prije

(b) Poslije

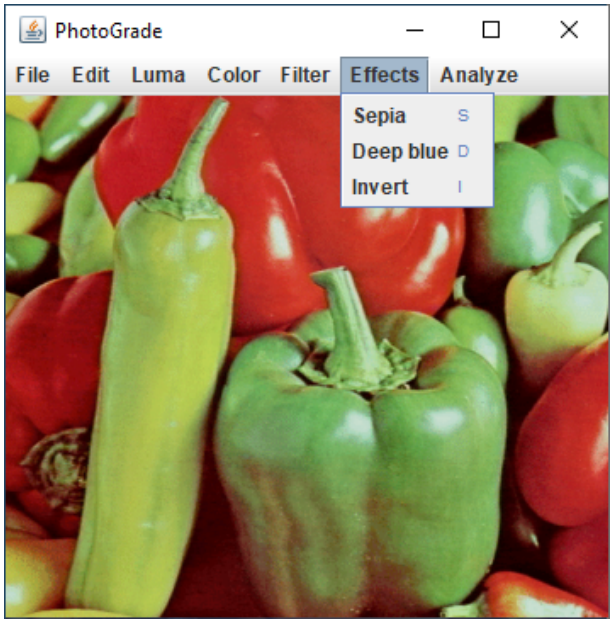
(c) FFT graf od a)

(d) FFT graf od b)

Slika 3.27: Primjer izoštravanja slike metodom unsharp mask

3.5. Efekti

Efekti su alati pomoću kojih korisnik jednostavno može urediti sliku. PhotoGrade implementira nekoliko efekata koji su navedeni i objašnjeni u ovom potpoglavlju. Svi efekti u alatu PhotoGrade nalaze se u izborniku 'Effects' (slika 3.28).



Slika 3.28: izbornik 'Effects'

3.5.1. Sepia

Sepia je efekt koji je rasprostranjen u alatima za obradu slika. Radi se o metodi koja sliku oboja u crvenkastosmeđu boju te joj tako daje izgled starenja kod starih fotografija ili izgled starijih fotografija koje su kemijski tretirane zbog vizualnih efekata ili u svrhu arhiviranja [10].

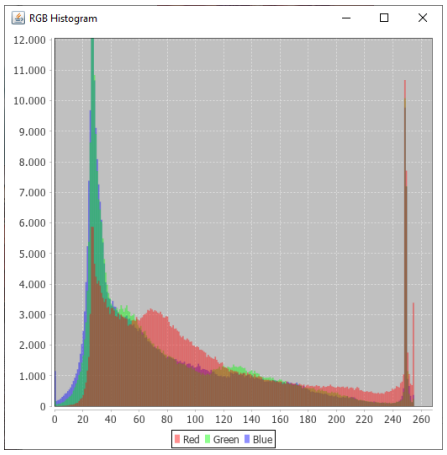
Sliku se boja u sepia ton pomoću transformacijske matrice koja se množi sa svakim pikselom $\boldsymbol{V} = \begin{bmatrix} B & G & R \end{bmatrix}^T$, gdje su B, G i R vrijednosti plave, zelene i crvene nijanse po RGB modelu. Transformacijska matrica dana je izrazom [11]:

$$\boldsymbol{T}_{sepia} = \begin{bmatrix} 0.272 & 0.534 & 0.131 \\ 0.349 & 0.686 & 0.168 \\ 0.393 & 0.769 & 0.189 \end{bmatrix} \tag{3.20}$$

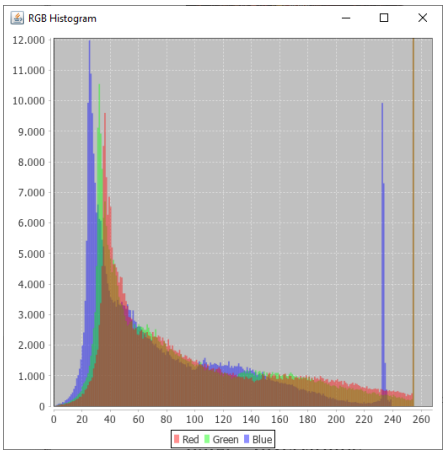
Sepia efektu se pristupa izbornikom Effects → Sepia.

(a) Prije

(b) Poslije



(c) Histogram od a)



(d) Histogram od b)

Slika 3.29: Primjer sepia efekta

3.5.2. Invertiranje

Invertiranje je postupak kojim se dobiva efekt fotografija razvijenih iz negativnog filma. Postupak invertiranja slike provodi se tako da se svaki element slike v_e, bila to komponenta boje ako je slika u boji ili vrijednost piksela ako se radi o crno-bijeloj slici (engl. *grayscale*), invertira pomoću izraza:

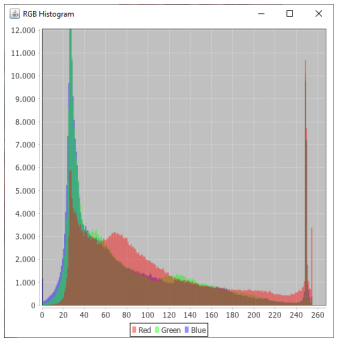
$$v'_e = 255 - v_e \tag{3.21}$$

gdje je v'_e nova vrijednost elementa slike. Pretpostavlja se da su slike 8-bitne pa se zato oduzima od broja $2^8 - 1 = 255$.

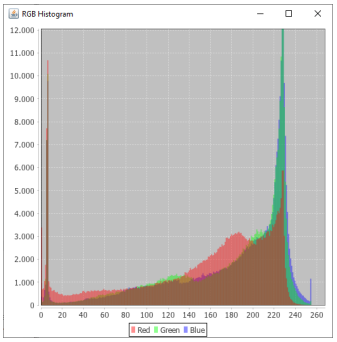
Na slici 3.30 nalazi se primjer invertiranja slike. Na histogramu invertirane slike očituje se zrcaljenje oko vertikalnog središta histograma.

(a) Prije

(b) Poslije



(c) Histogram od a)



(d) Histogram od b)

Slika 3.30: Invertiranje slike

3.5.3. Deep blue

Deep blue je efekt koji kombinira efekt sepia i efekt invertiranja. Rezultat jest bojanje slike u tamno plavu boju (slika 3.31). Deep blue efekt se postiže tako da sliku prvo invertiramo, zatim na takvu invertiranu sliku primijenimo sepia efekt te konačno dobivenu sliku još jednom invertiramo. Efektu deep blue pristupa se izbornikom Effects → Deep blue.

Slika 3.31: Primjer efekta deep blue

4. Programsko rješenje aplikacije PhotoGrade

Postoje mnogi alati i aplikacije koji pružaju mogućnost manipulacije slikovnim elementima na neki način. Matlab kao programski jezik sadrži nekoliko paketa s funkcijama koje programeru omogućavaju razne načine obrade slika. Također, programski jezici poput pythona su popularan izbor zbog velikog broja, koji i dalje raste, raznih biblioteka koje programerima omogućavaju široku paletu načina obrade i manipulacije slikama.

S druge strane, postoji potreba za gotovim programskim rješenjima koji nude mogućnosti obrade slika bez znanja programiranja. Dizajneri, fotografi, digitalni umjetnici pa čak i obični korisnici imaju potrebu za stvaranjem digitalnog slikovnog sadržaja ili oblikovanjem, doradivanjem i manipulacijom već gotovih digitalnih slika dobivenih iz digitalnih fotoaparata ili pametnih telefona. Dakle, potreba za aplikacijama koje nude mogućnosti uređivanje slika postoji za profesionalce i obične korisnike.

Danas postoje mnoge aplikacije koje pružaju navedene mogućnosti. Jedne od najpopularnijih su Photoshop i Gimp koje pružaju sveobuhvatne mogućnosti stvaranja novog slikovnog sadržaja ili obrađivanja postojećeg. Usporedno s njima, postoje aplikacije koje pružaju specifičnije funkcionalnosti koje su usmjerene samo na obradu digitalnih fotografija. Jedna takva aplikacije jest Lightroom za koju bi se moglo reći da je „mlađa sestra“ od aplikacije Photoshop jer ih je proizvela ista tvrtka Adobe.

Glavna svrha alata PhotoGrade jest pružiti korisniku jednostavan i intuitivan način za obradu fotografija. Iako alat nudi funkcionalnosti za tehničko poboljšanje fotografije, glavna namjena mu je obrada fotografije iz umjetničkih razloga. Korisnik kombiniranjem raznih alata unutar PhotoGrade-a može postići željeni rezultat. PhotoGrade je softversko rješenje napisano u programskom jeziku Java

uz pomoć biblioteke OpenCV[1] koja je korištena za izravnu manipulaciju slikovnim elementima. U sljedeća dva poglavlja objasnit će se sve funkcionalnost alata PhotoGrade.

