SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Sveučilišni studij elektrotehnike

Vedran Furač

UKLANJANJE ŠUMA IZ SLIKE KORIŠTENJEM DVODIMENZIONALNE METODE RELATIVNOG PRESJECIŠTA INTERVALA POUZDANOSTI

DIPLOMSKI RAD

Rijeka, 2009.

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Sveučilišni studij elektrotehnike

Vedran Furač

UKLANJANJE ŠUMA IZ SLIKE KORIŠTENJEM DVODIMENZIONALNE METODE RELATIVNOG PRESJECIŠTA INTERVALA POUZDANOSTI

DIPLOMSKI RAD

Mentor: doc.dr.sc. Viktor Sučić

Vedran Furač
0069022285
Smjer: Automatizacija
postrojenja i mehatronika

Rijeka, rujan 2009.

TEHNIČKI FAKULTET
Povjerenstvo za diplomske ispite
Sveučilišnog studija elektrotehnike
Br.: 602-04/09-01/76
Rijeka, 05.06.2009.

ZADATAK
za diplomski rad

Pristupnik: **VEDRAN FURAČ** Matični broj: 0069022285

Naziv zadatka: **Uklanjanje šuma iz slike korištenjem dvodimenzionalne metode relativnog presjecišta intervala pouzdanosti**

Sadržaj zadatka:

Potrebno je ukloniti aditivni šum iz slike korištenjem dvodimenzionalne metode relativnog presjecišta intervala pouzdanosti. Za svaki slikovni element, metodom je potrebno odrediti odgovarajuću dvodimenzionalnu regiju, te estimirati iznos slikovnog elementa bez šuma na osnovi dobivene regije. Postupak je potrebno ponoviti za svaki slikovni element slike. Metodu je nužno testirati na realnim slikama i dobivene rezultate numerički iskazati i usporediti s postojećim metodama za uklanjanje šuma iz slike.

Zadano: **05.06.2009.** Predati: **04.09.2009.**

Mentor:

Doc.dr.sc. Viktor Sučić, dipl.ing.

Predsjednik Povjerenstva:

izv.prof.dr.sc. Vera Gradišnik, dipl.ing.

Zadatak preuzeo dana: 05.06.2009.

(potpis pristupnika)

Dostaviti:
- Predsjednik Povjerenstva
- Mentor
- Djelovođa povjerenstva
- Evidencija studija
- Pristupnik
- Arhiva Zavoda

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Sveučilišni studij elektrotehnike

IZJAVA:

U skladu s člankom 94. Pravilnika o dodiplomskom studiju Tehničkog fakulteta u Rijeci, izjavljujem da sam samostalno izradio Diplomski rad prema zadatku br. 602-04/09-01/76 od 05.06.2009.

Rijeka, rujan 2009.

_____

Ime prezime

1. Sadržaj

1. Uvod

U ovom je diplomskom radu opisan postupak uklanjanja šuma iz slike korištenjem dvodimenzionalne metode relativnog presjecišta intervala pouzdanosti (eng. *relative intersection of confidence intervals* – RICI). Na samom početku dana je teorijska pozadina digitalnih slika, šuma kao i osnova teorije estimacije. Zatim je prikazana jednodimenzionalna ICI metoda, kao i poboljšanja RICI metoda koja će biti korištena prilikom reduciranja šuma i na kojoj se bazira dvodimenzionalna metoda predstavljena u slijedećem poglavlju. Izvršen je veliki broj simulacija čiji su rezultati, odnosno neki od njih, prikazani kako slikom, tako i numeričkom mjerom kvalitete kako bi se pronašla optimalna kombinacija parametara. Na kraju je dana usporedba RICI metode s drugim, često korištenim, metodama kao i programski kôd samog algoritma razvijen za potrebe ovog rada.

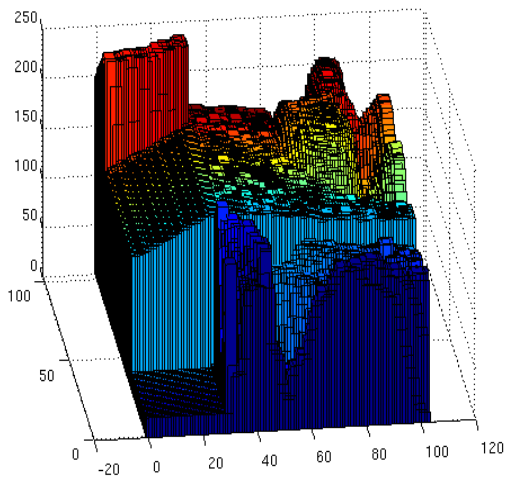
Kao što je poznato, svakim prikupljanjem informacija, odnosno u našem slučaju snimanjem svakog signala, neželjena, ali uvijek prisutna je pojava šuma. Svaki dio signala koji ne sadrži korisne informacije, već najčešće slučajne fluktuacije mjerene veličine u vremenu, naziva se šumom. Šum nastaje između samog izvora signala i detektora, kao i zbog nesavršenosti mjernog uređaja. Bez obzira koliko se učini s ciljem sprječavanja šuma određena količina će uvijek biti prisutna jer mu je podložna svaka elektronička naprava. Ovaj rad se prvenstveno bavi reduciranjem slučajnog šuma (eng. *random noise*) čija je posljedica nasumična neželjena promjena intenziteta pojedinih slikovnih elemenata. Metoda obrađena u ovom radu iskorištava činjenicu da će srednja vrijednost takvog šuma biti jednaka nuli. Postoji veliki broj algoritama koji imaju za cilj ukloniti ili reducirati šum obrađujući signal bilo u vremenskoj (odnosno prostornoj ako je signal slika), frekvencijskoj ili domeni valića (eng. *wavelets*). Ovdje opisani algoritam je lokalno adaptivan te obrađuje signal u prostornoj (eng. *spatial*) domeni. Te karakteristike omogućuju mu veliku učinkovitost pri uklanjanju šuma, a posebno ako signal sadrži skokove i nagle promijene strmine. Pokazalo se kako je srednja kvadratna pogreška estimacije dobivena RICI postupkom manja od one dobivene korištenjem postupaka temeljenih na valićima.

2. Osnove digitalnih slika

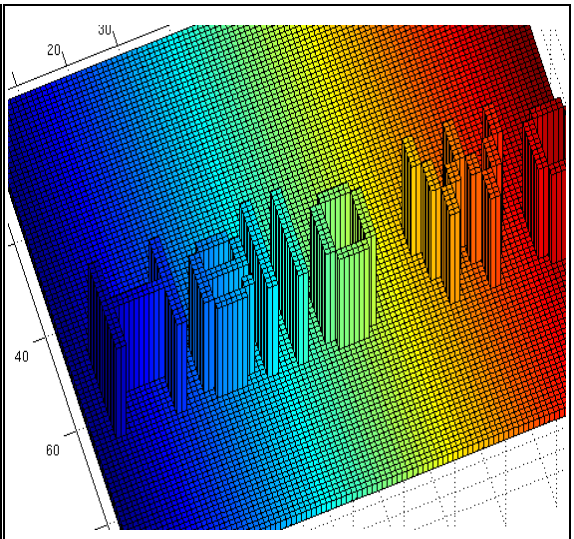
Slika se definira kao dvodimenzionalna funkcija, odnosno funkcija dvije varijable, gdje vrijednosti funkcije *f(x,y)* predstavljaju svjetlinu bilo koje točke i mogu poprimiti bilo koje realne vrijednosti između, na primjer, 0.0 (crne) i 1.0 (bijele).

Da bi sliku bilo moguće obraditi na računalu potrebno ju je imati u digitalnom formatu. Većina fotoaparata danas automatski snima slike u standardizirani digitalni format (JPEG, PNG, TIFF), a ukoliko to nije slučaj, sliku je potrebno digitalizirati.

Digitalna slika, za razliku od gore navedene, predstavlja funkciju *f(x,y)* čije su vrijednosti diskretizirane odnosno jednoliko kvantizirane. Svaki se takav slikovni element, čija je pozicija unutar slike definirana vrijednostima *x* i *y*, naziva **piksel** (eng. *picture element – pixel*). Prostorna vizualizacija digitalnih slike prikazana je na slikama 2.1 i 2.2.



Slika 2.1: 3D prikaz vrijednosti slikovnih elemenata fotografije



Slika 2.2: 3D prikaz vrijednosti slikovnih elemenata slike s bijelim tekstom na crnoj podlozi

Vrijednosti slikovnih elemenata su cijeli brojevi, a ovisno o tome koje sve cjelobrojne vrijednosti mogu poprimiti, digitalne slike dijelimo na [1]:

- Binarne (1-bitne)
- Monokromatske (npr. *gray-scale* slike u nijansama sive boje)
- Višekanalne slike (slike u boji)

Kod binarnih slika svaki slikovni element može poprimiti jednu od samo dvije moguće vrijednosti, 1 ili 0.

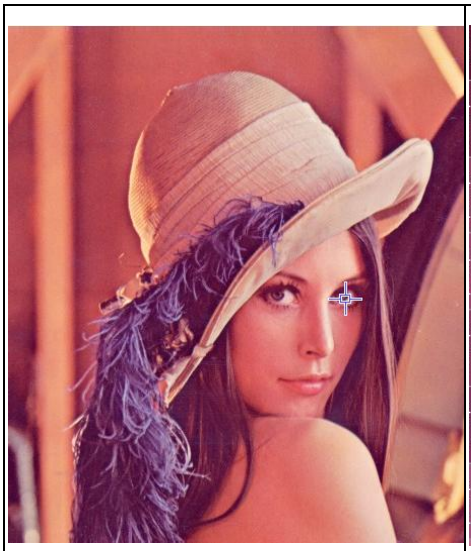
Monokromatske slike sastoje se od piksela koji poprimaju vrijednosti samo jedne boje. Iako nije nužno, takve slike su često u nijansama sive boje. Broj nijansi određen je brojem kojim nazivamo **dubina boje** *(*eng. *color depth)*, a najčešće se označavaju s potencijama broja 2. Tako najčešće imamo 4-bitne, 8-bitne (slika 2.3) ili 16-bitne dubine boja koje daju 16, 256 odnosno 65536 nijansi.

Slika 2.3: 8-bitna monokromatska slika

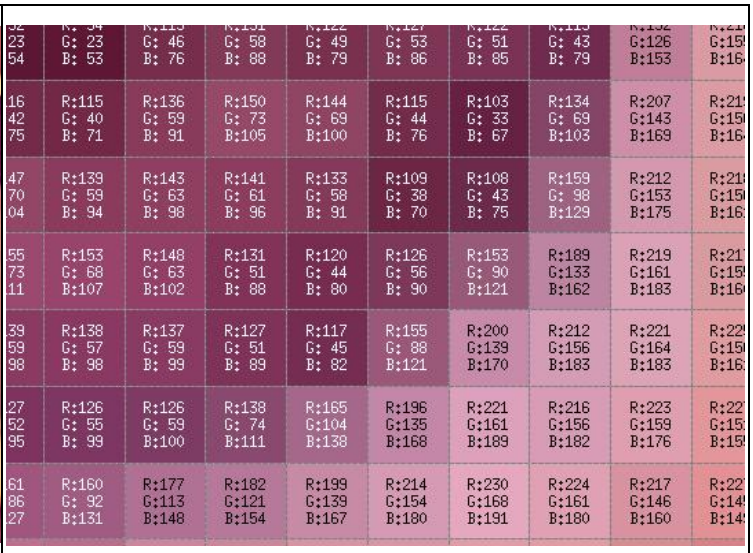
219	215	214	203	196	136	123	144	139	148	161	161
207	202	205	213	202	132	133	174	170	166	176	177
204	215	221	220	202	140	148	175	181	178	180	183
220	221	221	222	199	158	178	186	186	188	190	197
217	216	223	226	194	145	127	106	125	125	124	139
212	214	203	174	126	91	86	55	57	58	57	60
174	147	128	124	102	82	86	57	45	43	44	42
103	98	114	134	135	60	51	44	44	43	41	41

Slika 2.4: Detalj slike lijevo s prikazanim vrijednostima slikovnih elemenata

Višekanalne slike za pojedini slikovni element koriste više od jedne vrijednosti. Najčešće se koriste aditivni model boja s tri vrijednosti (crvena, zelena i plava - RGB) i subtraktivni s četiri vrijednosti (modrozelena (eng. *cyan)*, ljubičasta (eng. *magenta*), žuta i crna - CMYK)[2]. Kombiniranjem tih vrijednosti određuje se boja svakog slikovnog elementa. Danas se najčešće za svaki kanal slike koristi dubina boje od 8 bita za svaki kanal (vrijednosti od 0 do 255) što u konačnici daje 2^{24} mogućih vrijednosti pojedinih slikovnih elemenata odnosno ~16.77 milijuna boja (slika 2.5).



