Univerzitet u Beogradu

Matematički fakultet

Neke primene teorije fazi skupova i fazi logike u procesiranju slika

- Master rad -

Nebojša Perić

1024/2013

Beograd, 2014.

Mentor: dr Aleksandar Jovanović

Matematički fakultet, Beograd

Članovi komisije: dr Miroslav Marić

Matematički fakultet, Beograd

dr Aleksandar Perović

Saobraćajni fakultet, Beograd

Datum polaganja: _____

Apstrakt

U ovom radu su predstavljene neke primene teorije fazi skupova i fazi logike u procesiranju slika. Značaj ovog rada je da se predstave potencijali ovakvog pristupa obradi i analizi podataka predstavljenih preko digitalnih slika, koji je drugačiji od češće korišćenog procesiranja slika u spacijalnom i u frekventnom domenu. Cilj ovog rada jeste da se korišćenjem tehnika zasnovanih na teoriji fazi skupova i fazi logike slika posmatra i tretira mnogo šire nego što je to jedna fotografija. Pod tim podrazumevamo da se sa slikom može operisati na različite načine, i da iz iste možemo da donosimo razne zaključke. U radu je predstavljeno nekoliko metoda, algoritama, kojima se ilustruje snaga i potencijal ovakvog pristupa procesiranju slika. Opisani su algoritmi za poboljšanje kontrasta, λ negativi , λ osvetljenje , i detekciju ivica .

(* * *)

Ovom prilikom želim da izrazim posebnu zahvalnost svom mentoru dr Aleksandru Jovanoviću na korisnim savetima, uputstvima, dobronamernoj kritici i pomoći pri odabiru literature. Isto tako želeo bih da se zahvalim porodici, kolegama sa fakulteta i članovima komisije na podršci i pomoći koju su mi pružili tokom izrade master rada.

Nebojša Perić

Beograd, 2014.

Sadržaj

Uvod

Za izgradnju i uspešnu implementaciju modernih kompleksnih sistema neophodna nam je dobra, jasna, i što preciznija reprezentacija znanja. Znanja koja poseduje čovek postaju sve važnija, može se slobodno reći da je to najvažniji ljudski resurs i kapital. Kako smo svi ograničeni u svojoj sposobnosti opažanja i rezonovanja skoro svakodnevno se suočavamo sa ***nesigurnošću*** (engl.: **uncertainty**) koja je rezultat nedostatka informacija kao što su nekompletnost informacije, leksički utisak i za nas ovde je od posebne važnosti nepreciznost merenja[1]. Drugi ograničavajući faktor u našoj želji za preciznošću jeste i prirodni jezik koji svakodnevno koristimo za opisivanje i razmenu informacija. U stanju smo da razumemo značenja reči i posedujemo sposobnost da komuniciramo precizno na zavidnom nivou, ali generalno gledano ne možemo se oko značenja i poimanja baš svake reči ili terma u potpunosti složiti. Dakle, prirodni jezici su neprecizni.

Percepcija sveta oko nas je protkana pojmovima koji nisu u potpunosti jasni, koji nemaju jasno definisane granice. Na primer, neki takvi pojmovi su: *mnogo, visok, mnogo veći od, nizak, težak,* itd.. . Upravo navedeni pojmovi su tačni samo do nekog stepena, ali su takodje i netačni do nekog stepena. Za ovakve pojmove(činjenice) se kaže da su *fazi* ili *neprecizni* (engl.: vague). Ljudski mozak je u stanju da veoma jasno i kvalitetno radi sa takvim podacima, dok računari nisu baš toliko dobri u radu sa istim. Dakle, lako možemo zaključiti da su prirodni jezici, koji su na mnogo većoj razini od programskih jezika, fazi dok programski jezici to nisu. Otuda vidimo da je neophodno koristiti jedan matematički i računarski alat kojim je moguće baratati sa fazi informacijama, koje su po svojoj prirodi jako kompleksne. Sama složenost ovakvih informacija dolazi iz nesigurnosti podataka, preciznije od višeznačnosti. Teorija koja omogućava rad sa takvim podacima je Fazi logika.

Fazi logiku je 1965. godine uveo dr Lotfi Zadeh u svom radu pod nazivom „Fuzzy sets“. i ona predstavlja matematički aparat koji omogućava rad sa ovakvim nepreciznim informacijama. Fazi logika nam obezbedjuje mehanizam za reprezentaciju jezičkih konstrukcija kao što su na primer: *mnogo, malo, sredina, često, itd.…* [1]. Uopšteno, fazi logika nam obezbedjuje strukturu koja omogućava sposobnost ljudskog rezonovanja do neke granice.

U proteklih više od tri decenije se takvi sistemi stalno usavršavaju u želji da se što više približe čoveku u pogledu donošenja odluka. Razvojem računarstva, a samim tim i razvojem sistema koji se koriste za rešavanje problema u realnom svetu, već postojeći matematički modeli nisu više bili dovoljni da se u potpunosti opišu neprecizne informacije i višeznačni pojmovi koji su bivali sve kompleksniji. Problemi iz realnog života koje je postalo moguće rešiti su postajali sve komplikovaniji, te tradicionalni sistemi za modelovanje i tehnike za analizu su postali suviše precizni, kruti. Da bi se smanjila kompleksnost takvih problema počele su da se uvode razna uprošćenja, pretpostavke i različite vrste ograničenja. Na taj način se obezbeđivao zadovoljavajući kompromis između količine informacija koje imamo sa jedne strane i njihove nesigurnosti sa druge strane. U tom aspektu za fazi logiku i fazi skupove možemo slobodno reći da su slične inženjerskim naukama, zato što daju karakterizaciju pojava i situacija iz realnog sveta na približan način.

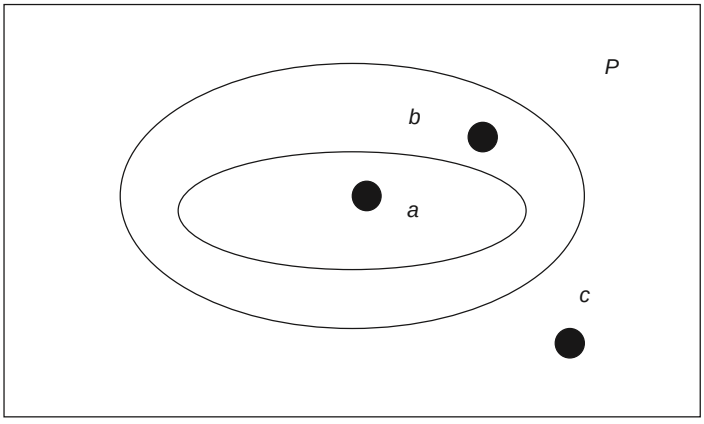
Fazi skupovi nam daju sredstva pomoću kojih modelujemo nesigurnost koja je povezana sa neodređenošću, nepreciznošću i nedostatkom informacija vezanih za dati problem[2]. Posmatrajmo sledeći primer. Razmotrimo značenje „nizak čovek“. Za osobu A, nizak čovek može biti osoba čija je visina manja ili jednaka 163 cm. Za osobu B, nizak čovek može biti osoba čija je visina manja od 166 cm. U našem primeru reč „nizak“ predstavlja lingvistički/jezički deskriptor. Kao što vidimo, term „nizak“ ima isto značenje kod osobe A i kod osobe B, ali osobe A i B nemaju isto poimanje terma „nizak“, tj. term nema jedinstvenu definiciju. Term „nizak“ bi bio efikasno shvaćen tek kada bi računar uporedio vrednosti visina osoba sa nekom unapred određenom vrednošću za pojam „nizak“. Promenljiva „nizak“ se naziva lingvistička promenljiva koja predstavlja nepreciznost u sistemu iz prethodnog primera.

Kao što smo videli nesigurnost može nastati usled nedostatka znanja ili nejasnoća u prirodnim jezicima. Uzimajući to u obzir, Lotfi Zadeh je predložio ideju pod nazivom „ the set membership “[3] , tj. ideju pripadnosti skupu pomoću koje se za svaki element vrši ispitivanje pripadnosti skupu koji predstavlja lingvističku promenljivu. Posmatrajmo ponovo primer sa visinama. Ako se za „nizak“ uzmu sve vrednosti manje ili jednake granici 165cm , tada bi osobu visine 163cm svrstali u kategoriju niskih osoba, ali bi osobu visine 166cm koja je neznatno viša, i po našem subjektivnom mišljenju jeste niska, svrstali u grupu visokih osoba. U ovom našem slučaju vrednost funkcije pripadnosti je jednaka 1 ako je testirana osoba, niska(pripada skupu niskih osoba), odnosno jednaka je 0 ako je ako testirana osoba nije niska, tj. ne pripada skupu niskih osoba. U ovom slučaju naša funkcija pripadnosti nije ništa drugo nego jedna binarna funkcija, i to karakteristična funkcija iz klasične teorije skupova:

$$\chi_A(x) = \begin{cases} 1 & , x \in A \\ 0 & , x \notin A \end{cases}$$

U fazi logici je to malo drugačije. Koncept karakteristične funkcije kakav poznajemo u klasičnim skupovima se proširuje na način da se sada meri „stepen pripadnosti“ skupu[1], što predstavlja vrednost iz intervala [0, 1] . Ono što čitaoca ne treba da zbunjuje jesu nazivi i termini karakteristična funkcija i funkcija mere pripadnosti. Funkcija pripadnosti nekom skupu u fazi logici nije ništa drugo nego karakteristična funkcija, ali zbog prirode stvari, da merimo koliko je element *pripada* nekom skupu, logično je nazivati funkciju takvim imenom, i upotreba oba termina se smatra ispravnim. Korišćenjem koncepta merenja pripadnosti skupu Zadeh je stvorio fazi skupove. Fazi skupovi se razlikuju od klasičnih skupova kod kojih je granica skupa precizna, tj. za dati element se jasno i nedvosmisleno može reči da li jeste ili nije unutar skupa. Zadeh je uopštio klasične skupove na način da je proširio skup valuacije {0,1} (jesi unutra/nisi unutra) na interval realnih brojeva [0, 1]. Možemo reći da stepen pripadnosti nekog elementa fazi skupu opisuje koliko je taj element kompatibilan, odnosno koliko odgovara pojmu/termu koji je reprezentovan fazi skupom[3]. Drugim rečima, to znači da ako je A neki fazi skup, tada on sadrži objekat x sa stepenom $\mu(x)$, gde se preslikavanje $\mu: X \to [0,1]$ naziva funkcijom pripadnosti ili karakterističnom funkcijom (engl.: membership function). Fazi skup A se sada može zapisati kao skup uređenih dvojki $A = \{(x, \mu(x)) \mid x \in X\}$. Fazi skupovi pokušavaju da opišu nejasnoće i nepreciznosti korišćenjem funkcija pripadnosti, čime se dobija na kvalitetu u pogledu reprezentacije podataka i pametnijem donošenju odluka[3]. Sada kada smo u mogućnosti da jasno opišemo nejasnoće, vidimo da granica našeg skupa nije jasno definisana kao kod klasičnih skupova. Ilustrujmo to sledećim primerom:

U- univerzum



Slika 1.1 : Granica fazi skupa

P predstavlja fazi skup, U je univerzalni skup, a , b i c su elementi. Sa slike jasno možemo videti da je a definitivno pripada fazi skupu P, da c ne pripada fazi skupu P, i da je pripadnost elementa b skupu nejasna. Dakle element a ima vrednost funkcije pripadnosti jednaku 1, vrednost za tačku c je jednaka 0, dok je vrednost za b vrednost između 0 i 1 , na primer 0.6 . Za takav element b se kaže da on delimično (engl.: partial) pripada skupu P.