4.1. Zahtjevi aplikacije

Načela oblikovanja programske potrebe nalažu da razvoj treba započeti s definiranjem funkcionalnih i nefunkcionalnih zahtjeva aplikacije. Funkcionalni zahtjevi opisuju kakve mogućnosti aplikacija pruža odnosno što sve može napraviti. S druge strane, nefunkcionalni zahtjevi opisuju kakva treba biti kvaliteta izvršavanja funkcionalnih zahtjeva.

4.1.1. Funkcionalni zahtjevi

U ovom odjeljku definirat će se funkcionalni zahtjevi aplikacije PhotoGrade koji bi trebali predstavljati minimalne funkcionalne zahtjeve svake aplikacije za obradu fotografija.

1. Učitavanje slika raznih formata pohranjenih u stalnoj memoriji
2. Pregled učitane slike neovisan o rezoluciji fotografije
3. Prikazivanje histograma slike
   (a) Luminantna komponenta
   (b) Svaki kanal boje zasebno
4. 2D transformacije nad slikom
   (a) Promjena veličine (engl. *resize*)
   (b) Odsijecanje (engl. *crop*)
   (c) Rotacija slike za 90° u oba smjera
   (d) Zrcaljenje – vertikalno i horizontalno (engl. *flip*)
5. Upravljanje luminantnom komponentom slike
   (a) Linearno upravljanje svjetlinom i kontrastom

[1] OpenCV (Open Source Computer Vision Library) jest biblioteka za računalni vid i strojno učenje.

(b) Upravljanje svjetlinom i kontrastom pomoću kubičnih splajn krivulja

(c) Odvojeno upravljanje svjetlinom na različitim luminantnim rasponima

i. Sjene

ii. Srednji tonovi

iii. Svijetli tonovi

(d) Upravljanje cjelokupnim luminantnim rasponom

(e) Gama korekcija

6. Upravljanje zasićenjem i tonom boje

(a) Linearna promjena HSV komponenata na cijeloj slici

(b) Bojanje slike u zadanu boju (engl. *tint*)

(c) Pretvorba slike u sive tonove, također se zove pretvorba u crno-bijelu sliku (engl. *grayscale*)

7. Primjena filtera

(a) Izoštravanje slike metodom unsharp mask

(b) Zamućenje slike metodom gaussian blur

8. Primjena unaprijed definiranih efekata

(a) Sepia efekt

(b) Deep blue efekt

(c) Inverzija slike (engl. *invert*)

9. Kretanje kroz povijest obrade fotografije

(a) Poništi promjene (engl. *undo*)

(b) Ponovi poništeno (engl. *redo*)

10. Usporedba izvorno učitane fotografije i obrađene fotografije

11. Prikaz promjena koje korisnik nad slikom u stvarnom vremenu

12. Pohrana uređenih slika u stalnu memoriju

13. Grafičko sučelje

Kao što je vidljivo iz navedenih funkcionalnih zahtjeva, oni su brojni čak i kod jednog skupa koji predstavlja minimalan oblik zahtjeva. U suštini, svaka aplikacija za obradu slike treba moću učitati sliku, obraditi je na neki način te pohraniti promijene. Dodatno, poželjan je zahtjev obrade slika u stvarnom vremenu tako da korisnici mogu odmah doživjeti promjene koje su napravili te im omogućiti veću slobodu i kreativnost pri uređivanju.

4.1.2. Nefunkcionalni zahtjevi

Nefunkcionalni zahtjevi navedeni u ovom odjeljku predstavljati će idealan skup nefunkcionalnih zahtjeva aplikacije PhotoGrade. Općenito programska rješenja imaju zadane idealne nefunkcionalne zahtjeve koje je u praksi često realizirati. Tako ni aplikacija PhotoGrade ne ispunjava ove zahtjeve u njihovoj potpunosti ili ih uopće ne ispunjava.

1. Učitavanje slika svih formata i rezolucija u trajanju kraćem od jedne sekunde

2. Aplikacija se niti u jednom trenutku ne smije srušiti usprkos korisnikovim radnjama

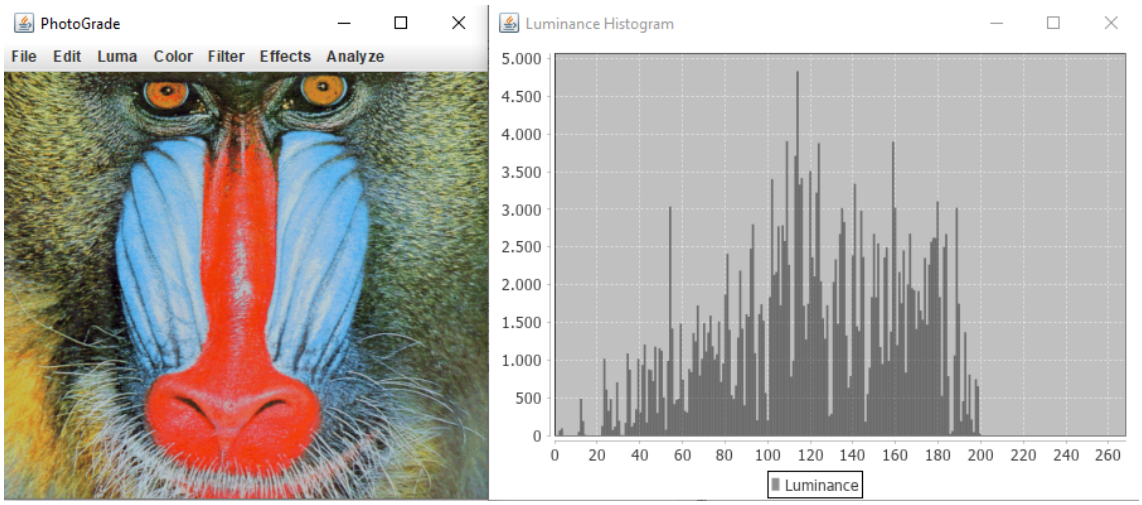
3. Prikaz promjena u stvarnom bez trzanja i zastajkivanja (engl. *lagging*)

4.2. Pomoćni alati u aplikaciji Photograde

U sklopu PhotoGrade-a nalaze se pomoćni alati koji pomažu i olakšavaju korisniku praćenje promjena koje je napravio odnosno objektivnije analiziranje slike koju uređuje. Prvi takav alat omogućuje korisniku usporedbu između izvorno učitane slike i trenutne slike koju je uredio. Ide se na izbornik Analyze → Toggle original/edited image nakon čega će se prikazati izvorna slika koju je korisnik učitao. Ponovnim odabirom navedenog alata se vraća trenutno uređena slika od korisnika.

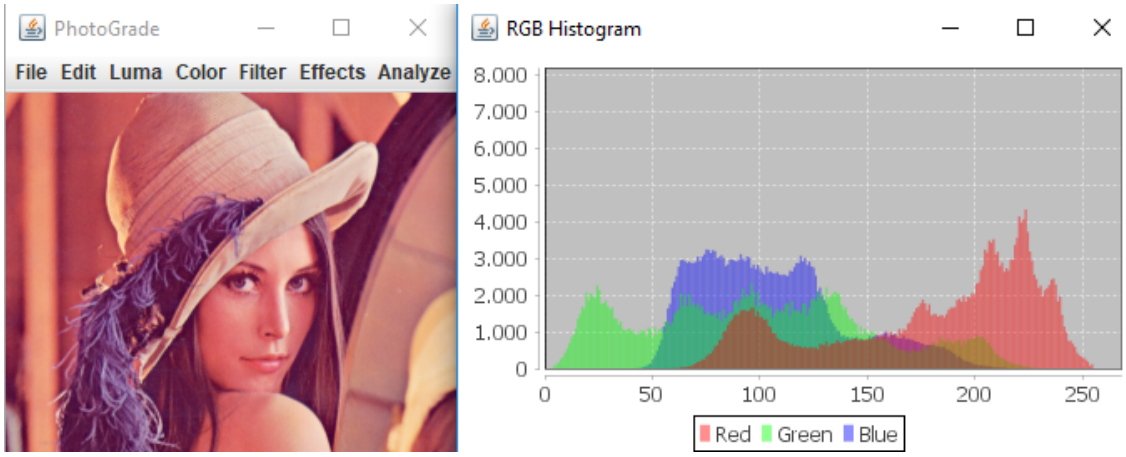
Sljedeći alat jest histogram koji prikazuje frekvencije luminantne komponente slike. Histogram jest općenito graf koji pobrojava frekvencije zadanih raspona vrijednosti. U slučaju 8-bitne slike, histogram pobrojava vrijednosti iz raspona $0 - 255$. Vizualno, histogram s lijeve strane prikazuje koliko sjena ima slika, s desne strane prikazuje koliko ima svijetlih tonova te u sredini prikazuje koliko srednjih tonova sadrži slika. Kako bi korisnik mogao vidjeti histogram luminantne

komponente (slika 4.1) slike koju uređuje ide na izbornik Analyze → Luminance Histogram.