Slika 2.5: 24-bitna truecolor slika



23 54	G: 23 B: 53	G: 46 B: 76	G: 58 B: 88	G: 49 B: 79	G: 53 B: 86	G: 51 B: 85	G: 43 B: 79	G:126 B:153	G:15 B:16
16 42 75	R:115 G: 40 B: 71	R:136 G: 59 B: 91	R:150 G: 73 B:105	R:144 G: 69 B:100	R:115 G: 44 B: 76	R:103 G: 33 B: 67	R:134 G: 69 B:103	R:207 G:143 B:169	R:21 G:15 B:16
47 70 04	R:139 G: 59 B: 94	R:143 G: 63 B: 98	R:141 G: 61 B: 96	R:133 G: 58 B: 91	R:109 G: 38 B: 70	R:108 G: 43 B: 75	R:159 G: 98 B:129	R:212 G:153 B:175	R:21 G:15 B:16
55 73 11	R:153 G: 68 B:107	R:148 G: 63 B:102	R:131 G: 51 B: 88	R:120 G: 44 B: 80	R:126 G: 56 B: 90	R:153 G: 90 B:121	R:189 G:133 B:162	R:219 G:161 B:183	R:21 G:15 B:16
39 59 98	R:138 G: 57 B: 98	R:137 G: 59 B: 99	R:127 G: 51 B: 89	R:117 G: 45 B: 82	R:155 G: 88 B:121	R:200 G:139 B:170	R:212 G:156 B:183	R:221 G:164 B:183	R:22 G:15 B:16
27 52 95	R:126 G: 55 B: 99	R:126 G: 59 B:100	R:138 G: 74 B:111	R:165 G:104 B:138	R:196 G:135 B:168	R:221 G:161 B:189	R:216 G:156 B:182	R:223 G:159 B:176	R:22 G:15 B:15
61 86 27	R:160 G: 92 B:131	R:177 G:113 B:148	R:182 G:121 B:154	R:199 G:139 B:167	R:214 G:154 B:180	R:230 G:168 B:191	R:224 G:161 B:180	R:217 G:146 B:160	R:22 G:14 B:14

Slika 2.6: Detalj slike lijevo s prikazanim vrijednostima slikovnih elemenata

Kao što je prije navedeno slike moraju biti u jednom od brojnih digitalnih formata kako bi mogle biti obrađivane na računalu. Za obradu ćemo koristiti programski paket MATLAB koji podržava veliki broj različitih formata. Nakon što učitamo sliku korištenjem funkcije *imread* u MATLAB-u ćemo dobiti matricu dimenzija MxN, gdje M i N predstavljaju dimenzije slike. Jednostavnosti radi, u ovom radu koristiti ćemo slike od kojih je svaka monokromatska, 8-bitna, te se kao takva sastoji od 256 nijansi sive boje (eng. *gray-scale*). Pri tome vrijednost 0 označuje najtamniju nijansu (crnu boju), dok 255 predstavlja najsvjetliju nijansu (bijelu boju). Slike korištene u ovom radu prikazane su na slikama 2.7 - 2.10 [3].

Slika 2.8: boat

Slika 2.7: montage

Slika 2.9: peppers

Slika 2.10: camera

3. O šumu

Šum predstavlja slučajne (stohastičke) varijacije svjetline ili boje u slici nastalih u senzoru i elektroničkom sklopu digitalne kamere ili skenera. Možemo ga predstaviti kao slučajnu varijablu sa svojom srednjom vrijednosti, standardnom devijacijom kao i varijancom. Šum se dijeli na [4]:

- Fotoelektronički (termalni, fotonski)
- Impulsni (*salt and pepper*)
- Strukturirani (periodični nestacionarni, periodični stacionarni, aperiodični)

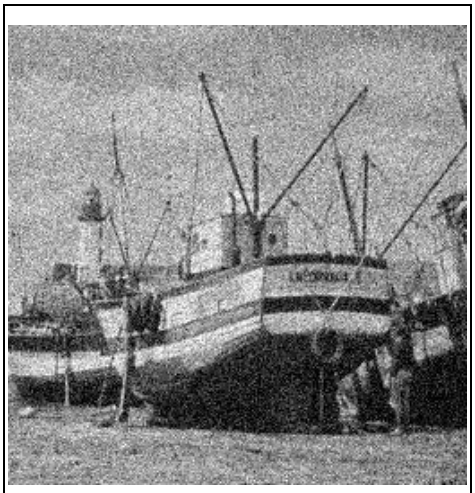
Za potrebe ovog diplomskog rada testnim slikama dodati ćemo Gaussov bijeli šum sa srednjom vrijednosti 0 i konstantnom standardnom devijacijom kao i varijancom. U praksi se pokazalo da je šum koji nastaje u procesu stvaranja slika upravo Gaussov bijeli šum. Taj je šum zatim potrebno ukloniti korištenjem lokalno adaptivnog algoritma u kombinaciji s pravilom (relativnog) presjecišta intervala pouzdanosti.

U MATLAB-u šum se najčešće dodaje korištenjem funkcije *imnoise(I, 'gaussian', mean, variance)*. Drugi argument funkcije označuje tip šuma kojeg dodajemo i ne mora nužno biti *gaussian*, već može biti i jedan od [4]:

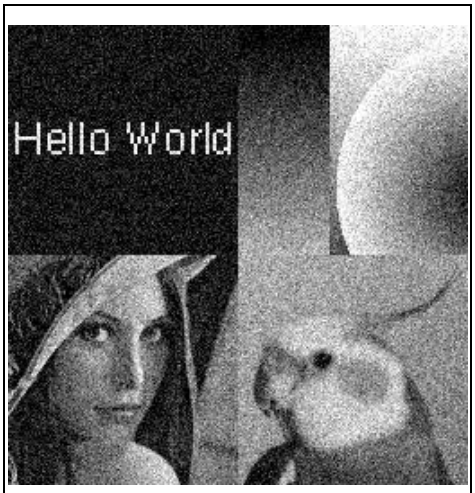
- *'localvar'*
- *'poisson'*
- *'salt & pepper'*
- *'speckle'*

Mean označuje srednju vrijednost, dok *variance* predstavlja varijancu odnosno kvadrat standardne devijacije. Radi veće fleksibilnosti i jasnijeg prikaza samog dodavanja šuma, u ovom je radu korištena funkcija *randn* koja generira matricu s normalno raspodijeljenim vrijednostima čija je srednja vrijednost jednaka 0, a standardna devijacija i varijanca iznose 1. Da bi se postigla druga vrijednost standardne devijacije dovoljno je pomnožiti vrijednosti matrice s iznosom standardne devijacije koju želimo dobiti. Dobivenu matricu šuma dovoljno je jednostavno pribrojiti matrici koja predstavlja učitanu sliku kako bi dobili zašumljenu sliku. Pritom treba paziti da vrijednosti koje tada nastanu ne budu manje od 0 i veće od 255, odnosno da vrijednosti slikovnih elemenata ostanu u intervalu od 0 do 255. Slike 3.1-3.4 su

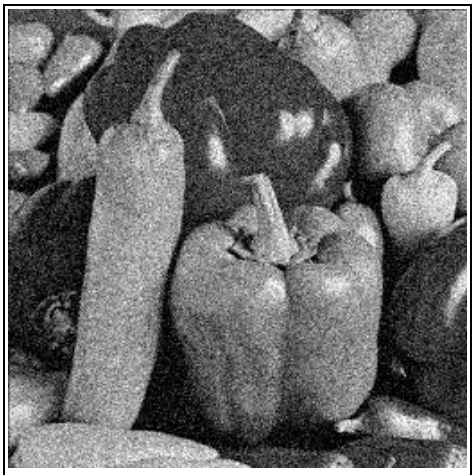
zašumljene Gaussovim bijelim šumom standardne devijacije u iznosu od 25 i cilj rada je razviti algoritam koji će ukloniti šum iz tih slika.



Slika 3.1:



Slika 3.2:



Slika 3.4:

Slika 3.3:

4. Teorija estimacije

4.1 Općenito o teoriji estimacije

U praksi nije rijedak slučaj da vrijednost nekog parametra nije moguće izmjeriti pa ga je potrebno nekako procijeniti, odnosno aproksimirati, estimirati. Metode estimacije možemo podijeliti u dvije osnovne grupe [5]:

- Metode kod kojih je vrijednost nekog parametra estimirana kroz dva broja koja predstavljaju interval unutar kojeg se, s određenom vjerojatnošću, nalazi vrijednost tog parametra (eng. *interval estimate*)
- Metode kod kojih je vrijednost parametra estimirana jednom vrijednošću, odnosno jednim brojem (eng. *point estimate*)

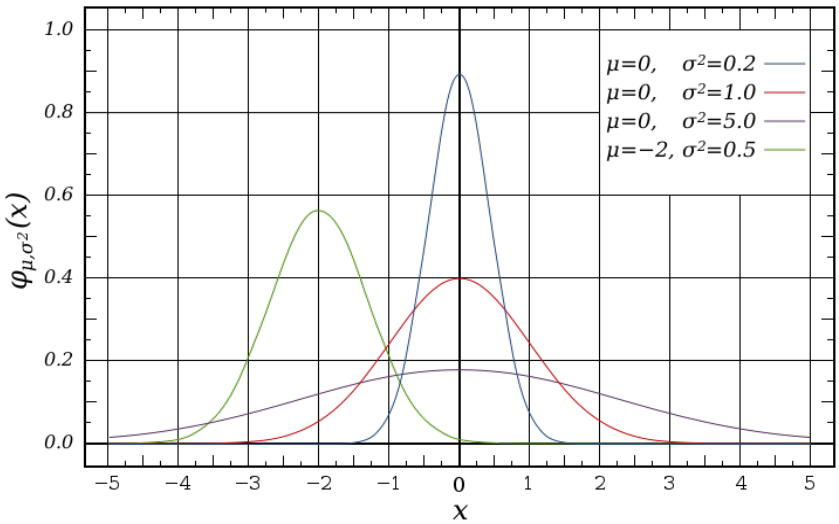
Ovaj diplomski rad bazira se na određivanju intervala pouzdanosti za svaki slikovni element odnosno vrijednosti unutar kojih se s određenom pouzdanošću nalazi stvarna vrijednost signala.

4.2 Normalna distribucija šuma

Prilikom snimanja digitalnih slika, šum koji se pojavljuje može se smatrati slučajnom varijablom koja je normalno raspodijeljena. Aritmetička sredina takve slučajne varijable jednaka je nuli. Funkcija gustoće kontinuirane normalne distribucije glasi [6]:

$$f(x)=\frac{1}{\sigma\sqrt{2\pi}}\mathrm{e}^{\frac{-(x-\mu)^2}{2\sigma^2}}, \tag{4-1}$$

gdje μ i σ označuju aritmetička sredinu i standardnu devijaciju.



Slika 4.1: Neke karakteristične krivulje gustoće normalne distribucije

Pripadna funkcija distribucije je [6]:

$$F(x)=\frac{1}{\sigma\sqrt{2\pi}}\int_{\infty}^{x} e^{\frac{-(v-\mu)^2}{2\sigma^2}}\,dv=P(X<x)\ . \tag{4-2}$$

Ako je Z standardizirana varijabla X s normalnom distribucijom onda vrijedi [6]:

$$Z=\frac{X-\mu}{\sigma}\ . \tag{4-3}$$

Pritom aritmetička sredina varijable Z raspodijeljene po normalnoj distribuciji iznosi 0, a varijanca kao i standardna devijacija iznose 1. Tada je funkcija gustoće reducirana na [6]:

$$f(z)=\frac{1}{\sqrt{2\pi}}e^{\frac{-z^2}{2}}\ , \tag{4-4}$$

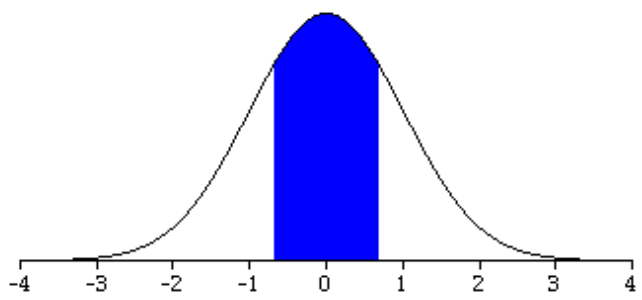
a odgovarajuća funkcija distribucije iznosi [6]:

$$F(z)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{z} e^{\frac{-u^2}{2}}du=\frac{1}{2}+\frac{1}{\sqrt{2\pi}}\int_{0}^{x} e^{\frac{-u^2}{2}}du=P(Z<z)\ . \tag{4-5}$$

Vjerojatnost da se vrijednost kontinuirane slučajne varijable Z nalazi u intervalu (z_1, z_2) jednaka je razlici funkcija distribucije u mjestima z_1, z_2[6]:

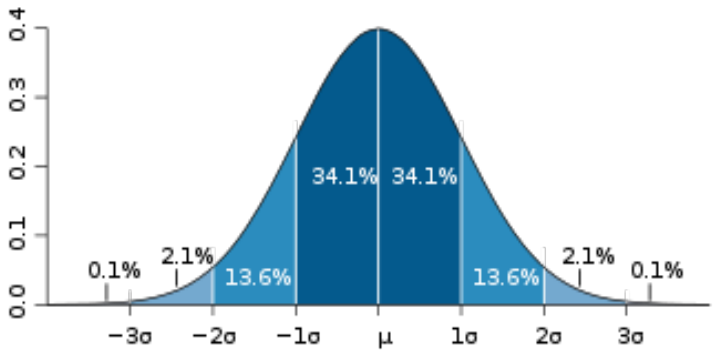
$$P(z_1<z\leq z_2)=\int_{z_1}^{z_2} f(z)dz=F(z_2)-F(z_1)\ . \tag{4-6}$$

Za primjer je na slici 4.2 uzet slučaj kada je -0.675<Z<0.675 što odgovara vjerojatnosti od 50%. Drugim riječima, s 50% vjerojatnosti smo sigurni da se varijabla Z nalazi unutar intervala od -0.675 do +0.675, a upravo površina ispod krivulje određuje tu vjerojatnost.