U prethodnom jednostavnom primeru smo videli značaj funkcije pripadnosti. Ona preslikavanjem na interval [0, 1] opisuje skup i njegove elemente na jedinstven način. Dodeljivanjem vrednosti 0 ili 1 jasno dajemo do znanja da li je element nije ili jeste u datom skupu respektivno, dok vredonsti između 0 i 1 reprezentuju „nejasnoću“, ili češće korišćen termin fuzziness.

U praksi je ovo dosta dobra stvar zato što se na taj način može meriti koliko je nešto na slici na primer ivica, pozadina, centralni objekat i dalje koristiti za donošenje kvalitetnijih odluka, pravljenje sofisticiranijih sistema i ono što je takodje važan aspekt fazi logike jeste programiranje rečima (engl. Computing with words) [4]. Dakle fazi logikom i fazi skupovima smo u stanju da na mnogim naučnim poljima rešavamo probleme zadate rečenicama na prirodnom jeziku[5] kao što su : „Detektuj ivice na slici“ , „Segmentirati sliku na celine“, „Povećati kontrast“ … . Tačnije u stanju smo da radimo sa problemima i pojmovima koji nisu matematički jasno definisani i to na način koji je u neku ruku približan rezonovanju čoveka, samo, naravno, na jedan primitivniji način. Iako se pomoću fazi logike i fazi skupova može opisati i raditi sa ograničenim skupom problema predstavljenih rečenicama prirodnog jezika, ona predstavlja najpraktičniji način za imitaciju ljudske ekspertize na realističan način[5].

Fazi logika i fazi skupovi imaju široku primenu u mnogim ekspertskim sistemima[5], naročito u radu u industriji, biologiji, medicini i ono što nas posebno zanima u ovom radu jeste u procesiranju slika.

Osnove fazi skupova i fazi logike

Podsećanje na klasične skupove

U ovom poglavlju ćemo se podsetiti teorije klasičnih skupova. Kao što smo rekli u Uvodu fazi skupovi predstavljaju uopštenje klasičnih skupova te je korisno ponoviti osnovne operacije, zakone i svojstva klasičnih skupova. Ovde ćemo se zadržati na nivou tzv. *naivne teorije skupova*[6].

Pojam **skupa** se obično ne definiše, već se uzima kao osnovni[6][7], a često se umesto tog termina koriste razni sinonimi, kao što su, na primer, *mnoštvo, familija, kolekcija* i sl. Za označavanje skupova najčešće koristimo velika slova latinice A, B, … . Ako je neki skup konačan ili prebrojivo beskonačan, te se njegovi elementi mogu nabrojati, koristimo se zapisom[7]:

$$A=\{x_1, x_2, x_3, ..., x_n\}, \qquad \text{odnosno} \qquad A=\{x_1, x_2, x_3, ...\};$$

Takođe, elemente nekog skupa možemo opisati ako koristimo određeno svojstvo $P(x)$ koje oni (i samo oni) zadovoljavaju[6]:

$$A=\{x \mid P(x)\} \ .$$

Dakle skup je određen svojim elementima; pripadnost elementa x skupu A označava se sa $x \in A$, a nepripadnost sa $x \notin A$.

Između skupova se uvode dve osnovne relacije – inkluzija i jednakost:

$$A \subset B \Leftrightarrow (\forall x)(x \in A \Rightarrow x \in B),$$
$$A = B \Leftrightarrow (\forall x)(x \in A \Leftrightarrow x \in B).$$

Neposredno iz ovih definicija je jasno da je

$$A = B \Leftrightarrow (A \subset B \wedge B \subset A).$$

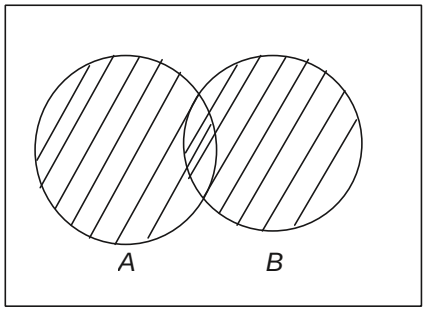
Posebno izdvajamo **prazan skup,** koji označavamo sa $\varnothing$ i možemo definisati na primer, pomoću $\varnothing = \{x \mid x \neq x\}$[6]. Taj skup ima osobinu da je $\varnothing \subset A$ za bilo koji skup A .Takođe ako su u okviru neke teorije svi skupovi sa kojima operišemo podskupovi nekog fiksiranog skupa, taj skup nazivaćemo **univerzalnim,** i često se označava sa *U*. Dakle taj skup ima osobinu da je $A \subset U$ za sve skupove *A* sa kojima operišemo u tom problemu[6].

Nad skupovima se mogu izvoditi razne operacije. Dajemo definiciju nekoliko osnovnih:

Unija

Unija dva skupa A i B, u zapisu $A \cup B$, predstavlja skup svih elemenata koji se nalaze u skupu A i svih elemenata koji se nalaze u skupu B

$$A \cup B = \{x \mid x \in A \vee x \in B\}.$$

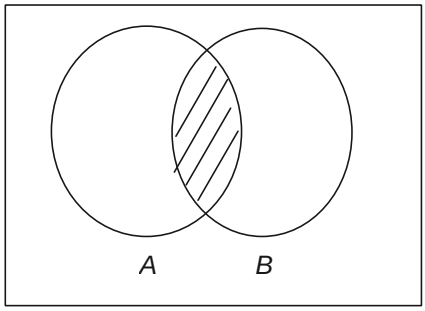


Slika 2.1 : Dijagram Unije.

Presek

Presek dva skupa A i B, u zapisu $A \cap B$, predstavlja skup elemenata koji se nalaze i u skupu A i u skupu B

$$A \cap B = \{x \mid x \in A \wedge x \in B\}.$$

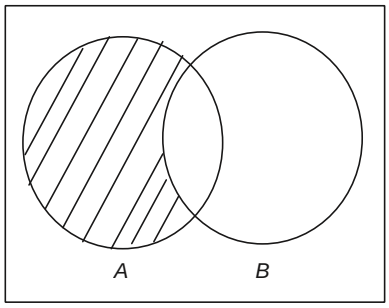


Slika 2.2 : Dijagram preseka.

Razlika

Razlika skupa A skupom B, u oznaci $A \setminus B$ je skup koji sadrži sve elemente skupa A koji nisu u skupu B

$$A \setminus B = \{x \mid x \in A \wedge x \notin B\}.$$

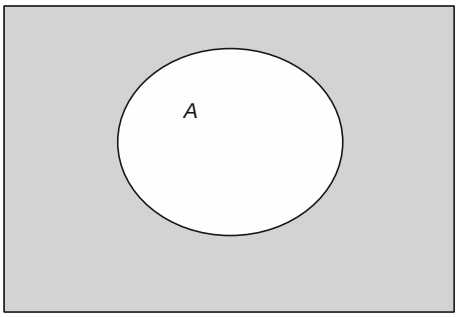


Slika 2.3 : Dijagram razlike.

Komplement

Komplement skupa A se označava sa A^c i definiše se kao kolekcija svih elemenata iz univerzalnog skupa koji nisu u skupu A

$$A^c = \{x \mid x \notin A \wedge x \in U\}.$$



Slika 2.4 : Dijagram komplementa.

Kod matematičkih operacija osobine igraju važnu ulogu. Navešćemo važne osobine :

Komutativnost

$$A \cup B = B \cup A,$$
$$A \cap B = B \cap A.$$

Asocijativnost

$$A \cup (B \cup C) = (A \cup B) \cup C,$$
$$A \cap (B \cap C) = (A \cap B) \cap C.$$

Distributivnost

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C),$$
$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C).$$

Idempotencija

$$A \cup A = A,$$
$$A \cap A = A.$$

Identitet

$$A \cup \varnothing = A,$$
$$A \cap U = A,$$
$$A \cap \varnothing = \varnothing,$$
$$A \cup U = U.$$

Tranzitivnost

$$\text{Ako } A \subseteq B \subseteq C, \text{ onda } A \subseteq C.$$

Involucija

$$A^{c^c} = A.$$

Zakon kontradikcije

$$A \cap A^c = \varnothing.$$

Zakon isključenja trećeg

$$A \cup A^c = U.$$

De Morganova pravila

$$(A \cap B)^c = A^c \cup B^c,$$
$$(A \cup B)^c = A^c \cap B^c.$$

Nad skupovima su definisana i preslikavanja, a za nas od posebnog značaja je karakteristična funkcija χ_A data pravilom :

$$\chi_A(x) = \begin{cases} 1 & , x \in A \\ 0 & , x \notin A \end{cases}$$

Karakteristična funkcija poseduje važna svojstva:

Jednakost

$$(\forall x \mid \chi_A(x) = \chi_B(x)) \Leftrightarrow A = B.$$

Komplement

$$\chi_{A^c}(x) = 1 - \chi_A(x).$$

Unija

$$\chi_{A \cup B} = \left\{ \begin{matrix} 1, & x \in A \vee x \in B \\ 0, & x \notin A \wedge x \notin B \end{matrix} \right\} = \left\{ \begin{matrix} 1, \chi_A(x) = 1 \vee \chi_B(x) = 1 \\ 0, \quad \chi_A(x) = 0 = \chi_B(x) \end{matrix} \right\} =$$

$$= \chi_A(x) \vee \chi_B(x) = \max(\chi_A(x), \chi_B(x)).$$

Presek

$$\chi_{A \cap B} = \left\{ \begin{matrix} 1, & x \in A \wedge x \in B \\ 0, & x \notin A \vee x \notin B \end{matrix} \right\} = \left\{ \begin{matrix} 1, \chi_A(x) = 1 \wedge \chi_B(x) = 1 \\ 0, \chi_A(x) = 0 \vee \chi_B(x) = 0 \end{matrix} \right\} =$$

$$= \chi_A(x) \wedge \chi_B(x) = \min(\chi_A(x), \chi_B(x)).$$

Uređenost

$$A \subseteq B \rightarrow \chi_A(x) \leq \chi_B(x).$$

Fazi skupovi

Fazi skupovi predstavljaju uopštenje klasičnih skupova. Kao što smo videli u prethodnom odeljku, klasični skupovi su dozvoljavali da neki element ili u potpunosti pripada ili u potpunosti ne pripada skupu (jesi ili nisi). Pripadnost elementa skupu je tada opisivana karakterističnom funkcijom χ_A koja vraća vrednost 1 ako je element u skupu, odnosno 0 ako element nije u skupu. Teorija fazi skupova proširuje taj koncept definisanjem *parcijalne pripadnosti*[4]. Ako imamo fazi skup *A* i neka je *U* univerzalni skup, tada je skup *A* određen karakterističnom (funkcijom pripadnosti) funkcijom $\mu_A(x)$ koja uzima realne vrednosti iz intervala [0, 1]. Logično, veće vrednosti funkcije označavaju veći stepen pripadnosti. Dakle, fazi skup je skup koji sadrži elemente sa različitim stepenima pripadnosti tom skupu. Ta ideja je u suprotnosti sa idejom pripadnosti u klasičnim skupovima, jer tamo element nije mogao pripadati skupu ako njegova pripadnost tom skupu nije potpuna. Treba naglasiti da elementi fazi skupova mogu biti elementi ne samo jednog, već i više fazi skupova od jednom. Fazi skupovi se koriste da bi predstavili jezičke varijable i modelovali probleme iz realno života kao što su: *spor, visok, brz, težak, visok, sredina, sivo, crno, ...* itd. . U ovom odeljku ćemo reći nešto o samim fazi skupovima.

Definicija 2.1 *Fazi skup A nad univerzalnim skupom U je određen svojom karakterističnom funkcijom(funkcijom pripadnosti)* $\mu_A(x): U \to [0,1]$ *, gde se za svako* $x \in U$ $\mu_A(x)$ *interpretira kao stepen pripadnosti elementa x fazi skupu A.*[1]

Ako $\mu_A(x)$ =0 tada element *x* uopšte ne pripada skupu A. Ako je $\mu_A(x)$ = 1 tada element *x* u potpunosti pripada skupu A.

Definicija 2.2 *Fazi skupovi A i B su jednaki, u oznaci A=B ako i samo ako važi* $\forall x \in U, \mu_A(x) = \mu_B(x).$

Stoga, ako je $\mu_A(x) \neq \mu_B(x)$ za neko $x \in U$, tada je $A \neq B$. Ova definicija jednakosti fazi skupova je konvencionalna. Da bismo odredili stepen jednakosti dva skupa koristimo meru jednakosti[4]:

$$E(A,B) = stepen(A = B) = \frac{|A \cap B|}{|A \cup B|}.$$

U opštem slučaju važi da je $0 \le E(A,B) \le 1.$

Definicija 2.3 *Nosač (podrška) skupa A je skup:*

$$supp(A) = \{x \in U \mid \mu_A(x) > 0\} \ .$$

Fazi skupovi se generalno obeležavaju velikim slovima latinice podvučenim znakom tilda, na primer $\underset{\sim}{A}$. Nije greška obeležavati ih i bez znaka tilda. Nadalje ćemo ih obeležavati bez znaka tilda. Fazi skup se formalno zapisuje kao skup uređenih parova[1]:

$$A = \{(x, \mu_A(x)) \mid x \in U\}.$$

Koristeći nosač skupa možemo fazi skup zapisati na sledeći način:

$$A = \frac{\mu_1}{x_1} + \cdots + \frac{\mu_n}{x_n} = \sum_{i=1}^{n} \frac{\mu_i}{x_i},$$

Znak „+“ označava uniju elemenata, a *i* označava odgovarajući stepen pripadnosti elementa *x* skupu *A*, pri čemu je $\mu_i = \mu_A(x_i) > 0.$ Ukoliko skup *U* nije konačan tada se fazi skup može predstaviti kao:

$$A = \int_U \frac{\mu_A(x)}{x}.$$

Definicija 2.4 *Jezgro skupa A je skup:*

$$ker(A) = \{x \in U \mid \mu_A(x) = 1\}.$$

Definicija 2.5 *Visina fazi skupa A je broj:*

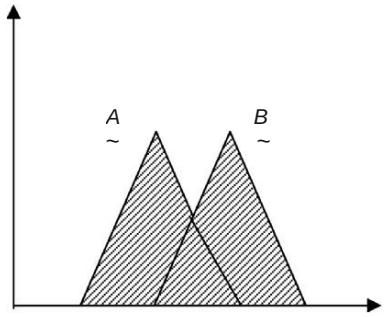
$$h(A) = sup_{x \in U} \mu_A(x).$$

Definicija 2.6 *Za fazi skup A kažemo da je normalizovan ako i samo ako važi* $\exists x \in U, \mu_A(x) = h(A) = 1.$

Nad fazi skupovima su moguće operacije:

Unija

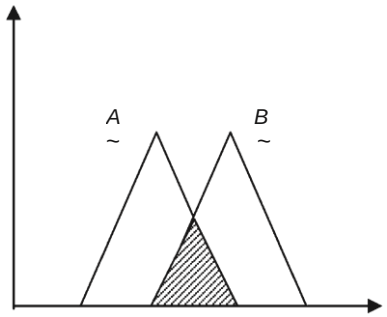
$$\mu_{A \cup B}(x) = \mu_A(x) \vee \mu_B(x) = \max(\mu_A(x), \mu_B(x)).$$



Slika 2.5: Unija dva fazi skupa A i B.

Presek

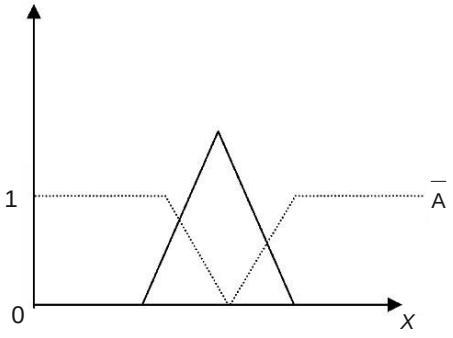
$$\mu_{A\cap B}(x) = \mu_A(x) \wedge \mu_B(x) = \min(\mu_A(x), \mu_B(x)).$$



Slika 2.6: Presek dva fazi skupa A i B.

Komplement

$$\mu_{A^C}(x) = 1 - \mu_A(x).$$



Slika 2.7 : Komplement fazi skupa A.

Takođe, kod fazi skupova svojstva igraju važnu ulogu u operacijama nad skupovima. Važnija svojstva fazi skupova su sledeća:

Komutativnost

$$A \cup B = B \cup A,$$
$$A \cap B = B \cap A.$$

Asocijativnost

$$A \cup (B \cup C) = (A \cup B) \cup C,$$
$$A \cap (B \cap C) = (A \cap B) \cap C.$$

Distributivnost

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C),$$
$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C).$$

Idempotencija

$$A \cup A = A,$$
$$A \cap A = A.$$

Identitet

$$A \cup \varnothing = A,$$
$$A \cap U = A,$$
$$A \cap \varnothing = \varnothing,$$
$$A \cup U = U.$$

Tranzitivnost

$$\text{Ako } A \subseteq B \subseteq C, \text{ onda } A \subseteq C.$$

Involucija

$$A^{cc} = A.$$

De Morganova pravila

$$(A \cap B)^c = A^c \cup B^c,$$
$$(A \cup B)^c = A^c \cap B^c.$$

Pravila koja kod fazi skupova ne važe, a kod običnih skupova važe, su pravilo kontradikcije i pravilo isključenja trećeg. Dakle kod fazi skupova važi sledeće:

$$A \cap A^c \neq \varnothing.$$

$$A \cup A^c \neq U.$$

Slično kao kod klasičnog skupa, kardinalnost se definiše kao broj elemenata sadržanih u njemu[7]. Kardinalnost, negde još nazvana i skalarna kardinalnost, fazi skupa *A* predstavlja sumu stepena pripadnosti svih elemenata *x* iz *A*, odnosno[4]:

$$|A| = \sum_{x \in U} \mu_A(x).$$

Relativna kardinalnost fazi skupa *A* definiše se kao:

$$|A|_r = \frac{|A|}{|U|}.$$

Pri čemu treba napomenuti da je $|U|$ konačna. Dakle, relativna kardinalnost određuje koji deo elemenata univerzalnog skupa *U* ima osobinu *A* kada je *U* konačan skup. Kada fazi skup *A* ima konačnu podršku, njegova kardinalnost se može predstaviti kao fazi skup. Ova fazi kardinalnost se označava sa $|A|_f$, i nju je definisao Zadeh 1978. godine kao[2]:

$$|A|_f = \sum_{\alpha \in \Lambda_A} \frac{\alpha}{|\mathrm{A}_\alpha|}.$$

Karakteristike i često korišćene karakteristične funkcije

Nejasnoće i nesigurnost u fazi skupovima se opisuju korišćenjem karakterističnih funkcija. Pomoću njih se u potpunosti opisuje svaki element skupa[1]. Karakteristične funkcije se često kod fazi skupova predstavljaju i grafički, i takva reprezentacija može uključivati više različitih oblika što nije svojstveno kod „klasične“ karakteristične funkcije. Naravno, i kod oblika karakterističnih funkcija koje koristimo postoje ograničenja i pravila. U ovom odeljku će biti više reči o oblicima i svojstvima karakterističnih funkcija.