Slika 4.1: Primjer histograma koji prikazuje luminantnu komponentu

Usporedno s histogramom luminantne komponente, PhotoGrade ima i histogram koji prikazuje svaki kanal zasebno. Kako bi korisnik mogao vidjeti histogram svakog kanala (slika 4.2) slike koju uređuje ide na izbornik Analyze → RGB Histogram. Ako korisnik trenutno radi s crno-bijelom slikom (engl. *grayscale*), navedeni histogram će jednostavno prikazati luminantnu komponentu slike.



Slika 4.2: Primjer histograma koji prikazuje svaki kanal zasebno

4.3. Alati za razvoj aplikacije

U ovom poglavlju razmotrit će se razni alati i načini na koji se može razviti programsko rješenje za uređivanje slika. Nadalje, demonstrirat će se razvoj apli-

kacije PhotoGrade kao jedan konkretan primjer razvoja aplikacije za uređivanje slika.

Pri razvoju bilo kojeg programskog rješenja treba se odlučiti o izboru programskog jezika u kojem će aplikacija biti razvijena. U uvodu je naveden matlab kao izbor programskog jezika u kojemu se mogu obrađivati slike. Iako matlab sadržava mnoge funkcije za obradu slika, ipak ne nudi dovoljne funkcionalnosti za razvoj samostalne aplikacije koju će korisnici moći pokrenuti na neovisnom operacijskom sustavu. Matlab ne nudi funkcionalnosti izrade grafičkog sučelja koje je preko potrebno za ovakvu vrstu aplikacije. Nadalje, matlab je interpretirani jezik što automatski postavlja usko grlo za daljnje optimizacije aplikacije koje bi bile potrebne za što bolje ispunjenje nefunkcionalnih zahtjeva.

Nadalje, izbor programskog jezika također ovisi o tome hoće li se manipulacija slikovnim elementima vršiti preko neke od biblioteka ili će se metode obrade slike programirati „iz nule". Potonje se preporuča jedino u slučaju kada velike tvrtke rade na velikom projektu i trebaju imati potpunu kontrolu nad funkcionalnošću i kodom. Dakle, preporučeno bi bilo koristiti neku od gotovih biblioteka. Razne biblioteke pružaju različitu fleksibilnost i mogućnosti pri manipulaciji slikovnim elementima.

Također, izbor programskog jezika ovisi o sustavu ili sustavima na kojima će se aplikacija izvršavati te načinu izvedbe grafičkog sučelja. Operativni sustavi Windows i OS X imaju mogućnost nativnog razvoja grafičkog sučelja, dok se za Linux sustave koriste specifikacije poput X Windows System ili Wayland. Dodatno, razvoj aplikacija za pametne telefone donosi sasvim drugačiji pristup izradi cjelokupne aplikacije pa tako i cijelog grafičkog sučelja. Usprkos različitim sustavima, postoje biblioteke koje nude podršku za razvoj grafičkih sučelja na više sustava odjednom. Neki programski jezici koji su razvijeni u svrhu višesistemske podrške nude u svojim standardnim bibliotekama podršku za razvoj više-sistemskih grafičkih sučelja. Npr. programski jezik Java sa svojim bibliotekama Swing i JavaFX te programski jezik C# s bibliotekama Mono i GTK#.

Iz svega navedenog vidljivo je da izbor programskog jezika, biblioteke za obradu slikovnih elemenata te izbor biblioteke za izradu grafičkog sučelja ovisi o puno parametara te za sobom povlači razna ograničenja. Aplikacija PhotoGrade napravljena je u programskom jeziku Java. Korištena je biblioteka Swing za izradu grafičkog sučelja te ekstenzija biblioteke OpenCV za manipulaciju slikovnim elementima. Ovakav izbor pruža više-sistemsku podršku te koristi visoko fleksibilnu biblioteku za manipulaciju slikovnim elementima. Tablica 4.1 prika-

zuje razne biblioteke za razvoj grafičkog sučelja i manipulaciju slikovnih elemenata za nekoliko odabranih programskih jezika.

Tablica 4.1: Programski jezici i biblioteke

Programski jezik	**Biblioteke za manipulaciju slikovnim elementima**	**Biblioteke za izradu grafičkog sučelja**
Python	Scikit-image	Kivy
	OpenCV	WxPython
	Mahotas	libavg
	SimpleTK	Pyglet
	SciPy	PyGtk
	Pillow	PyQt
	Matplotlib	
Java	OpenCV	Swing
	BoofCV	JavaFX
	Deeplearning4j	SWT
	JavaCV	Apache Pivot
	AlgART	
C++	OpenCV	Qt
	dlib	GTK+
	ITK	wxWidgets
	OTB	JUCE

4.3.1. CUDA i OpenCL

Obrada slike u stvarnom vremenu jest računalno zahtjevno te je potrebno izvođenje slikovnih operacija prebaciti na grafičku karticu (engl. *Graphical processing unit – GPU*). Današnji relevantni standardi za opće-namjensko procesiranje na grafičkim karticama su CUDA i OpenCL. Slučaj u kojem se razmatra programiranje obrada slika „iz nule" podrazumijeva da bi se operacije nad slikama trebale implementirati pomoću navedenih standarda. CUDA je programsko sučelje koju je razvila tvrtka Nvida za svoje grafičke kartice. OpenCL je otvoreni standard kojeg održava neprofitna organizacija Khronos Group. Oba standarda su široko korištena ovisno o namijeni.

Prednost koje grafičke kartice imaju za razliku od običnih procesora (engl. *Central processing unit - CPU*) jest paralelno izvođenje mnogih jednostavnih operacija odjednom. Kod običnih procesora, izvršavanje operacija je sekvencijalno. Dakako, obični procesori mogu izvršavati puno kompleksnije operacije od grafičkih kartica. Prva namjena grafičkih kartica bila je prikazivanje kompleksnijih 3D grafičkih elemenata. Naime, procesiranje i vizualizirane grafičkih elemenata zahtijeva izvršavanje puno jednostavnih paralelnih operacija. Dodatni razvoj tehnologije doveo je do uviđanja potencijala grafičkih kartica kao opće-namjenskih jedinica za paralelno procesiranje te su tako nastali standardi CUDA i OpenCL. Kako su digitalne slike nakupina mnogo jednostavnih slikovnih elemenata koje treba paralelno obradi, tako je njihova obrada na grafičkim karticama logičan izbor.

Navedeni standardi se danas široko primjenjuju u područjima obrade videa i slike, računalnom vidu, neuronskim mrežama te u raznim zadacima kojima pogoduje paralelno izvršavanje. Problem s kojim su se grafičke kartice oduvijek susretale jest usko grlo prebacivanja podataka iz radne memorije (RAM) u memoriju grafičke kartice (VRAM). Naime, aplikacije koje trebaju raditi u stvarnom vremenu i koriste procesiranje na grafičkim karticama moraju pažljivo odabirati trenutke kada će i koliko podataka prebaciti u memoriju grafičke kartice kako ne bi došlo do zastoja.

4.3.2. OpenCV

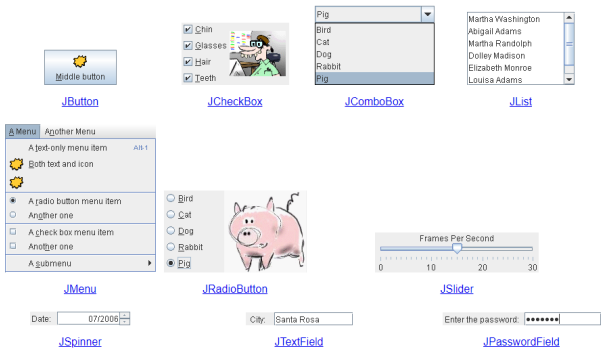
Jedna od najpoznatijih i najfleksibilnih biblioteka za manipulaciju slikovim elementima jest OpenCV – biblioteka za računalni vid u stvarnom vremenu. Primarno je pisana u jeziku C++, ali sadrži ekstenzije za mnoge druge popularne programske jezike poput Jave i Pythona. OpenCV svoje funkcije implementira kroz CUDA i OpenCL sučelje zbog najveće efikasnosti. Automatski detektira komponente sustava na kojemu se izvodi te koristi prigodne optimizirane funkcije na grafičkim karticama ako je to moguće.

OpenCV sadrži mnoge funkcije za obradu i manipulaciju slika, učitavanje i spremanje slika, primjena raznih transformacija te mnoge optimizirane algoritme za računalni vid i strojno učenje. Centralni razred koji predstavlja slike u biblioteci jest razred Mat. Podržava spremanje slika različitih tipova i dimenzija.