Slika 4.2: Vjerojatnost da se vrijednost slučajne varijable nalazi u intervalu (-0.675, +0.675) jednaka je površini ispod krivulje koja je u ovom slučaju 50% od ukupne vrijednosti

Ukupna površina ispod funkcije gustoće standardne normalne distribucije iznosi 1. Općenito, površina ispod standardizirane normalne krivulje, a između dviju ordinata podignutim na mjestima apscise z_1 i z_2 je ujedno i vjerojatnost da se slučajna varijabla Z nalazi između vrijednosti z_1 i z_2.



Slika 4.3: Postoci vjerojatnost u ovisnosti o površini i broju standardnih devijacija od aritmetičke sredine

4.3 Intervali pouzdanosti za estimaciju parametra skupa

Neka su μ_s i σ_s aritmetička sredina i standardna devijacija slučajne varijable S. Ako su vrijednosti slučajne varijable S normalno raspodijeljene (što vrijedi za statističke uzorke veličine n>30), možemo očekivati da se S nalazi unutar intervala $\langle \mu_s - \sigma_s, \mu_s + \sigma_s \rangle$, $\langle \mu_s - 2\sigma_s, \mu_s + 2\sigma_s \rangle$ i $\langle \mu_s - 3\sigma_s, \mu_s + 3\sigma_s \rangle$ s pripadnim vjerojatnostima u iznosu od 68.27%, 95.45% i 99.73% [7].

Jednako možemo očekivati da se μ_s nalazi u intervalu $\langle S - \sigma_s, S + \sigma_s \rangle$, $\langle S - 2\sigma_s, S + 2\sigma_s \rangle$ i $\langle S - 3\sigma_s, S + 3\sigma_s \rangle$ s vjerojatnostima 68.27%, 95.45% i 99.73%. Stoga te intervale nazivamo intervalima pouzdanosti za estimaciju μ_s (odnosno za estimaciju parametra skupa u slučaju nepristranog S). Rubne vrijednosti intervala se nazivaju granicama intervala pouzdanosti od 68.27%, 95.45% i 99.73%.

Slično tome, $\langle S - 1.96\sigma_s, S + 1.96\sigma_s \rangle$ $\langle S - 2.58\sigma_s, S + 2.58\sigma_s \rangle$ su granice intervala pouzdanosti 95% i 99%. Postotne vrijednosti se često nazivaju razinom vjerojatnosti. Brojevi 1.96, 2.58, itd. u granicama intervala pouzdanosti se obično nazivaju koeficijenti pouzdanosti ili kritične vrijednosti, a označavaju se sa z_c. Slijedeća tablica prikazuje vrijednosti z_c koje odgovaraju različitim razinama vjerojatnosti koje se često susreću u praksi. Ostale vrijednosti

koje nisu prisutne u tablici mogu se jednostavno izračunati iz formule za Gaussovu normalnu krivulju.

Razina vjerojatnosti (%)	99.73	99	98	96	95.45	95	90	80	68.27	50
z_c	3	2.58	2.33	2.05	2	1.96	1.65	1.28	1	0.67

Tablica 4.1: Razine vjerojatnosti i koeficijenti pouzdanosti

4.4 Intervali pouzdanosti za distribuciju srednjih vrijednosti

1. Broj elemenata skupa veći od 30.

Ako je slučajna varijabla S srednja vrijednost $\overline{X}$ nekog skupa, onda intervali pouzdanosti od 95% i 99% za estimaciju srednje vrijednosti skupa iznose $\overline{X}\pm 1.96\sigma$, odnosno $\overline{X}\pm 2.58\sigma$. Općenitije, može se reći da intervali pouzdanosti za estimaciju srednje vrijednosti skupa iznose [8]:

$$\bar{X} \pm z_c \sigma_{\bar{X}} \ , \tag{4-7}$$

gdje $\mathbf{z_c}$ ovisi o traženoj razini vjerojatnosti.

U slučaju uzimanja uzorka iz beskonačnog skupa, odnosno ako je skup konačan, a uzorke uzimamo na način da jedan uzorak može biti uzet više puta, onda vrijedi slijedeći izraz za standardnu devijaciju distribucije srednjih vrijednosti uzoraka[8]:

$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}} \tag{4-8}$$

pri čemu σ označava standardnu devijaciju cijelog skupa. Za taj slučaj intervali pouzdanosti za estimaciju srednje vrijednosti skupa iznose [8]:

$$\bar{X} \pm z_c \frac{\sigma}{\sqrt{n}} \ . \tag{4-9}$$

Na ovom se izrazu zasniva metoda (relativnog) presjecišta intervala pouzdanosti, a koji predstavlja teorijsku osnovu za ovaj rad.

S druge strane, ako je veličina skupa konačna ili se uzorci uzimaju bez ponavljanja, onda izraz glasi [8]:

$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}} \sqrt{\frac{N-n}{N-1}} \ , \tag{4-10}$$

a izraz (4-9) postaje:

$$\bar{X} \pm z_c \frac{\sigma}{\sqrt{n}} \sqrt{\frac{N-n}{N-1}} \ . \tag{4-11}$$

2. Broj elemenata skupa manji od 30

Prethodni izrazi vrijede ako je broj elemenata skupa velik. U protivnom, kada je njihov broj manji od 30 potrebno je koristiti T razdiobu kod koje izraz za intervale pouzdanosti srednjih vrijednosti skupova glasi [8]:

$$\bar{X} \pm t_c \frac{\hat{S}}{\sqrt{n}} \ . \tag{4-12}$$

U ovom su radu korišteni izrazi (4-8) i (4-9) jer je skup koji koristimo konačan, ali se istim elementom skupa može računati više puta.

5. Metoda presjecišta intervala pouzdanosti

Prije nego krenemo na obradu slike dvodimenzionalnom, pozabavit ćemo se jednodimenzionalnom metodom presjecišta intervala pouzdanosti. na kojoj se bazira dvodimenzionalna. Naš lokalno adaptivni filtar koji koristi metodu presjecišta intervala pouzdanosti za svaki uzorak signala računa intervale pouzdanosti lijevo i desno od njega. Intervali pouzdanosti računaju se prema izrazima [8]:

$$U_k(n) = \bar{X} + z_c \frac{\sigma}{\sqrt{n}} \tag{5-1}$$

i

$$L_k(n) = \bar{X} - z_c \frac{\sigma}{\sqrt{n}} \ , \tag{5-2}$$

za gornju, odnosno donju granicu intervala pouzdanosti. Zbog $\sqrt{n}$ u nazivniku svaki će slijedeći interval biti uži od prethodnog. Za prvi uzorak vrijedi n=1, $\bar{X}$ je jednak vrijednosti uzorka x_n (aritmetička sredina skupa od samo jednog člana jednaka je vrijednosti tog člana), a z_c i σ su konstante. Za slijedeći interval pouzdanosti na desno vrijednosti su n=2, $\bar{X} = \frac{x_n + x_{n+1}}{2}$, dok su vrijednosti z_c i σ iste kao i za prethodni uzorak. Sada treba vidjeti da li se intervali međusobno presijecaju što vrijedi ako je iznos najvišeg donjeg manja ili jednaka najnižem gornjem intervalu pouzdanosti [9]:

$$max_{i=1,\ldots,k}(L_i(n)) \ \leq \ min_{i=1,\ldots,k}(U_i(n)) \tag{5-3}$$

Ako je gornji izraz istinit, preklapanje postoji pa krećemo na slijedeći uzorak na desno: n=3, $\bar{X} = \frac{x_n + x_{n+1} + x_{n+2}}{3}$, a vrijednosti z_c i σ ostaju iste. Zatim ponovo provjeravamo postoji li preklapanje sva tri intervala i ako više ne postoji (ili smo došli do kraja snimljenoga signala) tu postupak staje pa krećemo s računanjem intervala pouzdanosti u lijevo od trenutnog uzorka. Postupak se ponavlja u lijevo sve dok se svi intervali međusobno presijecaju ili dok ne dođemo do prvog uzorka signala.

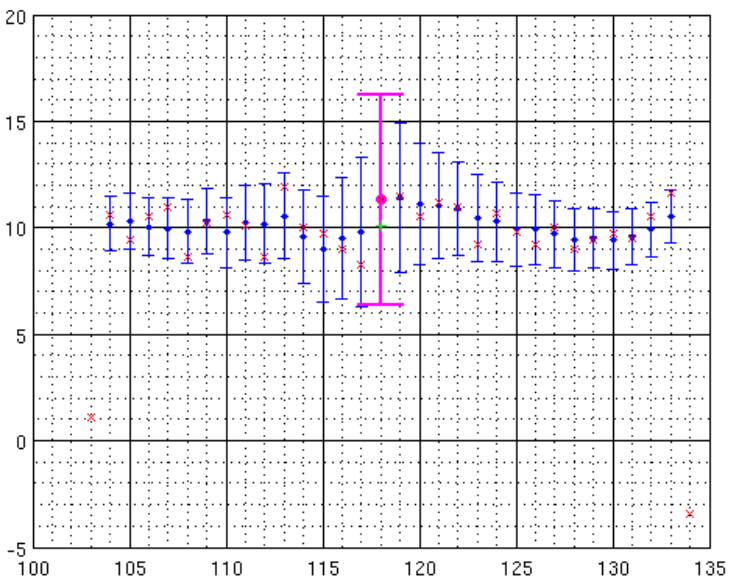
U praksi se pokazalo da bolje rezultate daje modificirana metoda u kojoj se mijenja način računanja $\bar{X}$. Promjena se sastoji u tome da se broj uzoraka za računanje $\bar{X}$ ograniči na posljednjih nekoliko (najčešće $2 \leq n_{\bar{X}} \leq 5$, ovisno o signalu). Tako na primjer, ako računamo četvrti interval pouzdanosti u desno i uzmemo samo posljednja tri uzorka, izraz za $\bar{X}$ neće biti

$\bar{X}=\frac{x_n+x_{n+1}+x_{n+2}+x_{n+3}}{4}$, nego $\bar{X}=\frac{x_{n+1}+x_{n+2}+x_{n+3}}{3}$. Isto vrijedi i na lijevo. Ako uzmemo samo posljednja dva uzorka pri računanju trećeg intervala pouzdanosti u lijevo, izraz neće biti $\bar{X}=\frac{x_n+x_{n-1}+x_{n-2}}{3}$, već $\bar{X}=\frac{x_{n-1}+x_{n-2}}{2}$.

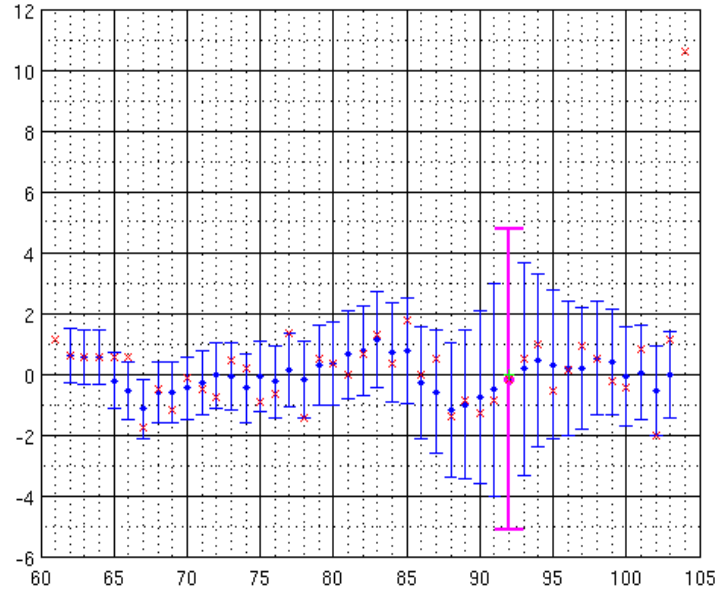
Pri izračunu nove vrijednosti uzorka signala (oko kojeg računamo intervale pouzdanosti), u obzir se uzimaju samo one vrijednosti kod kojih je došlo do presijecanja intervala pouzdanosti. Širina intervala može se mijenjati promjenom koeficijenta pouzdanosti $\mathbf{z_c}$. Interval će biti širi ako izaberemo koeficijent kojemu odgovara veća vjerojatnost, odnosno uži ako odaberemo koeficijent s nižom vjerojatnosti.

Konačno, da bi dobili estimiranu vrijednost signala potrebno je izračunati aritmetičku sredinu svih onih uzoraka na lijevo i desno čiji se intervali pouzdanosti međusobno presijecaju. Time je postupak dovršen s izračunatom novom, estimiranom vrijednosti jednog uzorka nekog signala. Postupak je potrebno ponoviti za svaki uzorak signala.

Radi lakšeg razumijevanja, metoda je grafički objašnjenja na slikama 5.1 i 5.2. Crvenim su znakovima x označene pojedini uzorci odnosno vrijednosti signala. Plavim su točkama označene $\bar{X}$ vrijednosti s pripadnim gornjim i donjim intervalima. Vidljivo je da rubni uzorci nisu prikazani s pripadnim intervalima jer se nisu presijecali sa svim prethodnim intervalima. Sa zelenim + označena je aritmetička sredina vrijednosti svih uzoraka signala čiji se intervali pouzdanosti preklapaju (plave linije na slici), što je ujedno i konačna estimirana vrijednost uzorka.