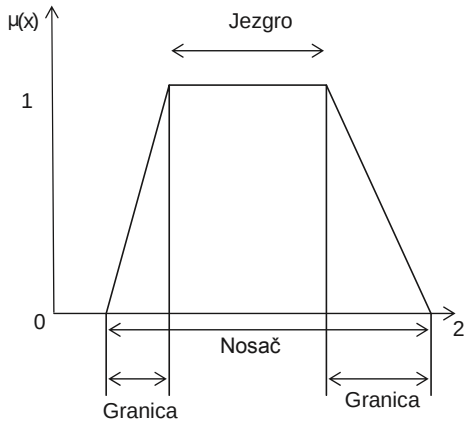
Oblik karakteristične funkcije je definisan trima svojstvima[4]:

1) *Jezgro*
2) *Nosač ili podrška*
3) *Granica*

Jezgro predstavlja deo univerzalnog skupa čiji elementi imaju vrednost karakteristične funkcije $\mu_A(x)=1.$ Dakle to su elementi koji u potpunosti pripadaju skupu *A*.

Nosač predstavlja deo univerzalnog skupa čiji elementi imaju nenula vrednost karakteristične funkcije, tj. $\mu_A(x)>0.$ To su elementi koji imaju delimičnu, parcijalnu, pripadnosti fazi skupu *A*.

Granica predstavlja deo univerzalnog skupa čiji elementi imaju parcijalnu pripadnost ali ne i potpunu pripadnost fazi skupu. Dakle deo su nosača, ali nisu deo jezgra, tj. $\mu_A(x)>0.$ i $\mu_A(x)\neq 1.$ Grafički se to reprezentuje na sledeći način:



Slika 2.8 : Jezgro, nosač i granica

Kod izgleda fazi skupa postoje dva važna termina. To su *Tačka prevoja* (engl.: Crossover point) i *Visina.* Pod tačkom prevoja funkcije pripadnosti se često podrazumeva tačka/element za koji funkcija pripadnosti uzima vrednost 0.5. Ono što ovde treba naglasiti jeste da sam korisnik u radu može definisati svoje tačke prevoja u skladu sa problematikom. Visina fazi skupa pak predstavlja maksimalnu vrednost karakteristične funkcije.

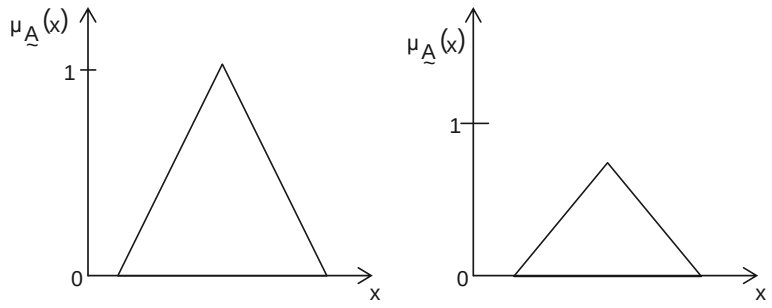
Sam izgled karakteristične funkcije može biti simetričan ili asimetričan. Na osnovu karakteristične funkcije može se vršiti klasifikacija fazi skupova:

Normalan fazi skup. Ako u fazi skupu *A* ima bar jedan element takav da je vrednost karakteristične funkcije za taj element jednaka jedinici, tada za taj fazi skup kažemo da je *normalan.*

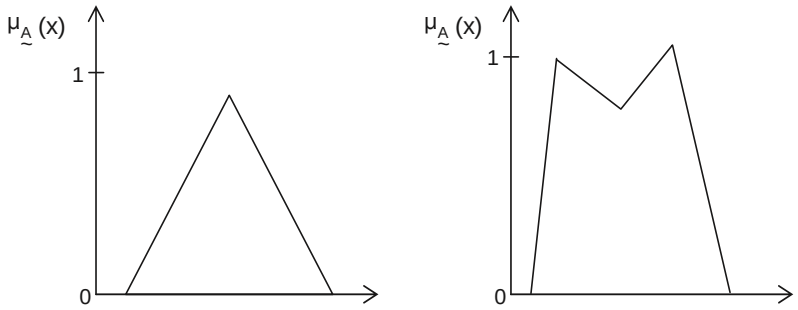
Subnormalan fazi skup. Ako u fazi skupu *A* nema ni jedan element takav da je vrednost karakteristične funkcije za taj element jednaka jedinici, tada za taj fazi skup kažemo da je *subnormalan.*

Konveksan fazi skup. Za fazi skup *A* kažemo da je konveksan ako za bilo koje $\lambda \in [0,1]$ važi:

$$\mu_A(\lambda x_1 + (1-\lambda)x_2) \geq \min(\mu_A(x_1), \mu_A(x_2)) \ .$$



Slika 2.9 : Normalan i subnormalan fazi skup.

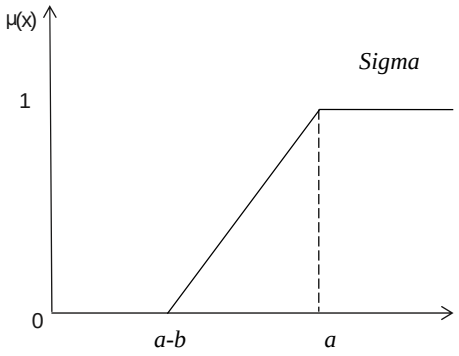


Slika 2.10: Konveksan i nekonveksan fazi skup.

U praksi, naročito kod procesiranja slika postoje karakteristične funkcije koje se često koriste. Navešćemo ovde neke od njih[8]:

Sigma:

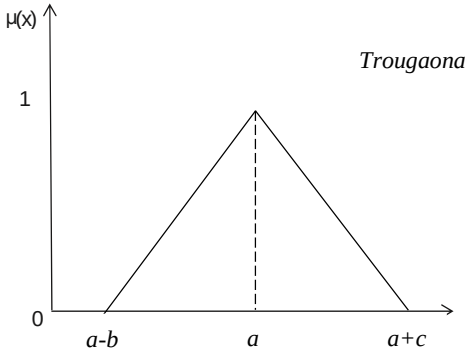
$$\mu(x)=\begin{cases} 1-(a-x)/b & a-b\le x\le a \\ 1 & x>a \\ 0 & inače \end{cases}$$



Slika 2.11 : Grafik funkcije Sigma

Trougaona:

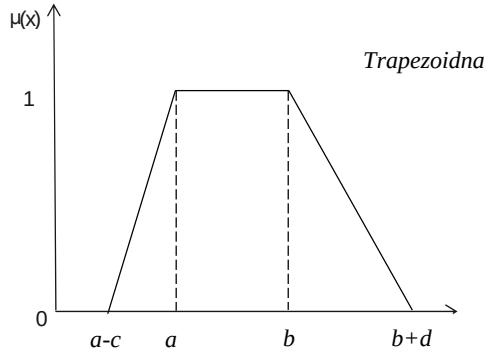
$$\mu(x)=\begin{cases} 1-(a-x)/b & a-b\le x<a \\ 1-(x-a)/c & a\le x\le a+c \\ 0 & inače \end{cases}$$



Slika 2.12 : Grafik funkcije Trougao

Trapezoidna:

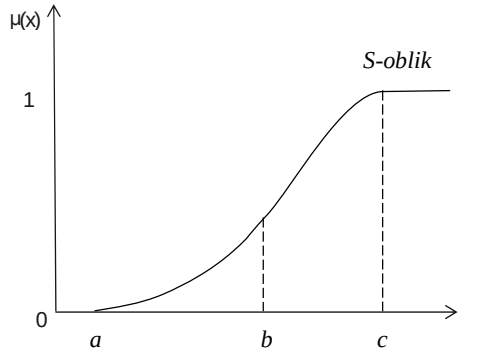
$$\mu(x)=\begin{cases}1-(a-x)/\mathrm{c} & a-c\le x<a\\ 1 & a\le x<b\\ 1-(x-b)/\mathrm{d} & b\le x\le b+d\\ 0 & inače\end{cases}$$



Slika 2.12 : Grafik funkcije Trougao

S-oblik:

$$\mu(x)=\begin{cases}0 & x<a\\ 2\left(\dfrac{x-a}{c-a}\right)^2 & a\le x\le b\\ 1-2\left(\dfrac{x-c}{c-a}\right)^2 & b<x\le c\\ 1 & x>c\end{cases}$$



Slika 2.13 : Grafik funkcije S-oblik

Fazi procesiranje slika

O procesiranju slika i istorijat

Procesiranje slika je važna i stara oblast u svetu računarstva. Koristi se kao vid poboljšanja interpretacije podataka zapisanih pomoću slika. Danas je procesiranje slika veoma rasprostranjeno i koristi se u mnogim naučnim poljima kao što su medicina, mikroskopija, molekularna biologija, vojska, industrija… .

Među autorima ne postoji opšta saglasnost šta sve spada pod procesiranje slika[9], gde počinje, a gde se završava. S obzirom na to, pod procesiranjem slika se smatra svaki proces na čijem se ulazu nalazi slika, a na izlazu se nalazi slika, ili izdvojeni atributi slike kao što su konture, prosečna osvetljenost, ivice … .

Da bismo bili u stanju da radimo sa slikama na dobar i efikasan način neophodna nam je pogodna reprezentacija slike. Za početak se prvo definiše sam oblik slike, i radi lakšeg rada se za oblik slike uzima pravougaonik[8]. Sama slika se sada definiše kao jedna funkcija ***f*** koja uzima dva argumenta $f(x, y)$[8], gde x, y predstavljaju koordinate svake tačke u slici, dok se vrednost funkcije f naziva *intenzitet* ili nivo sivog (engl**.:** gray level). Ukoliko su sve vrednosti x, y, f konačne i diskretne vrednosti, tada za takvu sliku koristimo termin *digitalna slika*[8]. Treba primetiti da se digitalna slika sastoji od konačnog broja elemenata i ti elementi se nazivaju elementi slike (engl.: *Picture elements*) ili Pikseli (engl.: *Pixels*)[8]. Dakle ukoliko imamo sliku pravougaonog oblika, onda je ona izdeljena na konačan broj piksela, gde se svaki piksel karakteriše svojom pozicijom u slici (x, y koordinatom) i vrednošću funkcije f u toj tački. Ovakvim definisanjem slike dobijamo pogodnu reprezentaciju za rad, jer se sada slika tretira i obrađuje kao matrica čiji su elementi pikseli. Kada se vrši procesiranje digitalnih slika, onda se često upotrebljava i termin *digitalno procesiranje slika* (engl.: digital image processing). [8]

Procesiranje slika, definisanih na ovaj način, se u opštem slučaju odvija na tri nivoa [10]

1) Pred procesiranje- Tu se primenjuju operacije niskog nivoa. Vrši se prilagođavanje slike uklanjanjem šumova, binarizacijom i drugim operacijama.
2) Srednje procesiranje- Bavi se morfološkim karakteristikama slike kao što su određivanje celina, krugova, trouglova … .