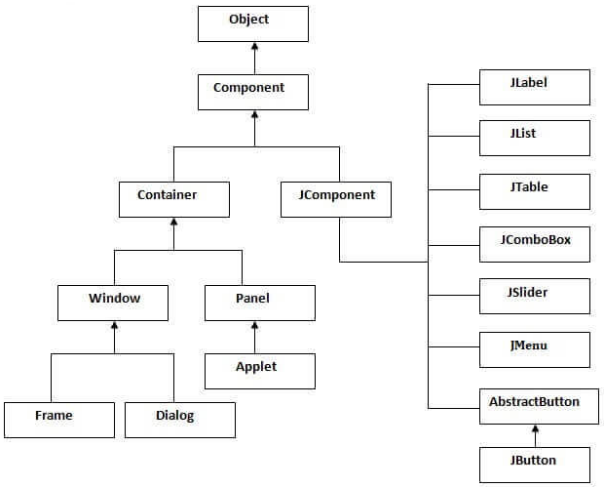
4.4. Razvoj grafičkog sučelja u aplikaciji PhotoGrade

U ovom poglavlju razmotrit će se razvoj grafičkog sučelja (GUI) u programskom jeziku Java uz pomoć biblioteke Swing za aplikaciju PhotoGrade. Općenito, grafičko sučelje sastoji se od raznih vrsta gumbova, izbornika, klizača, tekstualnih okvira i oznaka, slika, itd. (slika 4.3). Korisnik komunicira s grafičkim sučeljem preko tipkovnice i miša.

U Swing-u grafički elementi se zovu komponente koje koji se mogu smjestiti u razne kontejnere koji imaju svoj zadani razmještaj elemenata (engl. *layout*). Swing ima svoju razvijenu hijerarhiju razreda iz koje su izgrađene razne komponente, kontejneri i naposljetku sami prozori (engl. *window*). Navedena hijerarhija prikazana je na slici 4.4.



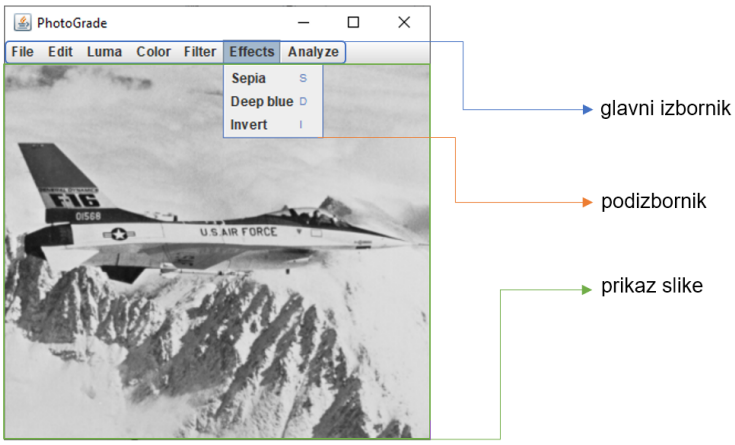
Slika 4.3: Osnovni elementi Swing GUI-a



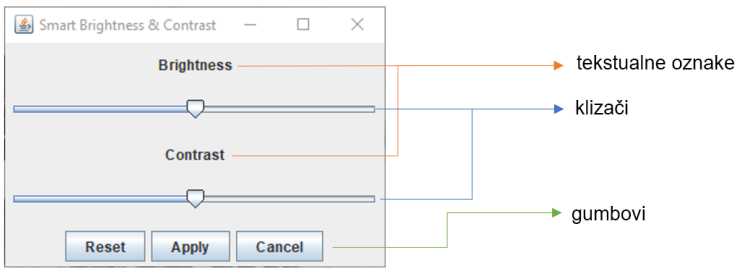
Slika 4.4: Hijerarhija razreda u Swing-u

Kako je izvođenje koda koji je zadužen za grafičko sučelje odvojen od glavne logike programa, on se izvodi na drugoj dretvi (engl. *thread*) koja je zadužena samo za prikazivanje grafičkih elemenata i njihovu asinkronu interakciju s drugim dijelovima programa.

Glavni prozor u aplikaciji PhotoGrade (slika 4.5) sastoji se od glavnog izbornika koji je izveden pomoću komponente JMenuBar te prikaz slike koja se obrađuje. Prikaz slike je izveden pomoću komponente JLabel. Klikom na neki od izbornika otvara se podizbornik (JMenu) s dodatnim alatima i izbornicima. Odabirom nekog od alata koji se nalaze u podizbornicima otvara se dodatni prozor za određivanje parametara alata. Parametri se namještaju pomoću klizača (engl. *slider*) izvedenih iz komponente JSlider. Klikom na gumb Reset korisnik vraća parametre u prvobitno stanje. Gumb Apply potvrđuje i sprema promjene na slici, gumb Cancel poništava promjene i zatvara alat. Slika 4.6 prikazuje prozor alata za upravljanje svjetlinom i kontrastom. Pokretanjem klizača mijenjaju se parametri alata te korisnik u stvarnom vremenu vidi promjene na slici.



Slika 4.5: Glavni prozor aplikacijse PhotoGrade



Slika 4.6: Alat za upravljanje svjetlinom i kontrastom

Svaki izbornik izveden je iz razreda Action koji je opće namjenski razred koji pohranjuje akcije koje sadrže naziv i callback metodu koja će se izvesti kada se akcija okine (engl. *trigger*). Akcije, također, mogu sadržavati sliku i akcelerator – prečac za okidanje akcije. Izvorni kod 4.1 prikazuje stvaranje akcije Undo:

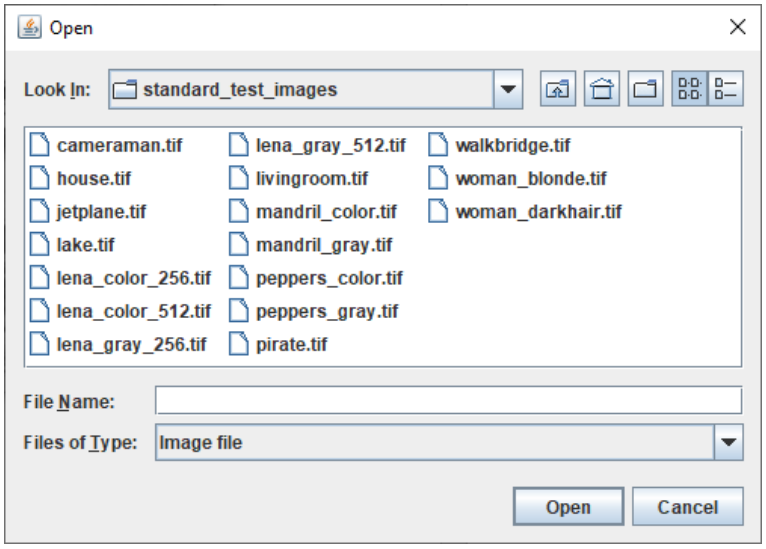
- Naziv: „Undo“
- Akcelerator: „ctrl + z“
- actionPreformed() je callback funkcija

```
1 private Action actionUndo = new MyAction("Undo", "control Z", this){
2     @Override
3     public void actionPerformed(ActionEvent e) {
4         photoChangeTracker.undo();
5         showCurrentImge();
6     }
7 };
```

Izvorni kod 4.1: Stvaranje akcije Undo

Dodavanje akcija u izbornik vrši se metodom add() pozvanom nad objektima razreda JMenu.

Učitavanje fotografija vrši se izbornikom File → Open. Zatim se otvara prozor izveden razredom JFileChooser koji omogućava odabiranje datoteka na računalu (slika 4.7).



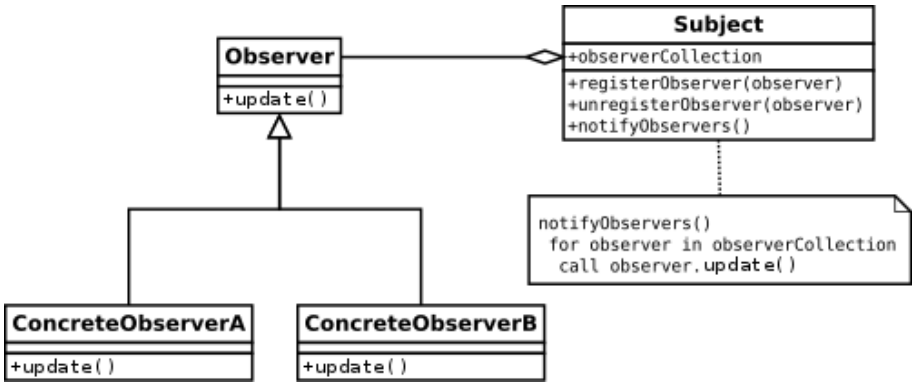
Slika 4.7: Prozor za učitavanje slika

4.4.1. Oblikovni obrazac promatrača

Grafičko sučelje je sustav temeljen na događajima. Kada korisnik klikne gumb, nešto se treba dogoditi. Kada korisnik u aplikaciji PhotoGrade pomakne klizač, očekivano je da će se slika osvježiti i prikazati trenutno stanje uređene slike s promijenjenim parametrom koji je bio pridružen navedenom klizaču. Događaj pritiska na gumb asinkrono treba pozvati neku funkciju koja će izvesti određeni dio koda. Funkcije koje se pokreću na asinkrone događaje općenito se nazivaju callback funkcije.