Slika 5.1: Vrijednosti signala i pripadni intervali pouzdanosti



Slika 5.2: Vrijednosti signala i pripadni intervali pouzdanosti

5.1 Metoda relativnog presjecišta intervala pouzdanosti

Ovisno o odabranoj vrijednosti koeficijenta $\mathbf{z_c}$ dobiti ćemo veće ili manje izglađivanje signala. Cilj je prilagoditi vrijednost z_c, tako da se zagladi samo dio signala koji nema nagle promijene, a da se pritom sačuvaju svi skokovi i nagle promijene signala. Niže opisana metoda ne zahtjeva prethodno poznavanje optimalne vrijednosti z_c, a temelji se na omjeru duljine presjecišta intervala pouzdanosti i duljine trenutnog intervala pouzdanosti [10]:

$$R_k(n) = \frac{U_{kmin}(n) - L_{kmax}(n)}{U_k(n) - L_k(n)} = \frac{U_{kmin}(n) - L_{kmax}(n)}{2z_c \frac{\sigma}{\sqrt{n}}} . \qquad (5\text{-}4)$$

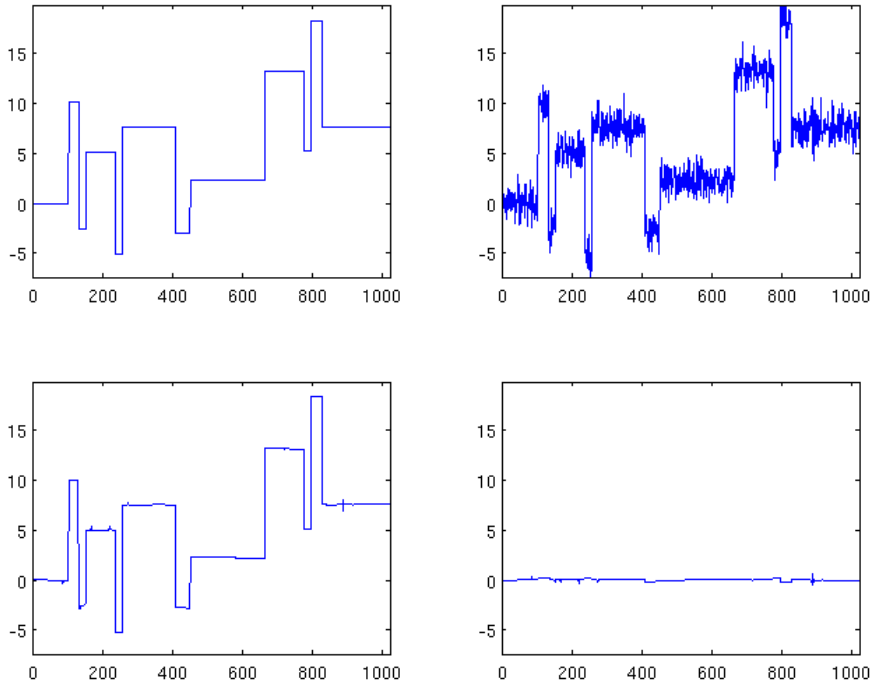
Zatim se $R_k(n)$ koristi kao dodatni kriterij (uz (5-3)) pri odabiru adaptivne pojasne širine filtra, odnosno broja uzoraka s lijeve i desne strane koji se koriste za izračun estimirane vrijednosti trenutnog uzorka [10]:

$$R_k(n) \geq R_c \qquad (5\text{-}5)$$

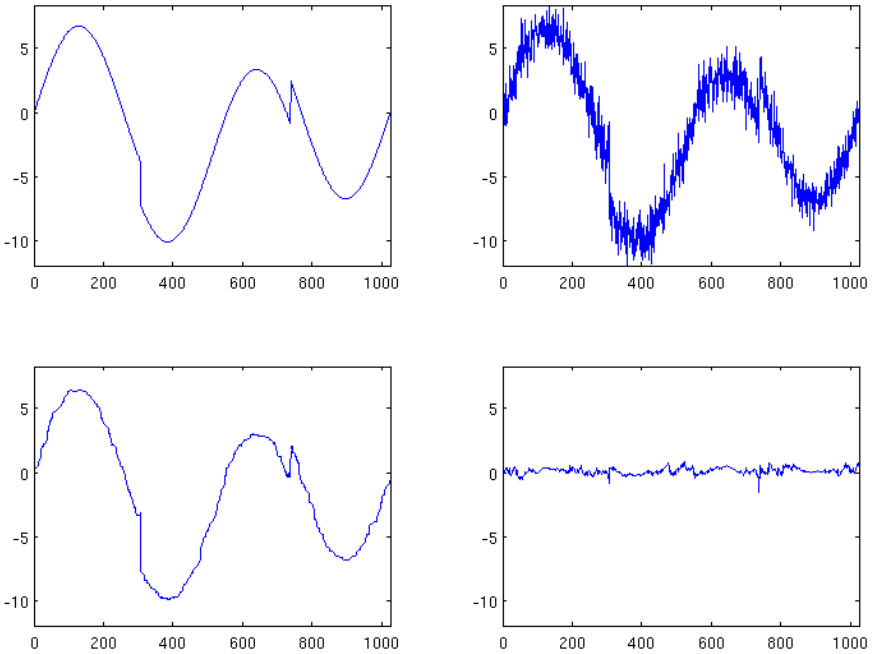
gdje je R_c zadana vrijednost praga odabrana na temelju većeg broja numeričkih simulacija. Iako optimalna vrijednost praga R_c ovisi o izboru koeficijenta z_c, koji i ne mora biti optimalan, prag R_c se može smatrati neovisnim o signalu. Budući da se ovaj postupak zasniva na metodi relativnog presjecišta intervala pouzdanosti, nazvan je RICI [10]. S ovim dodatnim kriterijom

postignuto je značajno poboljšanje u detekciji rubova (skokova) u signalu, a samim tim je povećana i točnost estimacije.

Na slikama 5.3 i 5.4 dani su primjeri uklanjanja šuma "*blocks*" i sinusnog signala korištenjem RICI metode. Na svakoj se slici nalaze četiri grafa: originalni signal, zašumljeni signal, filtrirani signal te razlika između filtriranog i originalnoga signala. Vidljivo je da su u filtriranom signalu sačuvani svi skokovi, a oni dijelovi signala s malim promjenama (uzrokovanim šumom) su zaglađeni.



Slika 5.3: "Blocks" signal



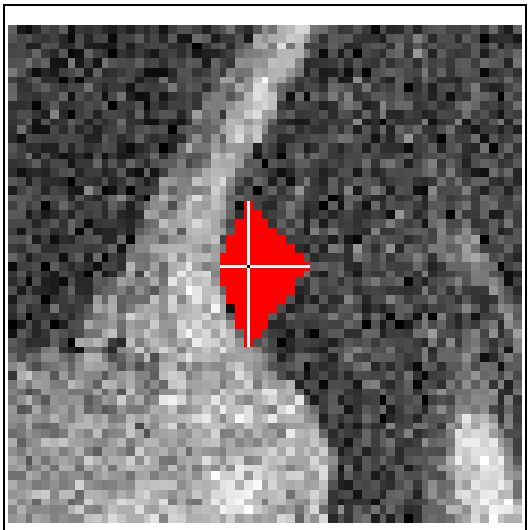
Slika 5.4: "Heavysine" signal

6. Dvodimenzionalna metoda relativnog presjecišta intervala pouzdanosti

6.1 Određivanje regija

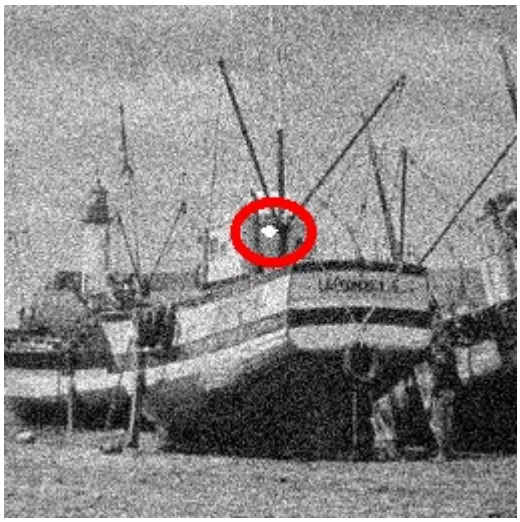
Za svaki slikovni element čiju vrijednost želimo estimirati, potrebno je oko njega pronaći regiju u kojoj se intervali pouzdanosti svakog slikovnog elementa međusobno presijecaju. U ovom radu ta regija je četverokut odnosno osmerokut čiji su vrhovi oni slikovni elementi kod kojih su se intervali pouzdanosti prestali presijecati. Lokacije vrhova su određene jednodimenzionalnom metodom i to na način da su se za trenutni slikovni element izvukli pripadni red i stupac slike na koje je zatim primijenjena RICI metoda. U slučaju osmerokutne regije RICI metoda je primijenjena i na dva dodatna pravca koja su zakrenuta za 45° u odnosu na red i stupac slike u kojima se nalazi slikovni element čija se vrijednost estimira.

Za izdvajanje regije korištena je MATLAB funkcija *roipoly* koja na osnovu danih točaka (vrhova poligona) vraća binarnu masku s regijom označenom jedinicama, a ostatak slike nulama. Zatim se ta binarna maska množi s originalnom slikom kako bi se dobile samo vrijednosti unutar regije, a izvan nje nule. Na kraju se korištenjem "*Logical Subscripting*" izdvajaju vrijednosti slikovnih elemenata regije te se sa funkcijom *mean* računa njihova aritmetička sredina. Na slici 6.1 prikazan je detalj sa slike na kojem je crnom točkom označen estimirani slikovni element, bijelom bojom pravci na koje je primijenjen jednodimenzionalani RICI algoritam kao i četverokutnu regiju oko njega (crveno).

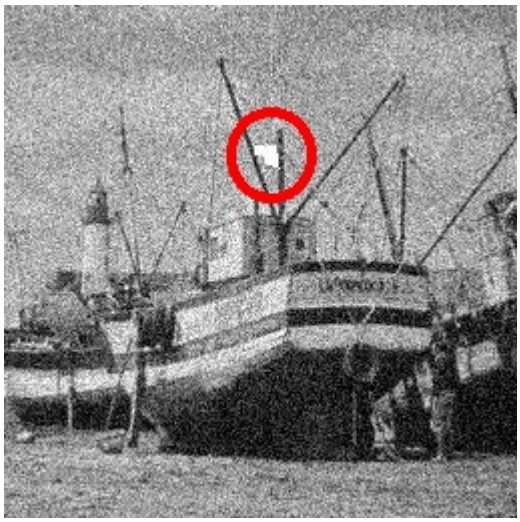


Slika 6.1: Detalj slike s označenom regijom

Na slikama 6.2-6.9 bijelom bojom su označene neke od četverokutnih i osmerokutnih regija na slikama koje se koriste kao primjeri uklanjanja šuma.



Slika 6.2:



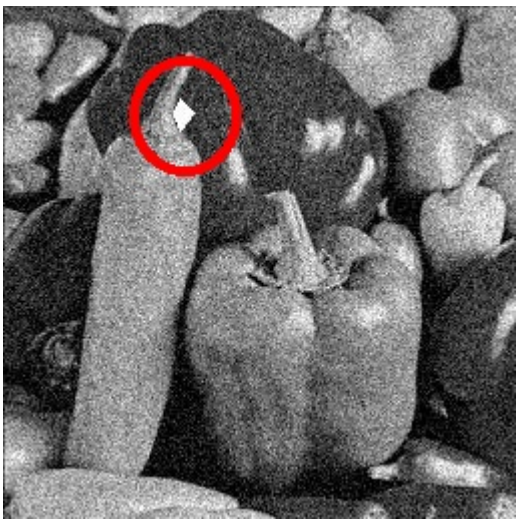
Slika 6.3:

Slika 6.4:

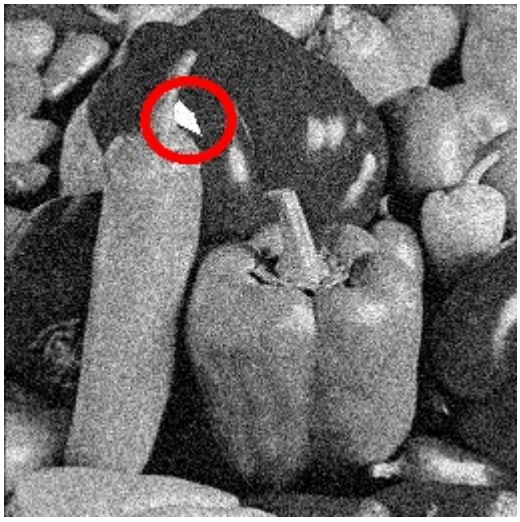
Slika 6.5:

Slika 6.6:

Slika 6.7:



Slika 6.8:



Slika 6.9:

6.2 Mjerenje kvalitete slike nakon uklanjanja šuma

Kvaliteta slike dobivene nakon uklanjanja šuma najčešće se mjeri kao odnos maksimalne snage signala i snage šuma koji je dodan slici (*Peak Signal-to-Noise Ratio - PSNR*). PSNR se definira preko srednje kvadratne pogreške *(Mean Square Error - MSE)* za dvije monokromatske slike X i Y veličina $m \times n$ [10]:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(X(i,j)-Y(i,j))^2 , \tag{6-1}$$

a PSNR je zatim[11]:

$$PSNR = 10\log\left(\frac{MAX_I^2}{MSE}\right) = 20\log\left(\frac{MAX_I}{\sqrt{(MSE)}}\right) , \tag{6-2}$$

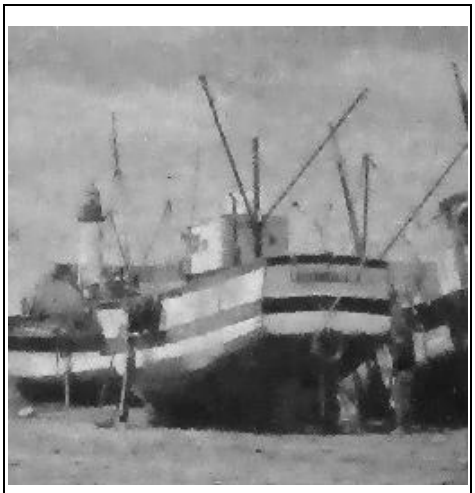
pri čemu MAX_I označava najveću moguću vrijednost koju element slike može poprimiti. S obzirom da su u ovom radu korištene 8-bitne slike, taj broj iznosi 255. Iz formule je vidljivo da će jedinica za PSNR biti dB. Veći PSNR označava da filtrirana slika bolje aproksimira originalnu sliku. U slučaju da su obje slike identične, MSE će biti 0, a PSNR beskonačan.