3) Analiza slike- Obavlja se posao vezan za veštačku inteligenciju. To je posao visokog nivoa, gde se slika prvo podešava pomoću koraka 1) i 2), a nakon toga se iz nje izvlače razni zaključci.

Jedna od prvih primena digitalnih slika je bila u novinskoj industriji 1920-tih kada su prvi put slike razmenjivane između Njujorka i Londona podvodnim kablom[8]. Tada se glavni akcenat u procesiranju slika stavljao na poboljšanje kvaliteta slika i na rekonstrukciju slika. Ono što treba reći jeste da ovo ne možemo smatrati procesiranjem digitalnih slika u pravom smislu te reči, jer se pod procesiranjem digitalnih slika podrazumeva procesiranje slika korišćenjem digitalnih računara. Prvi računari koji su bili dovoljno jaki da urade značajnije procesiranje slika su se pojavili početkom 1960-tih godina[10]. Tada se procesiranje slika koristilo i razvijalo za popravljanje kvaliteta slika koje su dolazila iz svemira(tačnije u svrhe svemirskog programa), da bi se krajem šezdesetih početkom sedamdesetih procesiranje slika počelo koristiti u medicini i astronomiji. Najznačajniji događaj u primeni procesiranja slika desio se 1970-tih, i to je bio pronalazak kompjuterizovane tomografije (CT) ili nama mnogo poznatiji kao medicinski skener[8]. Od tada pa na ovamo se primena digitalnog procesiranja slika proširila i razvila u mnogim pravcima, na bezbroj oblasti kao što su vojska, geografija … .

Kako se primena procesiranja slika proširivala na druge oblasti tako su se u istu uključivale razne matematičke teorije pomoću kojih su kreirani mnogi uspešni algoritmi u procesiranju slika. Tako je osim obrade koja se odvija isključivo na nivou piksela, tzv. rad u spacijalnom domenu, stvoreno procesiranje slika u frekventnom domenu koje se pretežno zasniva na numeričkoj matematici, tačnije na furijeovim redovima i talasićima[8][11]. Osim ova dva „standardna“ pristupa problemu procesiranja slika postoji i pristup u kome se u okviru procesiranja slika uvode elementi fazi skupova i fazi logike i takvo procesiranje slika se naziva ***fazi procesiranje slika***, o čemu će biti više reči u sledećem odeljku.

Fazi procesiranje slika

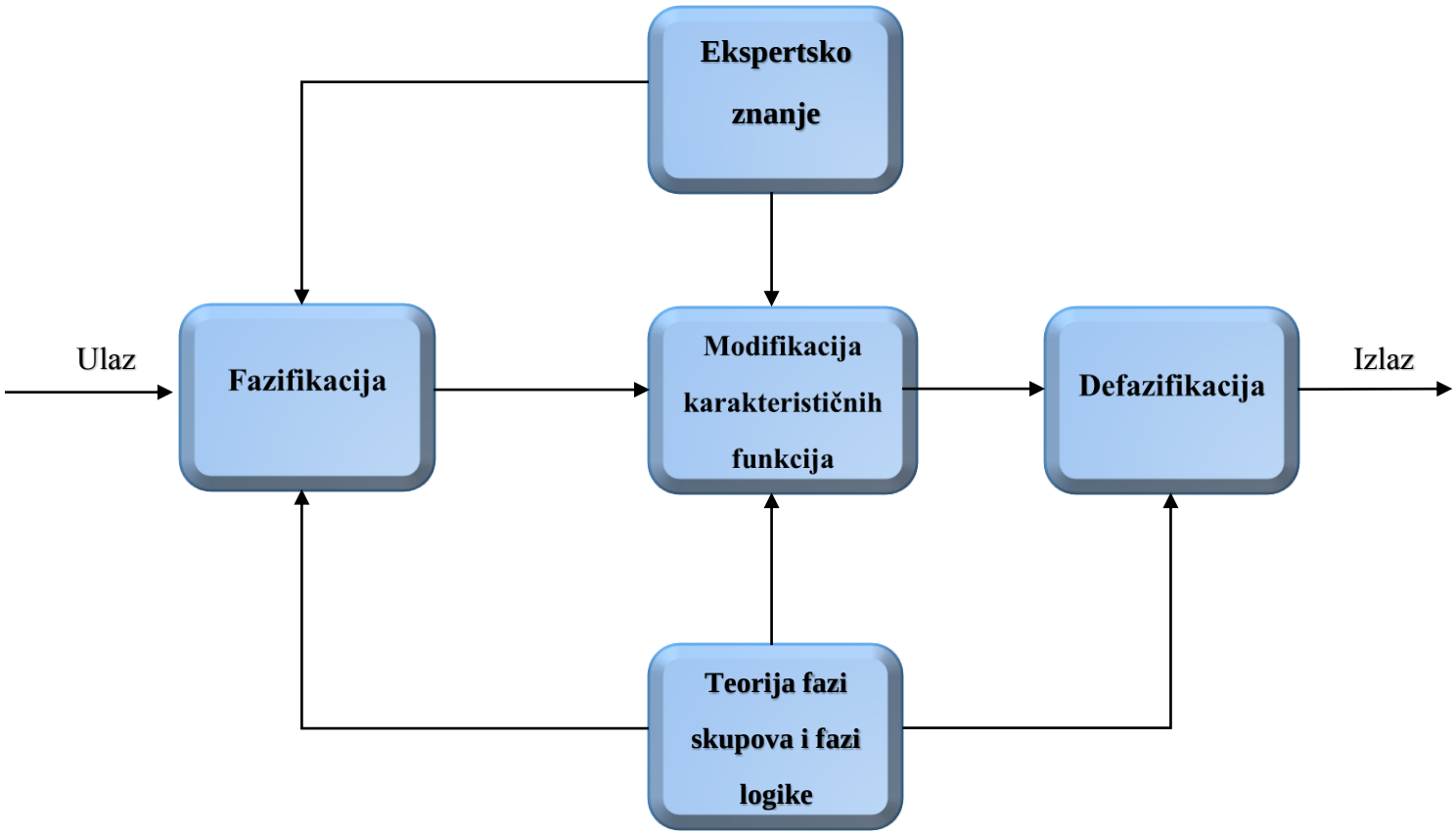
Fazi procesiranje slika se koristi u obradi slika, kada radimo sa „nejasnim“ pojmovima, odnosno kada treba izvršiti nekakvu transformaciju nad slikom koja je zadata kao jezička promenljiva, ili kada je neophodno simulirati nekakvu veštačku inteligenciju nad slikom[12][13][14]. Do sada smo već bili upoznati sa dva standardna načina procesiranja slika, u spacijalnom i frekventnom domenu, i sebi postavljamo jedno sasvim logično pitanje : Da li je zaista neophodan još jedan pristup procesiranju slika, i koje su prednosti fazi procesiranja ? Odgovor na ovo pitanje je potvrdan. Naime, fazi logika i fazi skupovi nam daju tri veoma važna razloga zbog čega ih treba upotrebiti[12]:

1) Tehnike zasnovane na fazi skupovima nam daju moćne alate za reprezentaciju znanja i procesiranje istog.
2) Fazi tehnike su u stanju da veoma dobro barataju sa neodređenim, nepreciznim i višeznačnim informacijama.
3) U mnogim primenama procesiranja slika neophodno nam je ekspertsko znanje da bi prevazišli poteškoće.

Korišćenjem prethodno navedenih prednosti možemo slike i/ili njihove delove i svojstva tretirati kao elemente teorije fazi skupova i fazi logike. Ono što treba naglasiti jeste da fazi procesiranje slika nije jedinstvena teorija, već predstavlja kolekciju raznih različitih fazi pristupa u procesiranju slika. Fazi procesiranje slika se definiše na sledeći način[13]:

Definicija 3.1: *Fazi procesiranje slika predstavlja skup svih pristupa koji razumeju, predstavljaju i procesiraju slike, njihove segmente i svojstva kao fazi skupove. Njihova reprezentacija i procesiranje zavise od odabranih fazi tehnika i problema kog treba rešiti.*

Fazi procesiranje slika se, u opštem slučaju, sastoji iz tri glavne faze i dve „pomoćne“ faze, a to su : *Fazifikacija, Modifikacija karakterističnih funkcija, Defazifikacija ,* kao glavne i *Teorija fazi skupova i fazi logike* **i** *Ekspertskog znanja* kao pomoćne[8][13]. Grafički se odnos između ovih faza može predstaviti na sledeći način:



Slika 3.1:Dijagram Fazi procesiranja slika

Dakle na ulazu i izlazu fazi procesiranja slika je, naravno, slika. Prva faza je proces fazifikacije. ***Fazifikacija*** je proces u kome se vrši transformacija slike sa ulaza u fazi domen[13]. Pod fazi domenom se podrazumeva *fazi skup* ili *fazi if-then* pravila, odnosno ulaz se transformiše na takav način da je na njega moguće primeniti operacije i svojstva iz fazi logike i fazi skupova. Najčešće se cela slika koja je izdeljena na piksele tretira kao fazi skup gde su elementi tog skupa pikseli sa odgovarajućom vrednošću karakteristične funkcije. ***Modifikacija karakterističnih funkcija i njihovih vrednosti*** je proces u kome se vrši promena vrednosti karakterističnih funkcija za svaki element dobijenog fazi skupa[13]. Ta modifikacija može biti i veoma prosta, dakle najobičnije množenje, deljenje, sabiranje i oduzimanje, a sa druge strane može uključivati i mnogo složenije operacije kao na primer kompozicije različitih funkcija, integracija korišćenjem fazi integrala i slično. U nekim situacijama se ta faza može preskočiti. Poslednja faza je proces ***Defazifikacije.*** Defazifikacija je proces u kome se vrši povratak iz fazi domena u domen slike[13]. U fazi domenu smo za svaki piksel iz slike sračunali odgovarajuću vrednost karakteristične funkcije i na osnovu te vrednosti vršimo povratak i računamo vrednost intenziteta za svaki piksel. Dakle na kraju tog procesa smo generisali izlaznu sliku. Postoji

nekoliko načina za implementaciju defazifikacije, od osnovnih kao što su preslikavanje iz intervala [0, 1] na interval [0, L-1], gde L predstavlja broj različitih vrednosti intenziteta u pikselima pa sve do složenijih kao što su *Centar sume, Centar mase, Centroidni metod, princip maksimalne pripadnosti i drugi* [15]. Za ove tri faze kažemo da su glavne, iz prostog razloga što su to koraci koje je neophodno preuzeti u postizanju željenog cilja. Osim upravo opisanih faza postoje dve faze za koje kažemo da su pomoćne faze. U tim fazama nema transformacije slike kao u glavnim fazama, već te pomoćne faze predstavljaju baze znanja koje se koriste u glavnim fazama. Postoje dve pomoćne faze. ***Teorija fazi skupova i fazi logike*** je faza koja se koristi u sve tri glavne faze, i ona predstavlja, kao što joj i samo ime kaže, znanje fazi logike i/ili fazi skupova neophodno za rešavanje problema. Druga pomoćna faza, koja se koristi u koracima fazifikacije i modifikacije, je ***Ekspertsko znanje.*** Pod time podrazumevamo „ne-fazi" znanje, odnosno znanje iz stručne oblasti koje nam je neophodno da bismo uspešno rešili dati problem. Jedan dobar primer koji bi ilustrovao značaj ove faze jeste zadatak „*detektovati ivice i oblasti na rendgenskom snimku*". Kada se nađemo u ovakvoj situaciji onda je neophodno i znanje iz konkretne oblasti gde vršimo procesiranje (u ovom slučaju medicine, konkretno radiologije) i znanja iz procesiranja slika (u ovom slučaju da se detekcija ivica zasniva na diferenciranju).

Ono što treba naglasiti jeste da kao i svaki pristup rešavanju nekog problema tako i fazi pristup procesiranju slika ima svoje prednosti i mane. Glavni nedostatak ovakvog pristupa procesiranju slika jeste da smo u mnogim situacijama ograničeni da radimo sa crno-belim(preciznije sivim) slikama[14]. Kažemo mana, zato što bismo mi želeli da spektar primene algoritama bude što veći, uzimajući u obzir da smo u realnom životu okruženi slikama u boji. U praksi i u konkretnim primenama kao što su biologija, medicina i mikroskopija to nije ograničavajući faktor. Prednost ovakvog pristupa jeste da možemo na jedan kvalitetan način obraditi slike i preciznije raditi na veoma osetljivim problemima kao što su segmentacija slike, klasifikacija ivica i drugi. Osim ovoga prednost fazi procesiranja slika jeste da mogu da inkorporiraju i druge matematičke i računarske pristupe, kao što su neuronske mreže, genetske algoritme, zaključivanje, induktivno rezonovanje, pa i intuiciju[14]. U sledećem odeljku ćemo prikazati algoritme za poboljšanje kontrasta, detekciju ivica, lambda negative, lambda osvetljenje, kao i primenu istih na standardnim test slikama.

Algoritmi

Algoritmi za isticanje kontrasta

Svaka slika sadrži nekoliko bitnih svojstava koja su važna za sam doživljaj slike. Jedno od tih svojstava jeste *kontrast*. Kontrast predstavlja odnos između svetlih i tamnih delova u slici[8][11]. Ako je velika razlika u vrednostima intenziteta u pikselima u slici onda za takvu sliku kažemo da ima veliki kontrast, a ako se sve vrednosti nalaze u uskoj zoni, na primer uzimaju vrednosti između 30 i 70 (na skali od 0 do 255), tada za kontrast u toj slici kažemo da je mali. Kada je slika malog kontrasta dolazi do gubitka detalja i poželjno je da se slika malo razvuče. Takvim transformacijama se postiže bolja percepcija slika od strane posmatrača, odnosno kvalitetnije opažanje detalja. U ovom odeljku ćemo predstaviti dva algoritma koja vrše isticanje kontrasta. Jedan je takozvani INT algoritam[14][15] koji u potpunosti ilustruje rad sa slikama kao sa fazi skupom, dok je drugi algoritam zasnovan na fazi if-then pravilima, dakle više se oslanja na kreiranje pravila i na pravila zaključivanja (modus ponens).

INT algoritam

INT algoritam, ili algoritam intenziteta, je algoritam koji spada u grupu fazi algoritama i koristi se za popravljanje slike isticanjem kontrasta. Glavna odlika ovog algoritma je da koristi dve karakteristične funkcije i da se te dve karakteristične funkcije primenjuju direktno na svaki pojedinačni piksel u slici. Ulaz našeg algoritma je slika, neka je se zove I, dimenzije $M \times N$, i neka je intenzitet svakog piksela celobrojna vrednost iz intervala [0, L-1] . Na samom početku se kreira jedan fazi skup, čiji su elementi pikseli, koji predstavlja osvetljenost, odnosno koliko je svaki piksel svetao. Proces Fazifikacije se odvija tako što se za svaki piksel izračuna njegova mera svetloga korišćenjem formule

$$\mu(x_{mn}) = \frac{x_{mn}}{L-1}.$$

Nakon što smo napravili fazi skup prelazimo na fazu modifikacije. Tu se primenjuje druga karakteristična funkcija koja će poboljšati kontrast. Ideja je da oni pikseli čija karakteristična funkcija μ ima vrednost manju od 0.5 „sabijemo“ bliže nuli, a da one veće od 0,5 „sabijemo“ bliže jedinici, dakle da ih potamnimo i posvetlimo respektivno. Za rešenje tog problema koristimo sledeću karakterističnu funkciju[14][15]:

$$\mu_{INT}(x_{mn}) = \begin{cases} 2\left[\mu(x_{mn})\right]^2, & 0 \le \mu(x_{mn}) \le 0.5 \\ 1 - 2(1-\mu(x_{mn}))^2, & 0.5 < \mu(x_{mn}) \le 1 \end{cases}$$

Sada je neophodno vratiti se iz fazi domena u domen slike. Dakle treba na osnovu vrednosti funkcije μ_{INT} upisati vrednosti intenziteta za svaki piksel. Dovoljno je primeniti formulu:

$$x_{mn} = round((L-1) \bullet \mu_{INT}(x_{mn})).$$

Ovom formulom se u piksel na poziciji (*m n*) u slici postavlja intenzitet na način da bude srazmeran dobijenoj vrednosti karakteristične funkcije. Zaokruživanje se vrši iz prostog razloga što su intenziteti celobrojne vrednosti, a karakteristične funkcije uzimaju realne vrednosti iz intervala [0, 1].

Što se tiče konkretne implementacije, treba primetiti sledeće: za obradu svakog piksela dovoljan nam je samo taj trenutni piksel sa kojim radimo i ni jedan više. Dakle dovoljno je u jednom prolazu kroz piksele slike primeniti ove operacije na svaki piksel. Pseudo kod algoritma dat je na sledeći način:

```
Algoritam INT(I)
Ulaz: Slika I dimenzije M × N, sa najviše L vrednosti
intenziteta
Izlaz: Modifikovana slika I
begin
     for i:=0 to M do
          for j:=0 to N do
               mi:= I[i, j]/(L-1);
               if mi ≤ 0.5 then
```

```
                    mi:= 2*mi*mi;
               else
                    mi:= 1 – 2*(1-mi)*(1-mi);
               intenzitet:= round( mi*(L-1));
               I[i, j]:=intenzitet;
return I;
end
```

Slika 4.1: Algoritam INT za isticanje kontrasta.

Napomena!: Ovaj algoritam ima jedno ograničenje, a to je da radi samo sa crno belim (sivim) slikama. Pod I[x, y] se podrazumeva vrednost intenziteta piksela na poziciji (x,y) u slici

Primena ovog algoritma, a tako će biti i sa ostalima, će biti prikazana na standardizovanim test slikama. Sa leve strane će biti prikazane originalne slike, a sa desne će biti prikazane slike dobijene ovim algoritmom. U sklopu testiranja za L je uzeta vrednost 256. Na slici 4.2 je prikazana originalna slika, dok je na slici 4.3 prikazana ista ta slika nakon primene algoritma INT:

Slika 4.2: Originalna slika.

Slika 4.3: Nakon primene algoritma INT.

Nekada nam je neophodna višestruka primena INT algoritma da bismo stigli do željenog rezultata. Na slici 4.4 je prikazana originalna slika, a na slikama 4.5 , 4.6 i 4.7 je prikazana jednostruka, dvostruka i trostruka primena INT algoritma respektivno:

Slika 4.4 :Originalna slika.

Slika 4.5 : Nakon primene algoritma INT.

Slika 4.6 :Nakon dve primene algoritma INT.

Slika 4.7: Nakon tri primene algoritma INT.