Oblikovni obrazac koji rješavam ovakav problem asinkronih događaja naziva se obrazac promatrača (slika 4.8). Promatrači su objekti koji osluškuju događaje. One će se pretplatiti subjektima da ih obavijeste kada se dogodi određeni događaj. Kada ih subjekt obavijesti o određenom događaju, promatrači izvršavaju pripadne callback funkcije.

U programskom jeziku Java, promatrači i subjekti implementirani su kao sučelja. Promatrači sadrže callback metodu update(). Subjekti sadrže metode registerObserver(observer) i unregisterObserver(observer) pomoću kojih se promatrači mogu pretplatiti i raskinuti pretplatu od subjekta. Naravno, subjekti sadrže metodu notifyObservers() koja će obavijestiti sve promatrače o događaju na koji su se pretplatili.

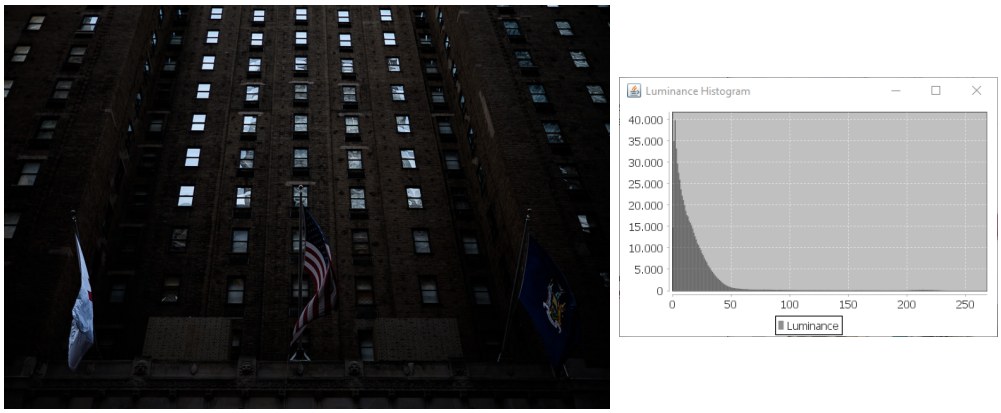


Slika 4.8: Oblikovni obrazac promatrača

5. Ispitivanje funkcionalnosti aplikacije PhotoGrade

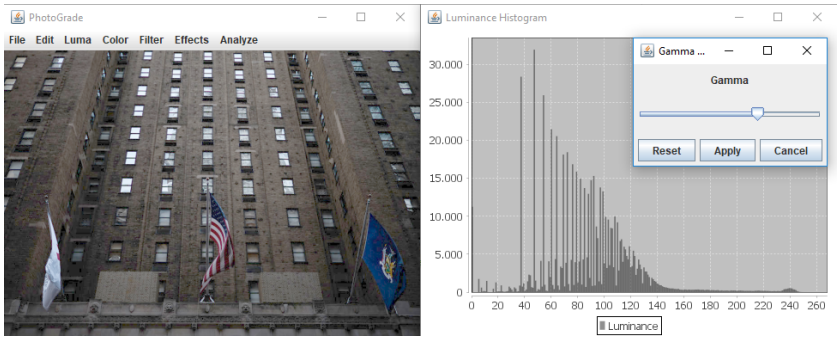
5.1. Obrada tamne fotografije

Radi se s pretamnom slikom (slika 5.1, preuzeto s `https://photographylife.com/underexposure-and-overexposure-in-photography`). Histogram ukazuje kako se većina vrijednosti nalazi u području sjena.

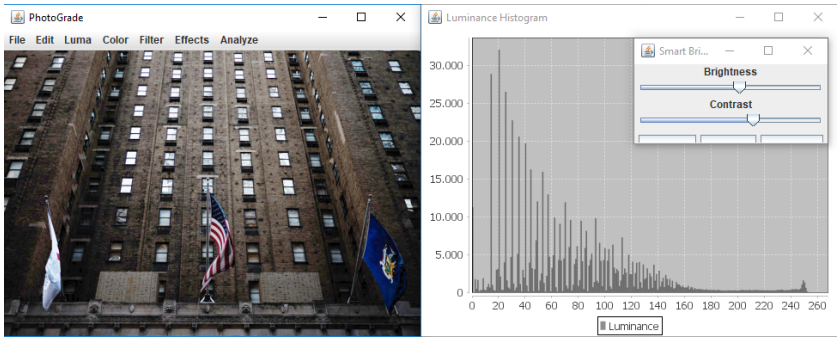


Slika 5.1: Izvorna slika i njen histogram luminantne komponente

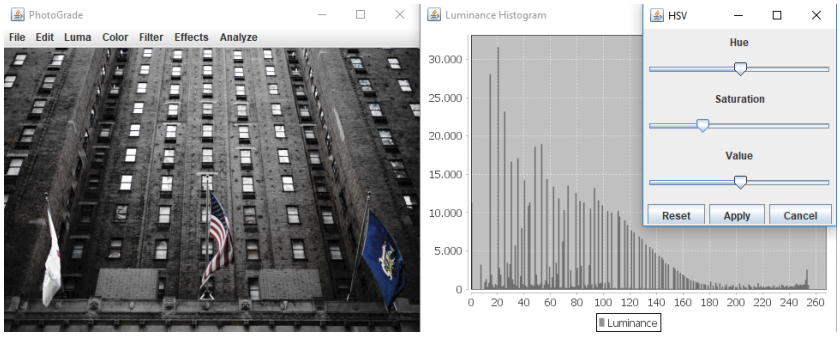
Cilj je posvijetliti sliku i istaknuti boje na zastavama. U prvom koraku (slika 5.2) je napravljena gamma korekcija. Zatim je blago pojačan kontrast pomoću alata 'Smart Brightness & Contrast' (slika 5.3). Potom je pomoću alata HSV smanjena saturacija boje na cijeloj slici čime je postignut efekt izblijedjelih boja oko zastava (slika 5.4). Slika 5.5 prikazuje usporedbu izvorne i uređene slike. Slika je posvijetljena te se boje ističu samo na zastavama.



Slika 5.2: Korak 1 - gamma korekcija



Slika 5.3: Korak 2 - pojačanje kontrasta



Slika 5.4: Korak 3 - smanjenje saturacije

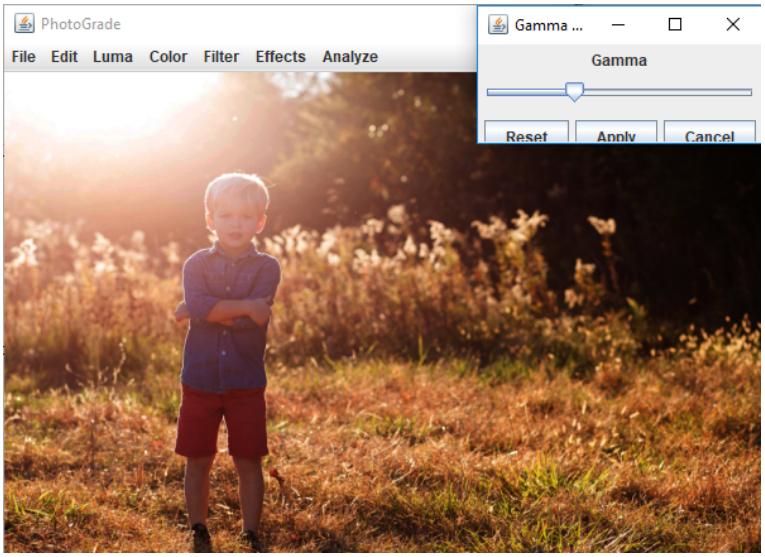
Slika 5.5: Usporedba izvorne i uređene slike

5.2. Obrada svijetle fotografije

Radi se sa slikom 5.6 (preuzeta s `https://photo.stackexchange.com/questions/84495/can-my-overexposed-hazy-backlit-photo-be-fixed`) koja je presvijetla. Cilj je urediti fotografiju tako da izaziva osjećaje nostalgije.

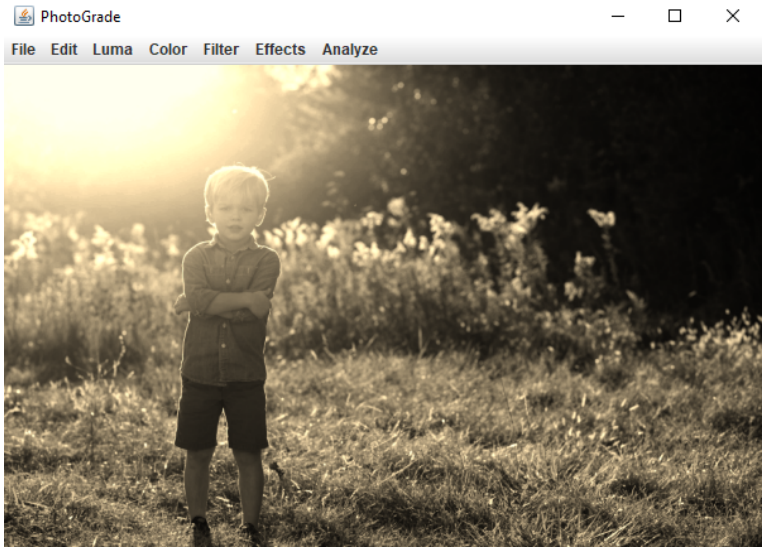
Slika 5.6: Izvorna slika

U prvom koraku (slika 5.7) se radi gamma korekcija kako bi se nelinearno smanjila svjetlina na slici.