6.3 Filtriranje slika RICI metodom

Slijedeće slike prikazuju neke od rezultata uklanjanja Gaussovog šuma standardne devijacije 25 i srednje vrijednosti 0 (R)ICI metodom ovisno o broju vrhova regije (R4 i R8) te nekim od karakterističnih vrijednosti parametara z_c, R_c, kao i $n_{\bar{X}}$. Značenje svakog od tih parametara je detaljno objašnjeno u poglavlju 5. Za svaku od četiri slike na koje je dodan šum, odabrano je deset slika iz kojih je uklonjen šum korištenjem različitih kombinacija prethodno navedenih parametara:

Slika 6.10: boat. PSNR: 28.79dB

Slika 6.11: z_c=1.7, R_c=0.7, $n_{\bar{X}}$=3 , R4, PSNR = 31.37dB



Slika 6.12: z_c=1.7, R_c=0.8, $n_{\bar{X}}$=3 , R4, PSNR = 31.41dB

Slika 6.13: z_c*=1.8,* R_c*=0.8,*
$n_{\bar{X}}=3$ *, R4, PSNR = 31.53dB*

Slika 6.14: z_c*=1.9,* R_c*=0.8,*
$n_{\bar{X}}=2$ *, R4, PSNR = 31.64dB*

Slika 6.15: z_c*=1.9,* R_c*=0.8,*
$n_{\bar{X}}=3$ *, R4, PSNR = 31.46dB*

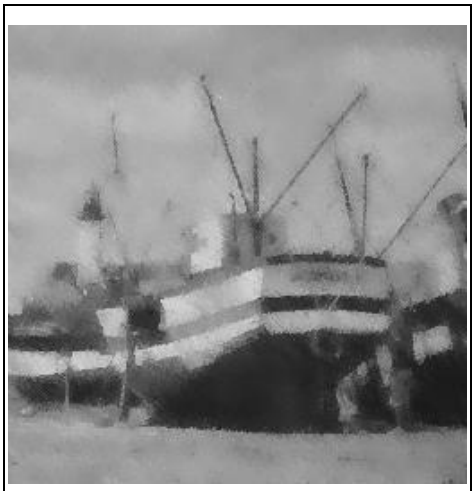
Slika 6.16: z_c*=2.0,* R_c*=0.7,*
$n_{\bar{X}}=3$ *, R4, PSNR = 31.41dB*