Algoritam za isticanje kontrasta korišćenjem pravila

Sada ćemo prikazati još jedan algoritam za isticanje kontrasta. Ovaj algoritam je drugačiji, jer se on više zasniva na isticanju kontrasta korišćenjem pravila. Ideja ovog pristupa jeste da možemo da programiramo rečima. Naime fazi logika i fazi skupovi su, kao što smo to već rekli, koriste da opišemo, i preciznije baratamo, sa pojmovima iz svakodnevnog života koji nisu precizni kao na primer *sivo, visok, svetao, ...* . Suština isticanja kontrasta jeste da *svetli* pikseli postaju *svetliji, tamni* da postaju *tamniji,* a oni *sivi* da postanu/ostanu *sivi.* Dakle algoritam za isticanje kontrasta se sastoji iz tri pravila[8][11][14]:

```
IF a pixel is dark, THEN make it darker.
IF a pixel is gray, THEN make it gray.
IF a pixel is bright, THEN make it brighter.
```

Ono što treba primetiti, jeste da su *dark, darker, gray, bright, brighter* zapravo fazi termi, i da njih možemo da izrazimo kao fazi skupove, odnosno preciznije kao karakteristične funkcije. Dakle neophodno je da za svaki od navedenih termova definišemo karakterističnu funkciju (može biti i konstantna vrednost) i da definišemo način kako da vršimo zaključivanje, zapravo kako simulirati *modus ponens* . Kako ćemo definisati karakteristične funkcije za termove je više stvar intuicije. Pretpostavimo da radimo sa ***sivim slikama***, pri čemu siva ima najviše 256 različitih nijansi, tj. vrednost L je 256. Tako za *bright* definišemo karakterističnu funkciju po šablonu da ako je vrednost intenziteta piksela veća od 165 onda je vrednost karakteristične funkcije jednaka 1, ako je vrednost manja od 127 onda je vrednost karakteristične funkcije jednaka 0, a kao je vrednost intenziteta između 127 i 165 onda se vrednost karakteristične funkcije *bright* računa kao $\mu(z) = 1 - \frac{(165 - z)}{38.25}$. Dakle :

$$\mu_{bright}(z) = \begin{cases} 0 & z < 127 \\ 1 - \dfrac{(165 - z)}{38.25} & 127 \le z \le 165 \\ 1 & z > 165 \end{cases}$$

Na sličan način se definišu i *dark i gray.* Za karakterističnu funkciju dark ćemo olakšati posao pošto je osobina tamno suprotna od osobine svetlo, te ćemo iskoristiti svojstvo komplementa karakteristične funkcije i definisati kao:

$$\mu_{dark}(z) = 1 - \mu_{bright}(z).$$

Funkciju *gray* ćemo opisati na taj način da na sredini (vrednost 127) karakteristična funkcija uzima vrednost 1, a kako se krećemo bilo levo bilo desno ta vrednost opada. Dakle definišemo karakterističnu funkciju gray kao :

$$\mu_{gray}(z) = \begin{cases} 0 & z < 89 \\ 1 - \dfrac{127 - z}{38.25} & 89 \le z < 127 \\ 1 - \dfrac{z - 127}{38.25} & 127 \le z < 165 \\ 0 & z > 165 \end{cases}.$$

Termove *darker, brighter, gray,* dakle termove iz desnih strana pravila ćemo, ilustracije radi, opisati kao singltone, odnosno kao konstantne vrednosti. Singlton za *darker neka ima vrednost* 25,5 (tačnije 0,1 * (L-1), gde L kod nas uzima vrednost 256), za brighter neka ima vrednost 229,5 (tačnije 0,9*(L-1)), i na kraju singlton gray neka ima vrednost 127. Umesto singltona se mogu koristiti funkcije jedino treba paziti da li su funkcije korektno definisane.

Sada kada imamo karakteristične funkcije, treba izvršiti zaključivanje, odnosno moramo imati alat koji ume da rastumači ulaz i na osnovu njega da odredi koje je pravilo zadovoljeno, te da izvrši to pravilo. Ovde nam pomaže proces defazifikacije. U procesu defazifikacije koristimo ponderisane sume, tj. koristimo formulu[14]:

$$z_{mn} = \frac{\mu_{dark}(z_{mn}) \times v_d + \mu_{bright}(z_{mn}) \times v_b + \mu_{gray}(z_{mn}) \times v_g}{\mu_{dark}(z_{mn}) + \mu_{bright}(z_{mn}) + \mu_{gray}(z_{mn})} \qquad (4.1)$$

Gde su v_d, v_b, v_g singltoni za *darker, brighter i gray* respektivno. Korišćenjem ove formule se na jedan uspešan način vrši zaključivanje. Za svaki piksel u slici se računaju vrednosti sve tri karakteristične funkcije, zatim se ovom formulom zaključuje o kom je pravilu reč i vrši sprovođenje istog. Treba naglasiti da rezultat dobijen ovom formulom treba

modifikovati, odnosno treba zaokružiti na najbližu celobrojnu vrednost. Algoritam je opisan sledećim pseudo kodom:

```
Algoritam Rule(I)
Ulaz: Slika I dimenzije M × N, sa najviše L vrednosti intenziteta
Izlaz: Modifikovana slika I
begin
     for i:=0 to M do
          for j:=0 to N do
               I[i, j]:= WAM(I[i, j]);
return I;
end
```

Slika 4.8: Algoritam za isticanje kontrasta korišćenjem pravila.

```
Funkcija WAM(z)
Ulaz: vrednost intenziteta piksela z
Izlaz: modifikovana vrednost intenziteta piksela z
begin
singlton_darker:=0.1*(L-1);
singlton_brighter:=0.9*(L-1);
singlton_gray:=0.5*(L-1);
z= round( ( mi_gray(z)* singlton_gray + mi_darker(z) *
singlton_darker + mi_brighter(z) * singlton_brighter
)/(mi_gray(z)+ mi_darker(z)+ mi_brighter(z) ) )
return z;
end
```

Slika 4.9: Funkcija WAM za računanje formule 4.1 .

Sada ćemo pokazati primenu algoritma za isticanje kontrasta zasnovanog na pravilima nad slikama 4.10 i 4.12. Rezultat primene algoritma nad slikom 4.10 je prikazan na slici 4.11, dok je za sliku 4.12 rezultat primene prikazan na slici 4.13:

Slika 4.10:Originalna slika.

Slika 4.11: Nakon primene algoritma Rule.

Slika 4.12: Originalna slika.

Slika 4.13: Nakon primene algoritma Rule.

Algoritam λ osvetljenja

U prethodnom delu smo videli dva algoritma za poboljšanje jedne od dve ključne osobine slike, a to je kontrast. Druga ključna osobina slike jeste ***osvetljenost.*** To je osobina od koje zavisi kako posmatrač vidi sliku, odnosno kako se detalji vide u slici, te je nekada važno popraviti osvetljenost. Nekada nije dobro popravljati osvetljenost piksela linearno, tj. povećati ili smanjiti intenzitete svih piksela za neku konstantnu vrednost jer onda dolazi do gubitka crnih ili belih piksela. Jedno rešenje ovog problema nam nudi fazi pristup rešavanju ovog problema. Na početku se za svaki piksel izračuna karakteristična funkcija osvetljenosti tj. :

$$\mu(x_{mn}) = \frac{x_{mn}}{L-1}.$$

Nakon toga korisnik treba sam da odredi u kom smeru želi da unapredi osvetljenje, da li da ga poveća ili smanji, i koliko želi da ga unapredi. Da bismo to uspeli neophodno je na početku definisati parametar **λ** gde je $-1 < \lambda < \infty$. Kada odaberemo parametar vršimo modifikaciju karakteristične funkcije preko formule[13]:

$$\mu(x) = \frac{(1+\lambda)\bullet\mu(x)}{1+\lambda\bullet\mu(x)}.$$

Gde *x* predstavlja piksel u slici koju obrađujemo. Na kraju se treba vratiti iz fazi domena u domen slike, i za to se koristi formula:

$$x = round((L-1)\bullet\mu(x)).$$

Primena ovog algoritma, za različite vrednosti λ, je prikazana na slikama 4.14, 4.15, 4.16:

Slika 4.14: Originalna slika.

Slika 4.15: Primena sa λ= 1.

Slika 4.16 :Primena sa λ= -0.5

Algoritam λ negativa

Jedna od osnovnih transformacija u klasičnom procesiranju slika jeste transformacija *negativa*. Negativ je transformacija u kojoj se intenzitet svakog piksela zameni njegovom dopunom do L-1, gde je L predstavlja ukupan broj nijansi sivog. Formalno se to zapisuje kao[8][11]:

$$I[x, y] = L - 1 - I[x, y].$$

U nekim situacijama ovakav negativ kakav poznajemo nije dovoljan. Tipičan primer jesu skener(CT) i magnetna rezonanca kada se usled lošeg pozicioniranja pacijenta i/ili nedovoljno dobrog kvaliteta i rukovođenja aparatom dobijemo sliku koja nije dovoljno dobra da se na njoj mogu videti detalji, ili koja je čak u stanju da dijagnostičku pretragu pošalje u pogrešnom pravcu. Glavna ideja ovog algoritma jeste da se napravi uopštenje klasičnog negativa, gde bismo umesto jednog snimka mogli da dobijemo beskonačno mnogo nijansi negativa. Algoritam koji nam to obezbeđuje je algoritam iz grupacije fazi algoritama, a to je algoritam λ negativa[15]. Na početku se napravi karakteristična funkcija koja daje meru osvetljenosti svakog piksela u slici I, dimenzije $M \times N$, sa L nivoa sivog :

$$\mu(x_{mn}) = \frac{x_{mn}}{L-1}.$$

Nakon toga korisnik bira parametar λ koji je ključan, tj. on određuje koji, odnosno kakav će biti izlazni negativ. Parametar λ je realan broj koji uzima vredonsti iz intervala (-1, ∞) . Sledeći korak je da se izvrši modifikacija karakteristične funkcije svakog piksela korišćenjem formule:

$$\mu(x) = \frac{1-\mu(x)}{1+\lambda \cdot \mu(x)}.$$

Na samom kraju je neophodno da se vratimo u domen slike, i da u svaki piksel upišemo odgovarajući intenzitet. To radimo korišćenjem formule:

$$x = round((L-1) \bullet \mu(x)).$$

Treba primetiti da kada je parametar λ jednak nuli, da je to onda klasičan negativ. Algoritam se može uopštiti da radi i za slike u boji, gde se ovaj postupak primenjuje na crvenu, zelenu i plavu komponentu RGB slike. Primena algoritma λ negativa je prikazana na sledećim slikama. Slika 4.17 je originalna slika, a slike 4.18, 4.19 i 4.20 su slike dobijene primenom ovog algoritma za različite vrednosti parametra λ.

Slika 4.17: Originalna slika.

Slika 4.18: λ negativ sa λ=0.

Slika 4.19: λ negativ sa λ = -0.5

Slika 4.20: λ negativ sa λ = =1.9.

Detekcija ivica

Detekcija ivica je zadatak od velikog značaja u procesiranju slika. Koristi se kao pred procesna radnja u mnogim kompleksnijim algoritmima kao što je na primer segmentacija slike na regione , a može se koristiti i u kompresiji slika. Prednost koju nam daje detekcija ivica jeste da odbacivanjem boja i nepotrebnih podataka štedimo na utrošenom prostoru, a da pri tome čuvamo strukturne osobine objekata u slici[16].