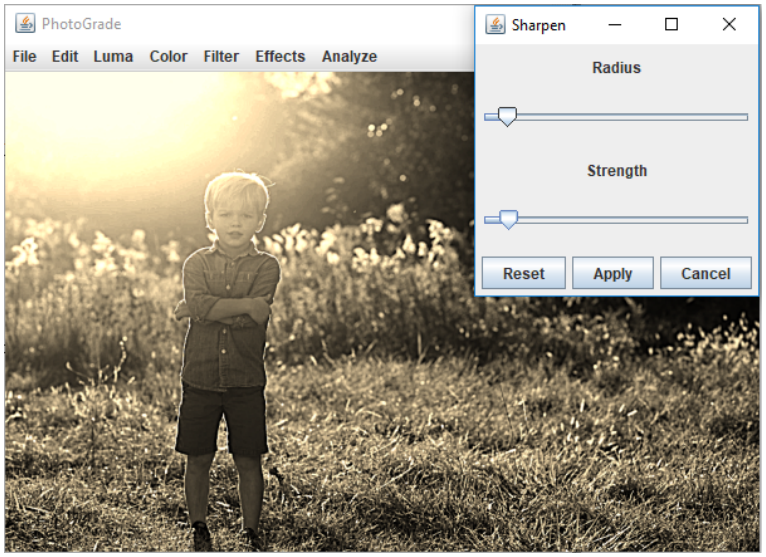
Slika 5.7: Korak 1 - gamma korekcija

Zatim se primjenjuje efekt sepia (slika 5.8) kako bi slika poprimila staromodan izgled.



Slika 5.8: Korak 2 - sepia efekt

Sljedeće je slika blago izoštrena (slika 5.9) kako bi dječakovo lice bilo istaknutije te kako bi se povećala zrnatost koja je svojstvena starijim fotografijama.



Slika 5.9: Korak 3 - izoštravanje

Slika 5.10 prikazuje usporedbu izvorne i uređenih slika. Druga uređena slika je odrezana kako bi se ostvarila centralna kompozicija.

Slika 5.10: Usporedba izvorne i uređene slike

5.3. Obrada fotografija iz umjetničkih razloga

5.3.1. Primjer 1

Radi se sa slikom 5.11 koja se želi obraditi iz umjetničkih razloga. Dakle, cilj nije tehnički popraviti fotografiju, već izazvati pobuđivanje osjećaja kod gledatelja. Način na koji umjetnička fotografija pobuđuje osjećaje kod gledatelja neće biti opisan u ovome radu, već će se objasniti postupak kojim je slika preobražena u alatu PhotoGrade.

Slika 5.11: Izvorna slika

U prvom koraku (slika 5.12) slika je pretvorena u crno-bijelu sliku (engl. *grayscale*).

Slika 5.12: Korak 1 - pretvorba u crno-bijelu sliku

U drugom koraku (slika 5.13) primijenjeno je vertikalno zrcaljenje.

Slika 5.13: Korak 2 - vertikalno zrcaljenje

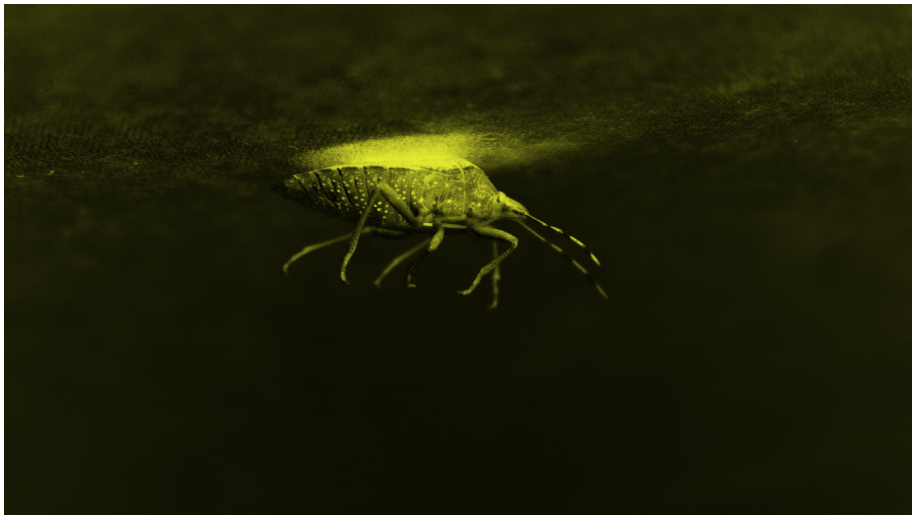
U trećem koraku (slika 5.14) napravljena je inverzija slike kako bi se postigao efekt negativa filma.

Slika 5.14: Korak 3 - inverzija

U četvrtom koraku (slika 5.15) pojačan je kontrast pomoću alata 'Smart Brightness & Contrast'.

Slika 5.15: Korak 4 - pojačanje kontrasta

U petom koraku (slika 5.16) slika je pobojana žutom bojom te je boja dodatno prilagođena po želji alatom HSV.



Slika 5.16: Korak 5 - bojanje slike

U završnom koraku slika je još odrezana kako bi se odrezao nepotreban prazan prostor te centrirao kukac. Slika 5.17 prikazuje usporedbu izvorne i uređene fotografije.

Slika 5.17: Usporedba izvorne i uređene slike

5.3.2. Primjer 2

Radi se sa slikom 5.18 koja se želi obraditi iz umjetničkih razloga. Cilj je obraditi fotografiju na način da izgleda nešto nalik crtežu.

Slika 5.18: Izvorna slika

U prvom koraku (slika 5.19) slika je pretvorena u crno-bijelu sliku (engl. *grayscale*).

Slika 5.19: Korak 1 - pretvorba u crno-bijelu sliku

U drugom koraku (slika 5.20) slika je izoštrena.

Slika 5.20: Korak 2 - vertikalno zrcaljenje

U trećem koraku (slika 5.21) povećan je kontrast i prilagođena svjetlost.

Slika 5.21: Korak 3 - prilagodba svjetline i kontrasta

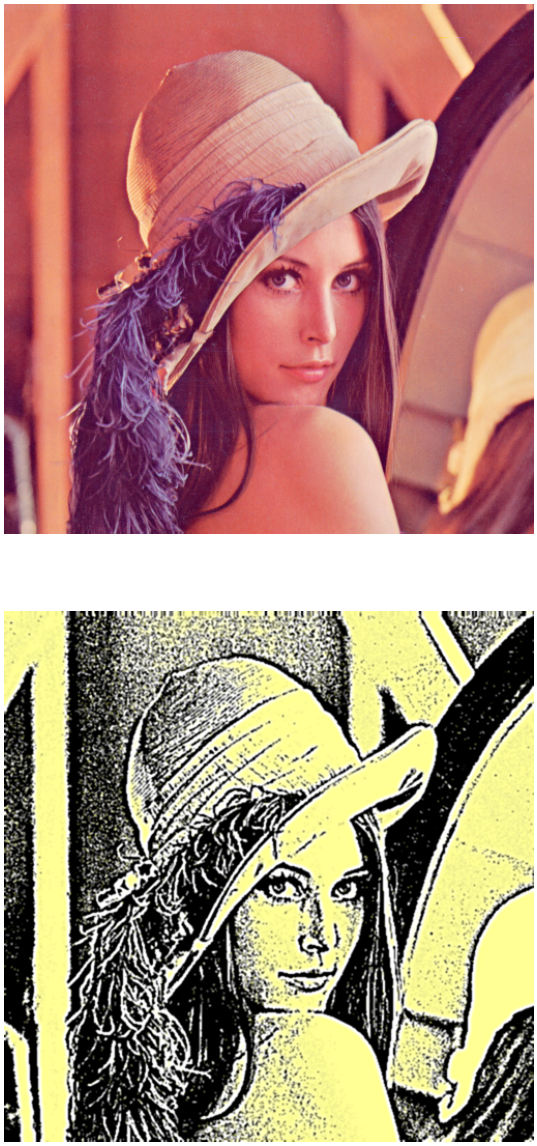
U četvrtom koraku (slika 5.22) slika je obojana u nijansu žute.

Slika 5.22: Korak 4 - bojanje slike

U petom koraku (slika 5.23) slika je dodatno izoštrena i zamućena kako bi se dobio željeni efekt.