Slika 6.17: z_c*=2.0,* R_c*=0,*
$n_{\bar{X}}=3$ *, R4, PSNR = 30.95dB*

Slika 6.18: z_c*=2.0,* R_c*=0.9,*
$n_{\bar{X}}=3$ *, R4, PSNR = 31.47dB*

Slika 6.19: z_c=1.8, R_c=0.8,

$n_{\bar{X}}=\infty$ *, R4, PSNR = 30.64dB*



Slika 6.20: z_c=1.9, R_c=0.8,

$n_{\bar{X}}=3$ *, R8, PSNR = 30.97dB*

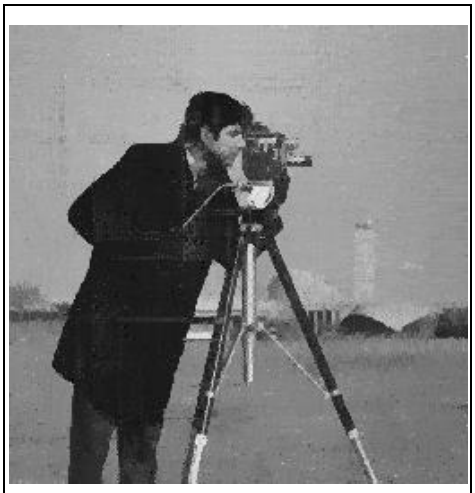
Slika 6.21: camera, PSNR: 28.8dB

Slika 6.22: z_c=1.4, R_c=0.8,

$n_{\bar{X}}=4$ *, R4, PSNR = 29.90dB*

Slika 6.23: z_c=1.5, R_c=0.8,

$n_{\bar{X}}=2$ *, R4, PSNR = 29.96dB*

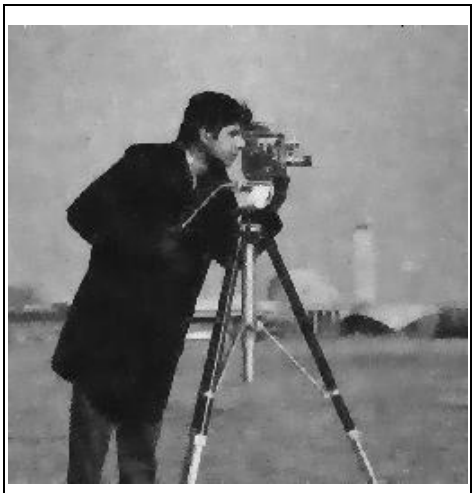


Slika 6.24: z_c=1.6, R_c=0.8,

$n_{\bar{X}}=\infty$ *, R4, PSNR = 29.61dB*

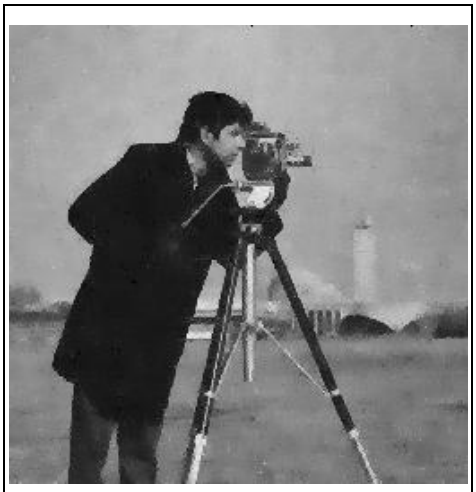
Slika 6.25: z_c=1.6, R_c=0.8,

$n_{\bar{X}}=3$ *, R8, PSNR = 29.93dB*



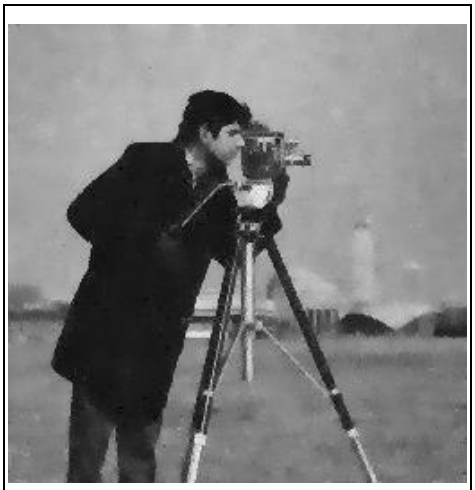
Slika 6.28: z_c=1.6, R_c=0,

$n_{\bar{X}}=3$ *, R4, PSNR = 30.02dB*



Slika 6.26: z_c=1.7, R_c=0.8,

$n_{\bar{X}}=4$ *, R4, PSNR = 30.15dB*

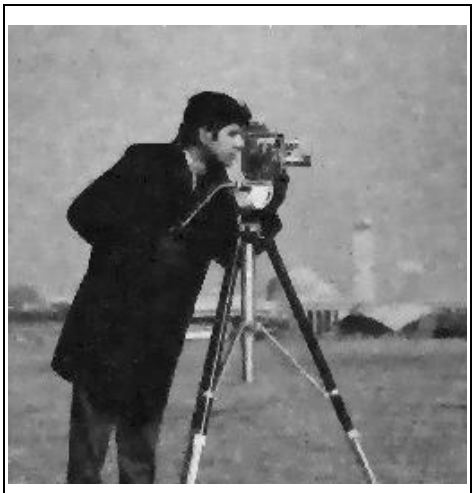


Slika 6.27: z_c=2.0, R_c=0.8,

$n_{\bar{X}}=3$ *, R4, PSNR = 30.11dB*

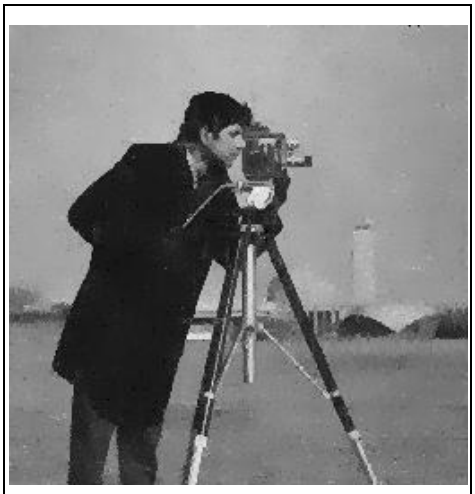
Slika 6.29: z_c=1.9, R_c=0.8,

$n_{\bar{X}}=4$ *, R4, PSNR = 29.89dB*



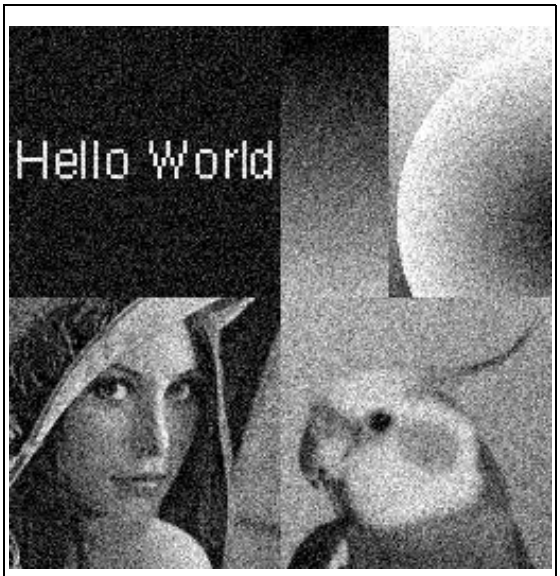
Slika 6.30: z_c=2.0, R_c=0.7,

$n_{\bar{X}}=2$ *, R4, PSNR = 30.14dB*



Slika 6.31: z_c=1.6, R_c=0.8,

$n_{\bar{X}}=8$ *, R4, PSNR = 29.86dB*



Slika 6.32: montage, PSNR: 28.92dB

Slika 6.33: z_c=1.1, R_c=0.85,

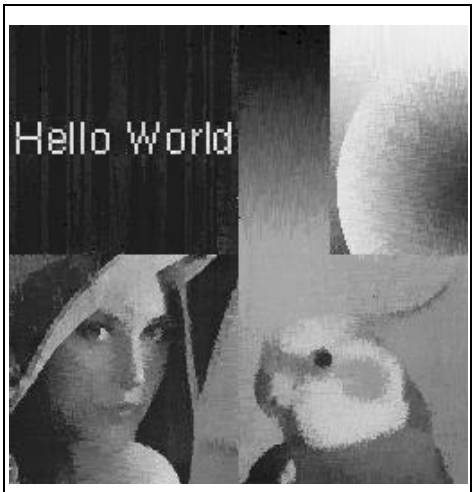
$n_{\bar{X}}=3$ *, R4, PSNR = 28.82dB*

Slika 6.34: z_c=1.5, R_c=0.85,

$n_{\bar{X}}=3$ *, R4, PSNR = 30.40dB*

Slika 6.35: z_c=1.7, R_c=0.85,

$n_{\bar{X}}=3$ *, R4, PSNR = 30.36dB*



Slika 6.36: z_c=1.9, R_c=0.85,

$n_{\bar{X}}=\infty$ *, R4, PSNR = 28.86dB*

Slika 6.37: z_c=1.9, R_c=0.85,

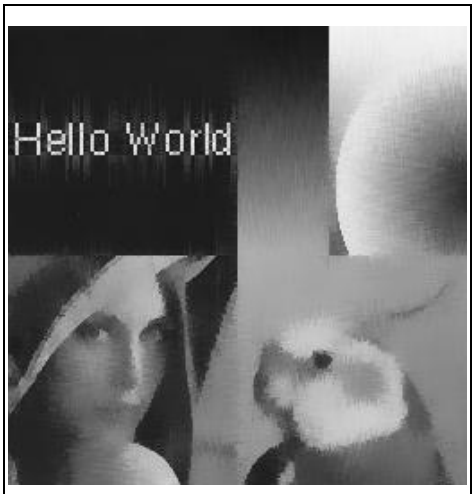
$n_{\bar{X}}=2$ *, R4, PSNR = 28.85dB*

Slika 6.38: z_c=1.9, R_c=0.85,

$n_{\bar{X}}=3$ *, R8, PSNR = 29.96dB*

Slika 6.39: z_c=1.9, R_c=0.85,

$n_{\bar{X}}=3$ *, R4, PSNR = 30.42dB*

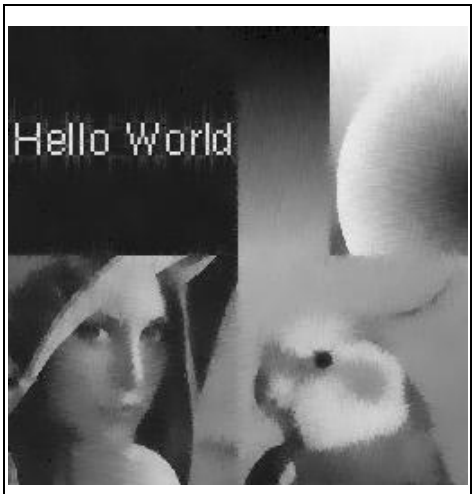


Slika 6.40: z_c=1.9, R_c=0,

$n_{\bar{X}}=3$ *, R4, PSNR = 28.88dB*

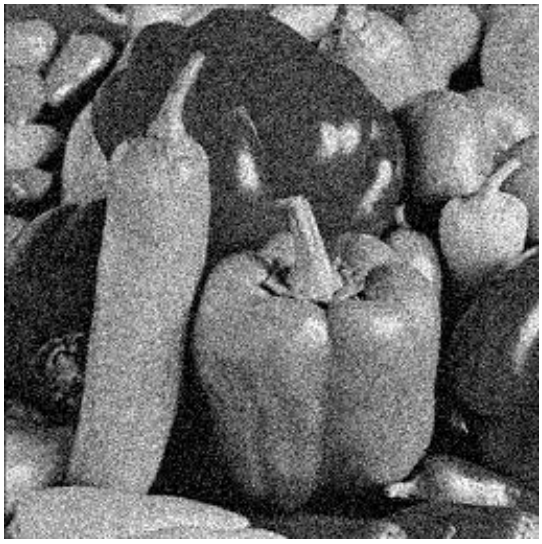
Slika 6.42: z_c=2.3, R_c=0.85,

$n_{\bar{X}}=3$, R4, PSNR = 30.40dB

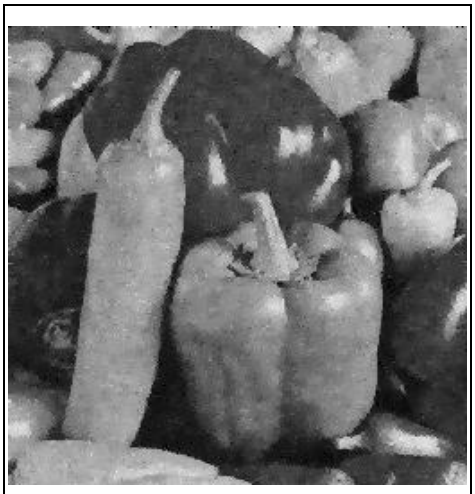


Slika 6.41: z_c=3.0, R_c=0.85,

$n_{\bar{X}}=3$, R4, PSNR = 28.84dB

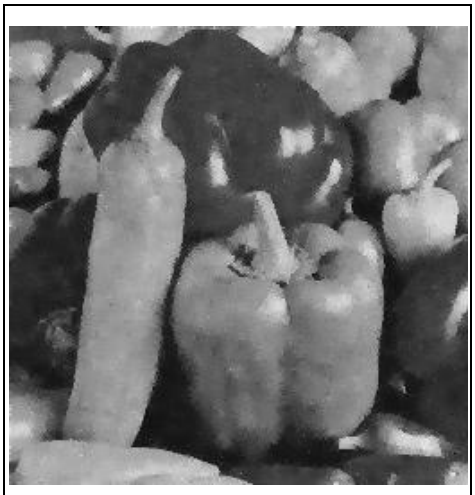


Slika 6.43: peppers, PSNR: 28.78dB



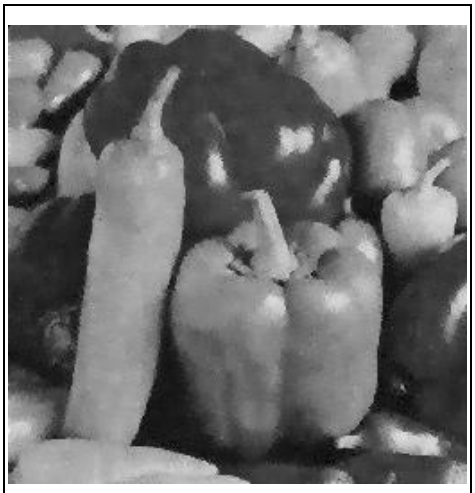
Slika 6.45: z_c=1.1, R_c=0.8,

$n_{\bar{X}}=3$, R4, PSNR = 30.46dB



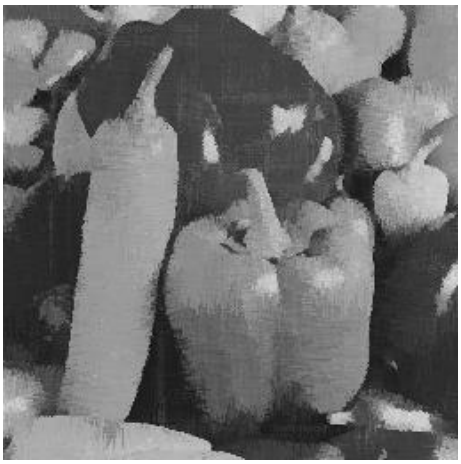
Slika 6.44: z_c=1.5, R_c=0.7,

$n_{\bar{X}}=3$, R4, PSNR = 30.78dB



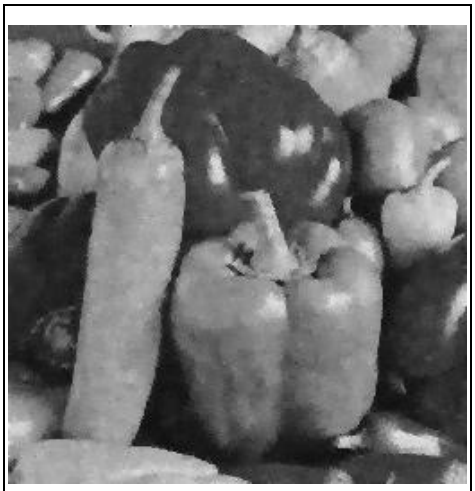
Slika 6.46: z_c=1.7, R_c=0.8,

$n_{\bar{X}}=3$ *, R4, PSNR = 30.40dB*



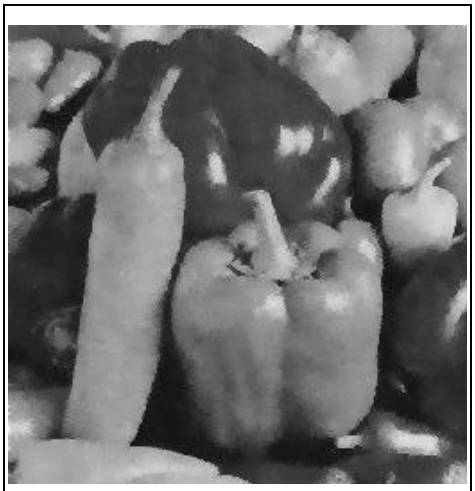
Slika 6.47: z_c=1.8, R_c=0.8,

$n_{\bar{X}}=\infty$ *, R4, PSNR = 29.45dB*



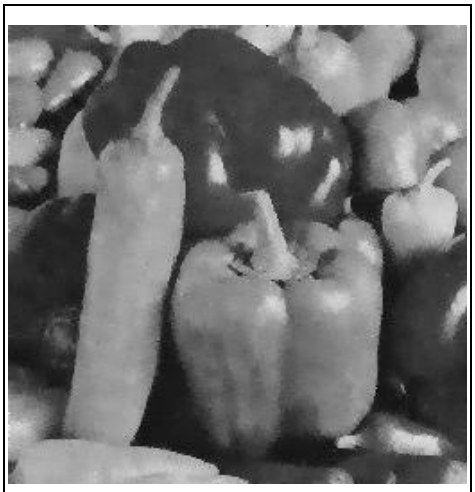
Slika 6.48: z_c=1.8, R_c=0.8,

$n_{\bar{X}}=2$ *, R4, PSNR = 30.97dB*



Slika 6.49: z_c=1.8, R_c=0.7,

$n_{\bar{X}}=3$ *, R4, PSNR = 30.77dB*



Slika 6.50: z_c=1.8, R_c=0.9,

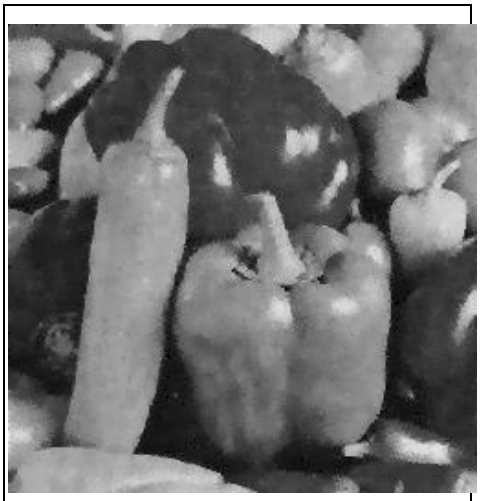
$n_{\bar{X}}=3$ *, R4, PSNR = 30.75dB*

Slika 6.51: z_c=1.8, R_c=0.8,

$n_{\bar{X}}=3$ *, R8, PSNR = 30.42dB*

Slika 6.52: z_c=1.8, R_c=0,

$n_{\bar{X}}=3$, R4, PSNR = 30.47dB



Slika 6.53: z_c=2.0, R_c=0.7,

$n_{\bar{X}}=2$, R4, PSNR = 31.12dB

Slika	z_c	R_c	$n_{\bar{x}}$	R4	R8	PSNR šum (dB)	PSNR odšumljen (dB)
Boat (6.10)	1.7	0.7	3	x		28.79	31.37
Boat (6.10)	1.7	0.8	3	x		28.79	31.41
Boat (6.10)	1.8	0.8	3	x		28.79	31.53
Boat (6.10)	1.9	0.8	2	x		28.79	31.64
Boat (6.10)	1.9	0.8	3	x		28.79	31.46
Boat (6.10)	2	0.7	3	x		28.79	31.41
Boat (6.10)	2	0.9	3	x		28.79	31.47
Boat (6.10)	2	0	3	x		28.79	30.95
Boat (6.10)	1.8	0.8	/	x		28.79	30.64
Boat (6.10)	1.9	0.8	3		x	28.79	30.97
Camera (6.21)	1.5	0.8	2	x		28.8	29.96
Camera (6.21)	1.7	0.8	4	x		28.8	30.15
Camera (6.21)	1.9	0.8	4	x		28.8	29.89
Camera (6.21)	1.4	0.8	4	x		28.8	29.89
Camera (6.21)	1.6	0.8	/	x		28.8	29.61
Camera (6.21)	1.6	0.8	8	x		28.8	29.86
Camera (6.21)	1.6	0.8	3		x	28.8	29.93
Camera (6.21)	1.6	0	3	x		28.8	30.02
Camera (6.21)	2	0.7	2	x		28.8	30.14
Camera (6.21)	2	0.8	3	x		28.8	30.11
Montage (6.32)	1.7	0.85	3	x		28.92	30.36
Montage (6.32)	1.5	0.85	3	x		28.92	30.4
Montage (6.32)	1.9	0.85	3	x		28.92	30.42
Montage (6.32)	2.3	0.85	3	x		28.92	30.4
Montage (6.32)	1.9	0.85	3		x	28.92	29.96
Montage (6.32)	1.9	0.85	2	x		28.92	28.85
Montage (6.32)	1.9	0.85	/	x		28.92	28.86
Montage (6.32)	1.1	0.85	3	x		28.92	28.82
Montage (6.32)	3	0.85	3	x		28.92	28.84
Montage (6.32)	1.9	0	3	x		28.92	28.88
Peppers (6.43)	1.7	0.8	3	x		28.78	30.4
Peppers (6.43)	1.5	0.7	3	x		28.78	30.78
Peppers (6.43)	2	0.7	2	x		28.78	31.12
Peppers (6.43)	1.8	0.7	3	x		28.78	30.77
Peppers (6.43)	1.8	0	3	x		28.78	30.47
Peppers (6.43)	1.8	0.8	3		x	28.78	30.42
Peppers (6.43)	1.1	0.8	3	x		28.78	30.46
Peppers (6.43)	1.8	0.8	/	x		28.78	29.45
Peppers (6.43)	1.8	0.9	3	x		28.78	30.75
Peppers (6.43)	1.8	0.8	2	x		28.78	30.97

Tablica 6.1: Tablični prikaz PSNR vrijednosti u zavisnosti o vrijednosti parametara

Na temelju prikazanih 40 slika kao i pripadnih vrijednosti PSNR, mogu se izvesti određeni zaključci:

- Optimalna vrijednost $\mathbf{z_c}$ nalazi se između 1.8 i 2.1 što odgovara pouzdanostima od 92.8% i 96.4%. Niži iznosi rezultiraju oštrijom slikom, ali ostavljaju više šuma, dok veće vrijednosti imaju za posljedicu veće izglađivanje slike i gubitak detalja.
- Pokazalo se da je u većini slučajeva najbolja vrijednost $R_c = 0.8 \pm 0.5$. Ukoliko se ne koristi RICI nego samo ICI, slike ispadaju osjetno mutnije s većim gubitkom detalja, kao i pojavom artefakata. Korištenje RICI algoritma u odnosu na ICI donijelo je u prosijeku povećanje PSNR od 0.5 dB.
- Kao što je bilo navedeno u 5. poglavlju, izmijenjen način računanja aritmetičkih sredina značajno povećava kvalitetu slika nakon uklanjanja šuma te smanjuje pojavu artefakata što se postiže reduciranjem broja uzoraka $n_{\bar{X}}$ koji se koriste pri izračunu aritmetičkih sredina na 2 ili 3.
- Iako je za očekivati da će osmerokutne regije povećati kvalitetu slika, to nije bio slučaj. Četverokutne regije su dale bolje rezultate kako vizualnim pregledom tako i proračunom PSNR.