Šta su zapravo ivice ? Ivice su mesta u slici gde se intenzitet naglo menja[17]. Gde je razlika vrednosti susednih piksela jako velika. To su na primer mesta kada prelazimo iz jedne oblasti u drugu oblast, sa belog na crno i slično. Formalno, kaže se da je Ivica *skup povezanih ivičnih piksela* (engl.: Edge pixels)[17], a ivični pikseli su zapravo ta „mesta“ gde dolazi do velike promene u intenzitetu. Do sada je razvijeno mnogo algoritama namenjenih za detekciju ivica koji rade u spacijalnom domenu, kao što su Sobelov , Previtov, Laplasijan, Kanijev i mnogi drugi. Zbog čega se i dalje kreiraju novi i novi algoritmi za detekciju ivica leži u razlogu da ni jedan od dosadašnjih algoritama „nije savršen“. Pod tim podrazumevamo da ni jedan od njih nije opšte namene, preciznije: da nismo u mogućnosti da uvek jedan te isti algoritam koristimo kada god se od nas zahteva detekcija ivica. Na primer, Sobelov algoritam je dobar kada je neophodno izvršiti detekciju oblika, ali nije dobar kada je neophodno vršiti detekciju ivica u realnom vremenu gde je veliki akcenat stavljen na brzinu (direktan prenos nekog sportskog događaja i računanje različitih statistika) i tada se koristi Kanijev algoritam[18]. Shodno ovome, postoji potreba da se nađe drugačiji, a možda u nekim segmentima i bolji algoritam za rešavanje ovog problema.

Jedan od načina kako da se vrši detekcija jeste da se koriste fazi logika i fazi skupovi. Naš pristup problemu je sledeći: Kreiramo fazi skup, gde su elementi uređeni parovi (piksel, vrednost karakteristične funkcije). Taj fazi skup predstavlja skup ivica, a karakterističnom funkcijom merimo koliko svaki piksel pripada tom skupu, te shodno tome u svaki piksel postavljamo vrednost intenziteta u skladu sa vrednošću karakteristične funkcije. Prema prethodnom, ako je I slika u kojoj želimo da detektujemo ivice sa L nivoa sivog, onda je naša izlazna slika oblika[2][16]:

$$Edge(I) = (L-1)\cdot \bigcup_m \bigcup_n \frac{\mu_{mn}}{I_{mn}}$$

Ključna stvar za određivanje ivica u ovom algoritmu jeste karakteristična funkcija. Da bismo napravili dobru karakterističnu funkciju neophodna su nam dodatna znanja. Tu na scenu stupaju pomoćne faze u fazi procesiranju slika : Ekspertsko znanje i Teorija fazi skupova i fazi logika. Iz ekspertskog znanja treba da uzmemo u obzir da *je ivica mesto gde se intenzitet naglo menja*, a iz teorije fazi skupova i fazi logike kako da to uskladimo i dobijemo dobro definisanu funkciju. Dakle treba uzeti u obzir razliku intenziteta piksela kog trenutno obradjujemo sa pikselima iz njegovog neposrednog susedstva, i da napravimo funkciju takvu da kada je ta razlika velika onda ona teži jedinici, a kada je razlika mala da ona teži nuli. Jedna funkcija koja to zadovoljava jeste:

$$\mu_{edge}(I(x,y)) = 1 - \frac{1}{1 + \frac{\sum_N |I(x,y) - I(i,j)|}{\Delta}} \quad (4.2)$$

I je ulazna slika. *N* su susedni pikseli pikselu (*x, y*) u slici, a Δ je konstanta koju bira korisnik. Za Δ se uzima vrednost L-1 ili maksimalni intenzitet sivog iz okoline trenutnog piksela. Sada kada imamo dobro definisanu karakterističnu funkciju definisaćemo sam algoritam. Algoritam se sastoji iz sledećih koraka:

1) Konverzija slike u boji u sivu sliku. Kod slika u boji boja je predstavljena preko tri komponente , *crvene, zelene i plave* tj. u RGB kolor sistemu. Na osnovu te tri komponente može se generisati siva slika tako što se na svaki piksel u slici primeni formula:

$$I(i,j) = \sqrt[2,2]{0,2126 \times I(i,j).red^{2,2} + 0,7152 \times I(i,j).blue^{2.2} + 0,0722 \times I(i,j).green^{2,2}}$$

Kako radi konverzija može se videti na slikama 4.21 i 4.22:

Slike 4.21 , 4.22 : Test slike za konverziju RGB u Gray.

2) *Fazifikacija* –Vrši se računanje karakteristične funkcije(4.2) μ_{edge} za svaki piksel iz slike. Pri tome pamtimo maksimalnu vrednost karakteristične funkcije. Neka to bude promenljiva MAX.
3) *Modifikacija vrednosti* - Za svaki piksel u slici se vrši modifikacija vrednosti karakteristične funkcije korišćenjem formule:

$$\mu_{edge}(I(i,j)) = \mu_{edge}(I(i,j)) / MAX$$

4) *Defazifikacija* – Generišemo izlaznu sliku na način da se u svaki piksel upiše intenzitet u skladu sa vrednošću karakteristične funkcije:

$$I(i,j) = (L-1) \times \mu_{edge}(I(i,j)).$$

Primena algoritma za detekciju ivica je prikazana na slikama 4.23 i 4.24:

Slika 4.23: Originalna slika.

Slika 4.24 : Slika sa izdvojenim ivicama.

Ono što je čitaocu možda zapalo za oko jeste da pikseli u slici imaju različite intenzitete bele svetlosti, odnosno da je na nekim mestima ona jako izražena, a da je na nekim mestima više liči na sivu. To je upravo posledica naše karakteristične funkcije, jer smo mi u svaki piksel „sipali“ onoliko belog koliko nam je ona rekla da uradimo. Ovakav aparat ima još jednu važnu prednost, a to je mogućnost ***klasifikacije ivica.*** Mi smo, zahvaljujući karakterističnoj funkciji,

u stanju da vršimo subjektivnu klasifikaciju ivica i da na osnovu iste izaberemo sa kojim ivicama želimo da radimo. To je korisno kod npr. segmentacije slike i označavanja regiona, kada treba odrediti granicu između regiona. Osim toga ova pogodnost se može koristiti i na sledeći način: Mi možemo da napravimo takvu klasifikaciju da neke ivice označimo ***irelevantnim*** [16], što može biti veoma korisno ukoliko radimo sa slikom koja ima šumove koji bi mogli uticati na dalji tok nekog složenog algoritma. Tada se takve „ivice“ obeležavaju kao irelevantne i odbacuju, čime nam ostaju samo one prave, prečišćene, ivice. Na slikama 4.25, 4.26, 4.27 i 4.28 ćemo videti kako se to odvija:

Slika 4.25: Originalna slika.

Slika 4.26: Primena algoritma bez prečišćavanja

Slika 4.27: Odbačene ivice sa $\mu<0.2$

Slika 4.28: Odbačene ivice sa $\mu<0.45$

Zaključak

Teorija fazi skupova i fazi logika imaju širok spektar primene u procesiranju slika. Neke od tih primena smo videli u ovom radu i one se koriste i kod dosta složenijih algoritama kao što su operacije sa histogramom, skeletizacija, segmentacija i obeležavanje regiona i dr.. Glavna prednost ovakvog pristupa procesiranju slika jeste da se ovakvi algoritmi mogu lakše implementirati, odnosno daju jedan jednostavniji način procesiranja slika. Naravno, osim prednosti ovi pristupi takodje imaju svoje nedostatke. Nedostatak ovakvog načina procesiranja slika može biti brzina. Dakle ukoliko je neophodno vršiti procesiranje slike, onda brzina ne predstavlja nikakav problem, ali ako želimo da koristimo fazi procesiranje za procesiranje video snimaka ili snimaka u realnom vremenu, onda ovakav pristup ne bi bio najsrećniji jer se procesiranje vrši kroz nekoliko faza, tačnije imamo nekoliko prolazaka kroz sliku. Ovakav načina procesiranja slika je pojednostavljuje obradu slika i može je proširiti na mnoge oblasti.

Spisak Literature

[1] Zadeh, Lotfi A. "Fuzzy sets." *Information and control* 8.3 (1965): 338-353.

[2] Negoita, C. V., L. A. Zadeh, and H. J. Zimmermann. "Fuzzy sets as a basis for a theory of possibility." *Fuzzy sets and systems* 1 (1978): 3-28.

[3] Bellman, Richard E., and Lotfi Asker Zadeh. "Decision-making in a fuzzy environment." *Management science* 17.4 (1970): B-141.

[4] Bergmann, Merrie. *An introduction to many-valued and fuzzy logic: semantics, algebras, and derivation systems*. Cambridge University Press, 2008.

[5] Tanaka, Kazuo. "An introduction to fuzzy logic for practical applications." (1997).

[6] Adnađević, Dušan, Zoran Kadelburg, and Ivan Brankovan. *Matematička analiza I*. Matematički fakultet, 2003.

[7] Perovic, Aleksandar, Aleksandar Jovanovic, and Boban Velickovic. "Teorija skupova." *Matematicki fakultet, Beograd* (2007).

[8] Gonzales, Rafael C., and Richard E. Woods. "Digital Image Processing, 2-nd Edition." (2002).

[9] Haberacker, Peter. "Praxis der digitalen Bildverarbeitung und Mustererkennung." *Carl Hanser Verlag, Munchen* (1995).

[10] Burger, Wilhelm, and Mark J. Burge. "Digitale Bildverarbeitung." *Eine Einführung mit* (2005).

[11] Eddins, Steven L., R. C. Gonzalez, and R. E. Woods. "Digital image processing using Matlab." *Princeton Hall Pearson Education Inc., New Jersey* (2004).

[12] Keller, James, Raghu Krisnapuram, and Nikhil R. Pal. *Fuzzy models and algorithms for pattern recognition and image processing*. Vol. 4. Springer, 2005.

[13] Kerre, Etienne E., and Mike Nachtegael, eds. *Fuzzy techniques in image processing*. Vol. 52. Springer, 2000.

[14] Chi, Zheru, Hong Yan, and Tuan Pham. *Fuzzy algorithms: with applications to image processing and pattern recognition*. Vol. 10. World Scientific, 1996

[15] Pal, Sankar K. "Fuzzy sets in image processing and recognition." *Fuzzy Systems, 1992., IEEE International Conference on*. IEEE, 1992.

[16] Becerikli, Yasar, and Tayfun M. Karan. "A new fuzzy approach for edge detection." *Computational Intelligence and Bioinspired Systems*. Springer Berlin Heidelberg, 2005. 943-951.

[17] Marr, David, and Ellen Hildreth. "Theory of edge detection." *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207.1167 (1980): 187-217.

[18] Canny, John. "A computational approach to edge detection." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 6 (1986): 679-698.