Slika 5.23: Korak 5 - uglađivanje oštrine

Slika 5.24 prikazuje usporedbu izvorne i uređene fotografije. Željeni efekt je postignut pomoću alata PhotoGrade



Slika 5.24: Usporedba izvorne i uređene slike

5.3.3. Primjer 3

Radi se s fotografijom 5.25. Cilj je poigrati se s granama na drvetu koje nalikuju na paukovu mrežu.

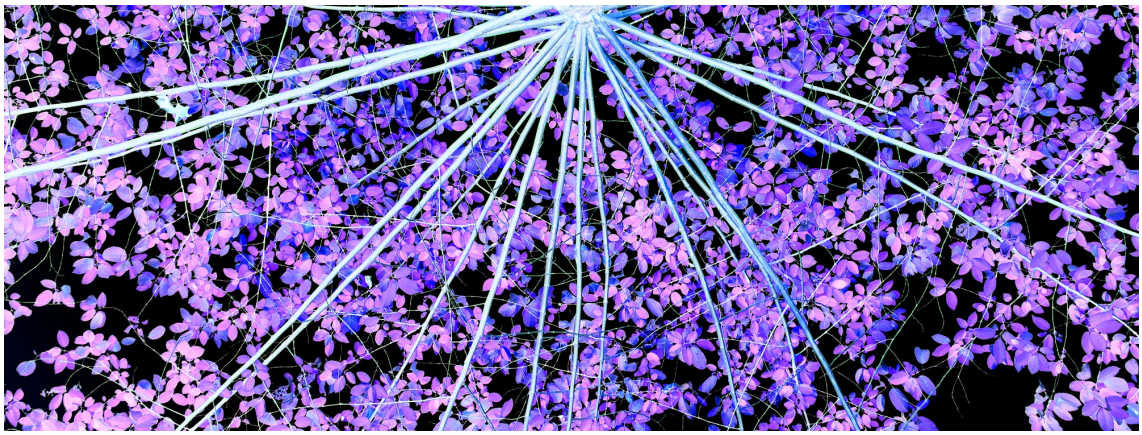
Slika 5.25: Izvorna slika

Slika 5.26: Korak 1 - odsijecanje

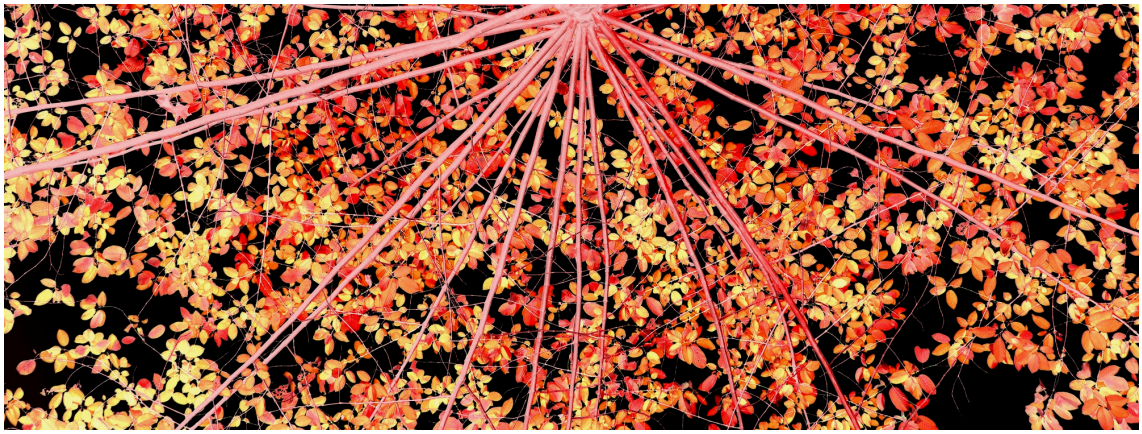
U prvom koraku (slika 5.26) odrezan je dio slike na kojemu se nalaze grane. U drugom koraku (slika 5.27) učinjeno je vertikalno zrcaljenje. U trećem koraku (slika 5.28) slika je invertirana. U četvrtom koraku (slika 5.29), pomoću alata HSV, pomaknute su boje i pojačana saturacija kako bi lišće bilo žuto-crveno. U

petom koraku (slika 5.30) slika je dodatno izoštrena kako bi se istaknuli rubovi i naglasila mrežna struktura. Slika 5.31 prikazuje usporedbu izvorne i uređene fotografije.

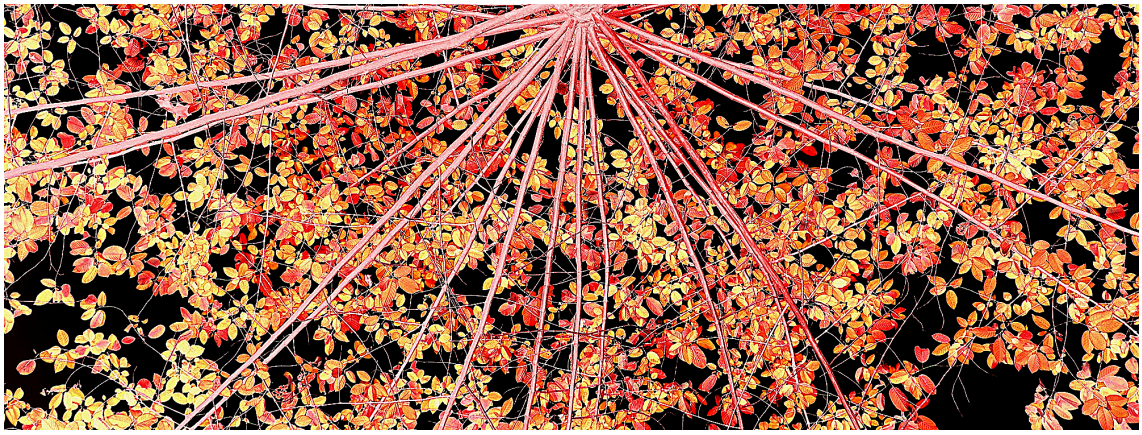
Slika 5.27: Korak 2 - vertikalno zrcaljenje



Slika 5.28: Korak 3 - invertiranje



Slika 5.29: Korak 4 - promijena boje



Slika 5.30: Korak 5 - izoštravanje

Slika 5.31: Usporedba izvorne i uređene slike

5.4. Fotografije uređene od korisnika

U ovom poglavlju prikazani su primjeri kako su korisnici iskoristili alat PhotoGrade za uređivanje fotografija. Sve osobe navedene u ovom poglavlju dale su svoju suglasnost da se njihova imena i uređene fotografije, koje su njihovo autorsko djelo, objave u ovom radu.

Fotografije su uređivali: Nevio Aradski (slike 5.32 - 5.39), Tihana Najdert (slike 5.40 - 5.45), Emanuela Răileanu (slike 5.46 - 5.47), Leona Turković (slika 5.48) i Vladimir Šeba (slike 5.49 - 5.53).

Slika 5.32: Nevio Aradski - slika 1 - izvorna slika

Slika 5.33: Nevio Aradski - slika 1 - crno-bijela verzija

Kako bi na fotografiji krajolika (slika 5.32) Nevio naglasio kompoziciju, pretvorio je fotografiju u crno-bijelu (slika 5.33).

Slika 5.34: Nevio Aradski - slika 1 - verzija s dorađenim bojama

Nadalje, na drugoj verziji fotografije (slika 5.34) uredio je boje kako bi promijenio ugođaj kojim fotografija odiše.

Slika 5.35: Nevio Aradski - slika 2 - izvorna slika

Slika 5.36: Nevio Aradski - slika 2 - hladna verzija

Slika 5.37: Nevio Aradski - slika 2 - topla verzija

Toplu fotografiju plaže (slika 5.35) Nevio je pretvorio u hladnu i snježnu fotografiju efektom invertiranja (slika 5.36). U drugoj verziji iste fotografije (slika 5.37), kako bi pojačao dojam topline, Nevio je malo povećao kontrast i saturaciju slike.

Slika 5.38: Nevio Aradski - slika 3 - izvorna slika

Slika 5.39: Nevio Aradski - slika 3 - crno-bijela verzija

Fotografija 5.38 ima snažnu kompoziciju. Kako bi gledatelja usredotočio samo na kompoziciju, Nevio je fotografiju pretvorio u crno-bijelu te doradio kontrast (slika 5.39) .

Pomoću efekta sepia, Tihana je fotografiju plaže (slika 5.40) pretvorila u užarenu pustinju (slika 5.41).