Ukupno gledano, RICI algoritam je, ovisno o slikama, nakon uklanjanja šuma povećao PSNR od 1dB do 2.5dB.

7. Usporedba RICI algoritma s drugim metodama za uklanjanje šuma

7.1 Metode koje koriste algoritam presjecišta intervala pouzdanosti

7.1.1 *A spatially adaptive nonparametric regression image deblurring, Katkovnik, V.Egiazarian, K. Astola, J., 2005, Image Processing [11]*

U radu "Odmagljivanje slike prostornom adaptivnom neparametarskom regresijom" opisan je postupak koji koristi algoritam baziran na lokalnoj aproksimaciji polinoma (*Local Polynomial Approximation* - LPA) kao i metodi presjecišta intervala pouzdanosti (ICI) koja služi za određivanje adaptivne promjenjive veličine otvora LPA estimatora. LPA-ICI algoritam nije linearan i prostorno adaptivan s obzirom na zaglađenost i nepravilnosti slike kao posljedice šuma. ICI algoritam daje optimalnu veličinu otvora za svaki element slike.

7.1.2 *Adaptive window size image denoising based on ICI rule, Egiazarian, K. Katkovnik, V. Astola, L., 2001., Acoustics, Speech, and Signal Processing [12]*

Članak s naslovom "Uklanjanje šuma iz slika otvorom adaptivne veličine temeljenim na ICI metodi" opisuje postupak uklanjanja šuma baziran na lokalno adaptivnoj veličini otvora. Metodu odlikuju dva bitna svojstva. Prvo, uklanja se ne samo Gaussov bijeli šum, već šum koji ovisi o slici (npr. *film-grain* ili *multiplicative* šum). Drugo, za transformirnu vrijednost parametara određujemo otvor promjenjive veličine metodom presjecišta intervala pouzdanosti. Na kraju, nova se vrijednost parametra određuje težinskim prosjekom susjednih parametara za koje dolazi do presijecanja intervala pouzdanosti.

7.1.3. *Local adaptive transform based image denoising with varying window size, Oktem, H. Katkovnik, V. Egiazarin, K. Astola, J., 2001, Image Processing [13]*

Članak "Uklanjanje šuma otvorom promjenjive veličine temeljenim na lokalnoj adaptivnoj transformaciji" opisuje učinkovit način uklanjanja zamagljenosti slike koji koristi lokalno adaptivno uklanjanje šuma, a, za razliku od transformacije fiksnom veličinom otvora, optimalnu veličinu otvora računa koristeći metodu presjecišta intervala pouzdanosti.

7.2 Ostale metode uklanjanja šuma sa slike

Postoje dva osnovna pristupa filtriranju šuma iz slike, prostorni filtri i filtri domeni transformacije[14].

I. Prostorno filtriranje

Uobičajeni način za uklanjanje šuma iz slike je korištenje prostornih filtera koji se dalje mogu klasificirati na nelinearne i linearne filtre [15].

a) Nelinearni filtri

Korištenjem nelinearnih filtera šum se uklanja bez prethodnog pokušaja identificiranja. Na grupu slikovnih elemenata primjenjuje se niskopropusni filter pod pretpostavkom da šum zahvaća viša područja frekvencijskog spektra. Općenito, prostorni filtri uklanjaju šum do određene granice uz neželjenu pojavu zamućenja slike zbog kojega rubovi u slici postaju nevidljivi. U novije vrijeme pojavio se niz filtera baziranih na *median* funkciji [15] koji za cilj imaju ukloniti taj nedostatak.

b) Linearni filtri

Usrednjavajući (eng. *mean*) filtri su optimalni filtri za Gaussov šum kada je riječ o srednjoj kvadratnoj pogrešci. Linearni filtri također imaju za posljedicu zamagljivanje oštrih rubova, uništavanje linija i drugih finih detalja te imaju loše rezultate u prisustvu šuma ovisnog o signalu. Za Wienerov filter nužna je informacija o spektru šuma kao i originalnom signalu i daje dobre rezultate samo ako je signal gladak. Wienerova metoda implementira prostorno zaglađivanje koje ovisi o veličini otvora. Da bi uklonili nedostatke Wienerovog filtera Donoho i Johnstone su predložili uklanjanje šuma bazirano na valićima [16].

II. Filtri u domeni transformacije

Filtri u domeni transformacije mogu se dalje podijeliti na adaptivne i neadaptivne metode[15]. Neadaptivne metode će biti obrađene prve jer su trenutno popularnije.

a) Prostorno-frekvencijsko filtriranje

Prostorno-frekvencijsko filtriranje odnosi se na korištenje niskopropusnog filtra upotrebom brze Fourieove transformacije (eng. *Fast Fourier Transform* - FFT). Kod frekvencijski zaglađujućih metoda [17], uklanjanje šuma postiže se dizajniranjem filtra u frekvencijskoj domeni i prilagodbi gornje granične frekvencije kada frekvencijske kompontente šuma nisu u korelaciji s korisnim signalom. Ove metode su vremenski

zahtjevne, ovise o graničnoj frekvenciji te o ponašanju filtra. Nadalje, mogu proizvesti umjetne frekvencijske komponente u procesiranoj slici.

b) Domena valića (eng. *Wavelet domain*)

Operacije filtriranja u domeni valića (eng. *wavelet*) mogu biti podijeljene na linearne i nelinearne metode.

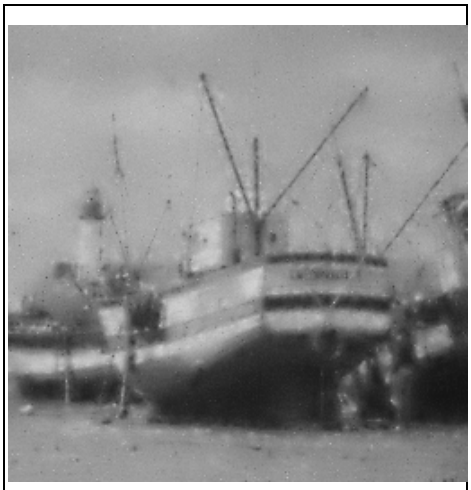
1. **Linearni filtri**

   Linearni filtri, kao što je Wienerov filtar u domeni valića daje optimalne rezultate kada šum može biti modeliran kao Gaussov proces, a kriterij za preciznost je srednja kvadratna pogreška (MSE) [18]. Međutim, filter dizajniran na ovoj pretpostavci često rezultira sa slikama koje su vizualno manje zadovoljavajuće u odnosu na originalni zašumljeni signal, bez obzirao što rezultirajuća slika ima niži MSE. U [19] predložen je prostorno adaptivni FIR Wienerov filtar u domeni valića.

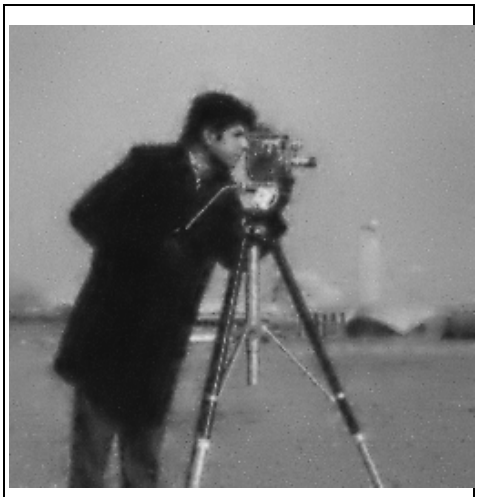
2. **Filtriranje nelinearnim pragom**

   Najistraživanije područje vezano za uklanjanje šum u domeni valića predstavljaju metode bazirane na **nelinearnim koeficijentima pragova**. Postupak koristi prorijeđenosti kod transformacije u domenu valića kao i činjenicu da transformacija pridružuje bijeli šum u prostornoj domeni signala s bijelim šumom u domeni transformacije. Tako, za razliku od energije šuma, energija signala postaje sve koncentriranija u manje koeficijenata. Ovaj bitan princip omogućuje separaciju signala od šuma. Metoda može koristiti adaptivne i neadaptivne pragove.

   Jedna od tih je i metoda korištena za "*wavelet denoise plugin*" u "GNU Image Manipulation Programu" (GIMP) [20]. Na slikama 7.1-7.4. dani su rezultati uklanjanja šuma iz slika koje su prethodno bile korištene za demonstraciju RICI metode uz parametre *treshold=0.6* i *softness=0.1*. Iako su dobivene PSNR vrijednosti približno jednake vrijednostima dobivenim RICI metodom, vizualno slike se znatno razlikuju. Slike su zaglađenije, rubovi su "razmazani", a također se mogu primijetiti i ostaci šuma na pojedinim dijelovima slika.

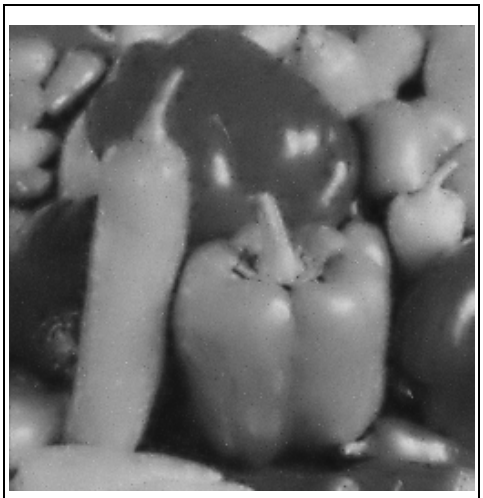


Slika 7.2: PSNR: 30.74dB



Slika 7.1: PSNR: 30.53dB

Slika 7.4: PSNR: 30.36dB



Slika 7.3: PSNR: 30.14dB

3. Neortogonalna transformacija u domenu valića

Nedecimirana wavelet transformacija (eng. *Undecimated Wavelet Transform* - UDWT) se također koristi za dekompoziciju signala kako bi se osiguralo bolje vizualno rješenje. Kako je UDWT nepromjenjiva s obzirom na pomak, izbjegnuto je pojavljivanje vizualnih artefakata. Iako UDWT rezultira značajnim poboljšanjima, zbog povećanih zahtjeva za računalnom snagom, ova metoda je manje upotrebljiva [21].

4. **Model koeficijenta valića**

   Ovaj se pristup fokusira na iskorištavanje multirezolucijske karakteristike *wavelet* transformacije. Ova tehnika identificira usku korelaciju signala na različitim rezolucijama promatrajući signal kroz višestruke rezolucije. Modeliranje valića može biti determinističko [22] ili statističko [23]. Metoda izvrsne rezultate, ali je računalno vrlo zahtjevna i skupa.

c) **Podatkovno adaptivne transformacije**

U novije vrijeme pojavila se nova metoda zvana **nezavisna analiza komponenti** (eng. *Independent component analysis* - ICA) koja je pobudila širu pozornost. ICA metoda je uspješno implementirana u [24] za uklanjanju ne-Gaussovog šuma. Iznimna odlika ICA metode je njena pretpostavka da je šum ne-Gaussov, što pomaže ukloniti šum sa slika kako s ne-Gaussovim tako i sa Gaussovim šumom. Nedostatak ICA metode, u usporedbi s metodama baziranim na valićima, je njena računalna zahtjevnost jer koristi klizeći otvor i zahtjeva uzorak signala bez šuma ili bar dvije slike iste scene. U nekim je slučajevima teško dobiti takve bešumne "trening" podatke.

8. Zaključak

Kako je šum najčešće neželjena, ali i neizbježna pojava, koja svojom prisutnošću narušava kvalitetu signala, odnosno, u našem slučaju slike, njeno uklanjanje ili bar reduciranje je od velike važnosti. Kao što je u prethodnom poglavlju prikazano, postoje mnoge metode kojima se to postiže, a najveći broj metoda koje se trenutno koriste baziran je na valićima.

U ovom je radu opisana nova metoda koja koristi lokalni prostorno adaptivni algoritam u kombinaciji pravilom presjecišta intervala pouzdanosti koji se odlikuje očuvanjem rubova u slici. Nadalje, ICI pravilo je bilo modificirano dodavanjem još jednog uvjeta kako bi bilo osjetljivije na skokove u signalu što je rezultiralo preciznijim estimatima smanjujući tako i srednju kvadratnu pogrešku. Novo RICI pravilo je ujedno i manje numerički zahtjevno od originalne metode. Za razliku od mnogih popularnih postupaka baziranih na valićima, slike filtrirane RICI metodom znatno su oštrije te sa bolje očuvanim rubovima, a samim time pogodnije za daljnju obradu.

Nedostatak metode je sporost algoritma, odnosno visoki zahtjevi za procesorskom snagom. Stoga je prethodno potrebno optimizirati algoritam prije nego može postati upotrebljiv u praksi. Nadalje, ostavljen je prostor za daljnja poboljšanja algoritma, posebno na području detekcije regija koje je moguće bitno preciznije odrediti, a time i dodatno povećati kvalitetu filtrirane slike.

Prilikom odabiranja algoritma za smanjenje šuma treba obratiti pažnju na više stvari. Bitan faktor je dostupna računalna snaga kao i vrijeme za postupak filtriranja. Tipična digitalna kamera ima slab ugrađeni procesor i svega dio sekunde za obradu slike. S druge strane, prosječno kućno računalo je mnogo snažnije i ima više vremena na raspolaganju. Nadalje, što je razina uklanjanja šuma veća, to će biti veći gubitak detalja na slici pa se postavlja pitanje gdje povući granicu. Na odluku o odabiru algoritma također utječu i karakteristike šuma kao i sam sadržaj slike.

9. Literatura

[1] A. McAndrew, "An Introduction to Digital Image Processing with Matlab", School of Computer Science and Mathematics Victoria University of Technology, 2004, p. 13.

[2] Ibid. pp. 14-20.

[3] USC-SIPI Image Database - Miscellaneous [Online], Available: http://sipi.usc.edu/database/database.cgi?volume=misc [Accessed: Sept., 2009]

[4] R. A. Schowengerdt, "Image Noise" [Online], Available: http://www.dig.cs.gc.cuny.edu/seminars/IPCV/pres12.pdf [Accessed: Sept., 2009]

[5] M. R. Spiegel, J. Schiller, and R. A. Srinivasan, " Schaum's Outline of Probability and Statistics", McGraw-Hill, 2001, p. 76

[6] Ibid. pp. 45-46

[7] Ibid. pp. 76-77

[8] Ibid. p. 78

[9] J. Lerga, M. Vrankić and V. Sučić, "A Signal Denoising Method Based on the Improved ICI Rule", IEEE Signal Processing Letters, vol. 15, pp. 601-602, May 2008.

[10] R.C. Gonzales, R.E. Woods, Digital Image Processing 2ed, Prentice Hall, 2002, pp. 419-420.

[11] Katkovnik, V., Egiazarian, K., Astola, J., "A spatially adaptive nonparametric regression image deblurring", Image Processing, IEEE Transactions on, vol. 14, Issue: 10, pp. 1469-1478, Oct. 2005.

[12] Egiazarian, K., Katkovnik, V. Astola, L., "Adaptive window size image denoising based on ICI rule", Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2001 IEEE International Conference on, Vol. 3, pp. 1869-1872.

[13] Oktem, H., Katkovnik, V., Egiazarin, K., Astola, J., "Local adaptive transform based image denoising with varying windowsize", Image Processing, 2001. Proceedings. 2001 International Conference, vol. 1, pp. 273-276.

[14] M. Motwani, M. Gadiya, R. Motwani, and Frederick C. Harris, Jr., "A Survey of Image Denoising Techniques" , Proceedings of GSPx, Sept. 2004.

[15] R. Yang, L. Yin, M. Gabbouj, J. Astola, and Y. Neuvo, “Optimal weighted median filters under structural constraints,” IEEE Trans. Signal Processing, vol. 43, pp. 591–604, Mar. 1995.

[16] David L. Donoho and Iain M. Johnstone., “Adapting to unknown smoothness via wavelet shrinkage”, Journal of the American Statistical Association, vol.90, no432, pp.1200-1224, December 1995. National Laboratory, July 27, 2001.

[17] A.K.Jain, Fundamentals of digital image processing. Prentice-Hall,1989

[18] V. Strela. “Denoising via block Wiener filtering in wavelet domain”. In 3rd European Congress of Mathematics, Barcelona, July 2000. Birkhäuser Verlag.

[19] H. Zhang, Aria Nosratinia, and R. O. Wells, Jr., “Image denoising via wavelet-domain spatially adaptive FIR Wiener filtering”, in IEEE Proc. Int. Conf. Acoust., Speech, Signal Processing, Istanbul, Turkey, June 2000.

[20] Wavelet denoise | GIMP plugin registry [Online], Available: http://registry.gimp.org/node/4235 [Accessed: Sept., 2009]

[21] M. Lang, H. Guo, J.E. Odegard, and C.S. Burrus, "Nonlinear processing of a shift invariant DWT for noise reduction," SPIE, Mathematical Imaging: Wavelet Applications for Dual Use, April 1995.

[22] R. G. Baraniuk, “Optimal tree approximation with wavelets,” in Proc. SPIE Tech. Conf. Wavelet Applications Signal Processing VII, vol. 3813, Denver, 1999, pp. 196-207.

[23] R. W. Buccigrossi, and E. P. Simoncelli, ‘Image compression via joint statistical characterization in the wavelet domain’, IEEE Image Process., Vol. 8, No 12, Dec.1999, pp. 1688-1701.

[24] A. Jung, “An introduction to a new data analysis tool: Independent Component Analysis”, Proceedings of Workshop GK "Nonlinearity" - Regensburg, Oct. 2001.

10. Dodaci

10.1 MATLAB kôd za proračun regije i estimirane vrijednosti svakog slikovnog elementa

```
clc; close all hidden;
im = imread('camera.tif');
std_dev = 25;
noise = int16(randn(size(im))*std_dev);
im_noised = uint8(int16(im)+noise);
%imnoise(im, 'gaussian', 0, 0.002);
%imtool(im);
%imtool(im_noised);
%imtool(im_noised-im);
Zc = 1.8;
Rc = 0.8;
draw_graphs_around = 0;
use_RICI = 1;
n_average = 3;
filtered_signal = [];
% Start from
n = 1;
rows = size(im_noised, 1);
cols = size(im_noised, 2);
new = ones(size(im_noised), 'uint8').*255;

for x = 1:rows
    for y = 1:cols
        val = double(im_noised(x,y));

        noisy_signal = im_noised(x,:);
        [left, right] = ICI_1D(noisy_signal, val, Zc, Rc, y,...
            use_RICI, n_average, draw_graphs_around, std_dev, x, y);
        noisy_signal = im_noised(:,y);
        [top, bottom] = ICI_1D(noisy_signal, val, Zc, Rc, x,...
            use_RICI, n_average, draw_graphs_around, std_dev, x, y);

        noisy_signal = diag(im_noised, y-x);
        if y>x
            axis = x;
        else
            axis = y;
        end
```

```
%       Octogonal region
%         [topleft, bottomright] = ICI_1D_part(noisy_signal, val, Zc, Rc,...
%            axis, use_RICI, n_average, draw_graphs_around, std_dev, x, y);
%         d1 = axis - topleft;
%         d2 = bottomright - axis;
%
%         noisy_signal = diag(fliplr(im_noised), y-x);
%         [topright, bottomleft] = ICI_1D_part(noisy_signal, val, Zc, Rc,...
%            axis, use_RICI, n_average, draw_graphs_around, std_dev, x, y);
%         d3 = axis - topright;
%         d4 = bottomleft - axis;
%
        % New estimated value of the current signal sample is average of the
        % values in the [from, to] interval of the signal
        if top > 1
            top = top - 1;
        end
        if right < cols
            right = right + 1;
        end
        if bottom < rows
            bottom = bottom + 1;
        end
        if left > 1
            left = left - 1;
        end
%         Octogonal region
%         mask = roipoly(im_noised,...
%            [y, y+d3+1, right, y+d2+1, y, y-d4-1, left, y-d1-1], ...
%            [top, x-d3-1, x, x+d2+1, bottom, x+d4+1, x, x-d1-1]);

        mask = roipoly(im_noised, [y, right, y, left], [top, x, bottom, x]);
        region = im_noised.*uint8(mask);
        values = region(region>0);
        new(x,y) = mean(values);
    end
    x
end
```

10.2 MATLAB funkcija koja računa 1D RICI za svaki slikovni element

```
function [start, stop] = ICI_1D(noisy_signal, val, Zc, Rc, axis, use_RICI,
n_average, draw_graphs_around, std_dev, x, y)
    noisy_signal = uint16(noisy_signal);
    n=axis;

    % Initialize arrays which will hold intervals on the left
    % and on the right of the current signal sample for which
    % we are doing estimation
    avrs_left = [];
    avrs_right = [];
    intervals_lo_left = [];
    intervals_lo_right = [];
    intervals_hi_left = [];
    intervals_hi_right = [];

    % Interval for the current signal sample
    d = Zc*std_dev;
    l = val - d;
    h = val + d;
    sum_left = val;
    sum_right = val;

    % Go left of the current signal sample until confidence intervals
    % stop intersecting or we hit the start of the signal
    i = n - 1;
    while i > 0
        % Compute average of the signal samples from the interval
        % [n-i+1, n-1], that is, from the current sample on the left. to the
        % sample for which confidence intervals stop intersecting
        sum_left = sum_left + noisy_signal(i);
        avr_left = sum_left/(n-i+1);

        % Compute average of the signal samples from the interval
        % [n-i+1, n-i+n_average]. Same as above but use max of
        % n_average samples when calculating average
        if n_average
            if n-i >= n_average
                sum_left = sum_left - noisy_signal(i+n_average);
                avr_left = sum_left/n_average;
            end
        end
```

```
    % Calculate confidence interval for samples on the left of the
    % current sample
    d = Zc*std_dev/((n-i+1)^(1/2));
    interval_lo = avr_left - d;
    interval_hi = avr_left + d;

    % Find max lower interval on the left
    L_max = max([intervals_lo_left interval_lo l]);

    % Find min upper interval on the left
    U_min = min([intervals_hi_left interval_hi h]);

    % ICI and RICI criterions, exits loop if true
    if use_RICI
        Rk = (U_min - L_max)/(2*d);
        if (L_max > U_min) | Rk < Rc
            break;
        end

    % Only ICI
    else
        if L_max > U_min
            break;
        end
    end

    % If there is intersection prepend intervals to existing arrays
    intervals_lo_left = [interval_lo intervals_lo_left];
    intervals_hi_left = [interval_hi intervals_hi_left];
    avrs_left = [avr_left avrs_left];

    % Next sample on the left
    i = i - 1;
end

% This is first sample (leftmost) of the interval which will be used to
% compute estimated value
start = i + 1;

% Go left from the current signal sample until confidence intervals
% stop intersectiong or we hit start of the signal
i = n + 1;
while i <= length(noisy_signal)
    % Compute average of the of the signal samples from the interval
    % [n-i+1, n-1], that is, from the current sample on the left to the
```

```
% sample for which confidence intervals stop intersecting
sum_right = sum_right + noisy_signal(i);
avr_right = sum_right/(i-n+1);

% Compute average of the signal samples from the interval
% [n-i+1, n-i+n_average]. Same as above but use max of
% n_average samples when calculating average
if n_average
    if i-n >= n_average
        sum_right = sum_right - noisy_signal(i-n_average);
        avr_right = sum_right/n_average;
    end
end

% Calculate confidence interval for samples on the left of the
% current sample
d = Zc*std_dev/((i-n+1)^(1/2));
interval_lo = avr_right - d;
interval_hi = avr_right + d;

% Find max lower interval on the left
L_max = max([l intervals_lo_right interval_lo]);

% Find min upper interval on the left
U_min = min([h intervals_hi_right interval_hi]);

% ICI and RICI criterions, exits loop if true
if use_RICI
    Rk = (U_min - L_max)/(2*d);
    if (L_max > U_min) | Rk < Rc
        break;
    end

% Only ICI
else
    if L_max > U_min
        break;
    end
end

% If there is intersection prepend intervals to existing
% arrays
intervals_lo_right = [intervals_lo_right interval_lo];
intervals_hi_right = [intervals_hi_right interval_hi];
avrs_right = [avrs_right avr_right];
```

```
    % Next sample on the right
    i = i + 1;

end

% This is last sample (rightmost) of the interval which will be used to
% compute estimated value
stop = i - 1;
%[from, to];
if draw_graphs_around
    % Concatenate left and right
    lower_intervals = [intervals_lo_left l intervals_lo_right];
    upper_intervals = [intervals_hi_left h intervals_hi_right];
    avrs = [avrs_left val avrs_right];

    figure;
    errorbar(start:stop, avrs, avrs-lower_intervals,...
         avrs-lower_intervals, 'b.'), hold on
    %plot(n, est_val, 'kx'), hold on
    errorbar(n, val, Zc*std_dev, Zc*std_dev, 'mo', 'LineWidth', 2), hold on
    plot(start:stop, noisy_signal(start:stop), 'rx'), hold on
    grid minor
end
```