Slika 5.40: Tihana Najdert - slika 1 - izvorna slika

Slika 5.41: Tihana Najdert - slika 1 - sepia verzija

Često fotografi imaju potrebu malo doraditi fotografije dobivene iz fotoaparata pomoću upravljanja kontrastom i svjetlinom, odsijecanjem ili promjenom boja. Takve dorade Tihana i Emanuela demonstriraju na slikama 5.42 - 5.47

Slika 5.42: Tihana Najdert - slika 2 - izvorna slika

Slika 5.43: Tihana Najdert - slika 2 - uređena verzija

Slika 5.44: Tihana Najdert - slika 3 - izvorna slika

Slika 5.45: Tihana Najdert - slika 3 - uređena verzija

(a) Prije

(b) Poslije

Slika 5.46: Emanuela Răileanu - slika 1

(a) Prije

(b) Poslije

Slika 5.47: Emanuela Răileanu - slika 2

Odsijecanjem Leona je stavila fokus na bitan dio fotografije te promjenom boje i kontrasta učinila ju zabavnom i stiliziranom (slika 5.48).

(a) Prije **(b)** Poslije

Slika 5.48: Le♡na Turković - uređena slika

Fotografiju koja prikazuje muškarca kako izvodi stoj na rukama (slika 5.49), Vladimir ju je prvotno posvijetlio i alatom HSV pomaknuo boje tako da more i majica postanu zelene (slika 5.50). Zatim, u sljedećoj verziji uređene fotografije, Vladimir je napravio inverziju kako bi naglasio konturu tijela (slika 5.51). Nadalje, odrezao je fotografiju kako bi promijenio kompoziciju fotografije i u samo središte pozornosti stavio tijelo koje izvodi gimnastički pokret te primijenio sepia efekt kako bi promijenio ugođaj (slika 5.52). Naposljetku, kako bi zornije dočarao pokret koje tijelo izvodi u stoju na rukama, Vladimir je i samu fotografiju okrenuo naopačke (slika 5.53).

Slika 5.49: Vladimir Šeba - izvorna slika

Slika 5.50: Vladimir Šeba - uređena verzija 1

Slika 5.51: Vladimir Šeba - uređena verzija 2

Slika 5.52: Vladimir Šeba - uređena verzija 3

Slika 5.53: Vladimir Šeba - uređena verzija 4

6. Zaključak

U ovom radu predstavljen je postupak izrade aplikacije za uređivanje fotografija nazvane PhotoGrade te njezine funkcionalnosti i mogućnosti uporabe. Aplikacija je napisana u programskom jeziku Java uz pomoć biblioteke OpenCV korištene za manipulaciju slikovnih elemenata. Aplikacija PhotoGrade korisniku pruža intuitivan način uređivanja fotografija. Uređivanje se obavlja kroz niz alata koje aplikacija nudi. Svi alati rade po sličnom principu te korisnik na taj način brzo usvaja način rada kroz koji može preoblikovati fotografije. Također, za svaku promjenu koju korisnik napravi rezultat vidi u stvarnom vremenu. Ovo je bitan aspekt aplikacije za uređivanje fotografija koji korisniku olakšava rad te pomože pri bržem i boljem donošenju odluka. Svaka promjena s kojom je korisnik zadovoljan se sprema kako bi se u budućnosti korisnik mogao vratiti na nju i eksperimentirati s daljnjim uređivanjem

Bitan aspekt izrade aplikacije za uređivanje fotografija jest razvijanje intuitivnog grafičkog sučelja. Neočekivano, programiranje grafičkog sučelja bio je najviše problematičan aspekt razvoja aplikacije. Naime, dio programskog koda odgovornog za obradu fotografije je relativno kratak i jednostavan. Dok je s druge strane, većinski ostatak programskog koda odgovoran za grafičko sučelje i njegovo responzivno funkcioniranje u stvarnom vremenu. Također, problematičan dio razvoja bio je integriranje i korištenje različitih biblioteka koje nisu usklađene za međusobno korištenje.

Odsijecanje, zrcaljenje, rotiranje, prilagodba svjetline i kontrasta te pretvorba u crno-bijelu sliku su neki od najčešće korištenih alati pri korekciji fotografija. PhotoGrade pruža svaki od tih alata te postiže zadovoljavajuće rezultate s njima. Zamjerke idu na neintuitivan način odsijecanja pomoću klizača (engl. *slider*) umjesto odabiranja područja slike pomoću metode klikni i povuci (engl. *click and drag*) koja ocrtava pravokutnik. Također, veća fleksibilnost bila bi postignuta rotiranjem slike pod svim kutevima, umjesto samo pod pravim. Prilagodba svjetline i kontrasta u aplikaciji PhotoGrade jest fleksibilna te pruža korisniku

fine razine uglađivanja.

Alati za prilagodbu boje pružaju korisniku izražavanje kreativnosti te realiziranje, ponekad, nerealističnih fotografija. Prilagodba boje u aplikaciji PhotoGrade nije fleksibilna te je potrebno pažljivo upravljati parametrima alata ako se želi dobiti realistični rezultat. Usprkos tome, sasvim su prikladni za stvaranje maštovitih, nadrealističnih, apstraktnih te stiliziranih slika u kojima se boje ne ponašaju kao u stvarnosti.

Alat za izoštravanje primjereno služi naglašavanju detalja na fotografijama te naglašavanju rubova. Također, može poslužiti kao alat ta izoštravanje zamućenih fotografija. Zamućenje se može koristi kada želimo ugladiti oštre teksture na slikama ili potpuno zamutiti fotografiju.

Često je potrebno obraditi samo neki određeni dio slike, npr. oko na fotografiji portreta koje treba izoštriti, prilagoditi svjetlost i kontrast te promijeniti mu boju. Iako PhotoGrade pruža sve ove funkcionalnosti, nema mogućnosti obrađivanja samo određenih dijelova fotografije.

PhotoGrade, iako jednostavan, pruža korisniku niz osnovnih funkcionalnosti za obradu fotografije kojima se može postići mnogo različitih zahtjeva. Adekvatan je za blage korekcije nad fotografijama. Izvrstan je u manipulaciji svjetlinom i kontrastom te zahtjeva korisnikovu maštovitost pri korištenju alata za manipulaciju bojama.

7. Literatura

[1] Color model, Lipanj 2019. https://en.wikipedia.org/wiki/Color_model, 6. Lipanj 2019.

[2] Russ J.C., Nea, F.B. The image processing handbook. Sedmo izdanje. Boca Raton, FL, SAD. CRC Press, Inc. 2015.

[3] Color conversions, 18. Prosinac 2015. https://docs.opencv.org/3.1.0/de/d25/imgproc_color_conversions.html, 3. Lipanj 2019.

[4] Čupić M., Mihajlovi Ž. Interaktivna računalna grafika kroz primjere u OpenGL-u. Radna inačica knjige. 2018.

[5] Changing the contrast and brightness of an image!, Lipanj 2019. https://docs.opencv.org/3.4/d3/dc1/tutorial_basic_linear_transform.html, 3. Lipnja 2019.

[6] McKinley S., Levine M. Cubic spline interpolation. College of the Redwoods, 1998.

[7] Charles A. Poynton Rehabilitation of gamma, Proc. SPIE 3299, Human Vision and Electronic Imaging III, 17. Srpnja 1998.

[8] Kernel (image processing), Svibanj 2019 https://en.wikipedia.org/wiki/Kernel_(image_processing), 5. Lipanj 2019.

[9] Unsharp masking, Lipanj 2019. https://en.wikipedia.org/wiki/Unsharp_masking, 5. Lipanj 2019.

[10] Sepia (color), Ožujak 2019. https://en.wikipedia.org/wiki/Sepia_(color), 6. Lipanj 2019.

[11] Amin Ahmadi, How to Apply Sepia Filter to Images Using OpenCV, Ožujak 2016. http://amin-ahmadi.com/2016/03/24/sepia-filter-opencv/, 6. Lipanj 2019.

PhotoGrade – aplikacija za obradu digitalne fotografije

Hrvoje Ditrih

Sažetak

U radu je opisana izrada aplikacije za obradu digitalne fotografije koja je nazvana PhotoGrade. Objašnjeni su načini pohrane digitalne fotografije te osnovni prostori boja. Detaljno su opisane metode obrade fotografije koje su korištene. Aplikacija je razvijena u programskom jeziku Java uz pomoć biblioteke OpenCV. Objašnjen je način izvedbe grafičkog sučelja u okruženju Swing te su navedene problematike kod izvođenja operacija nad slikama u stvarnom vremenu. Demonstriran je rad aplikacije kroz primjere uređivanja fotografija te su priložene umjetničke fotografije koje su uredili korisnici pomoću aplikacije PhotoGrade.

Ključne riječi: PhotoGrade, digitalna obrada slike, uređivanje fotografija, Swing, modeli boja

PhotoGrade – digital image editing software

Hrvoje Ditrih

Summary

This paper describes the development of a digital image editing software named PhotoGrade. It explains how are digital images stored and describes basic color spaces. Image processing methods are explained in detail. PhotoGrade was developed in Java with the help of the OpenCV library. It demonstrates the development of a graphical user interface with the Swing framework and introduces the difficulties of executing image processing methods in real-time. PhotoGrade is demonstrated through photo editing examples and artistic photographs that the users edited with the help of PhotoGrade are attached.

Keywords: PhotoGrade, digital image processing, photo editing, Swing, color spaces