SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVOD ZA ELEKTRONIČKE SUSTAVE I OBRADBU INFORMACIJA

Primjena 2D wavelet transformacije u kompresiji slike u bežičnim senzorskim mrežama

Projekt iz kolegija *Napredne metode digitalne obradbe signala*

Anita Ivković 0036414945

Vana Jeličić 0036411639

Ana Svirčić 0036416838

Zagreb, siječanj 2009.

Sadržaj

1. Uvod

U bežičnim senzorskim mrežama javlja se sve veća potreba za akvizicijom slike i videa. Potrebne su sofisticirane metode obrade i prijenosa slike koje zadovoljavaju nisku potrošnju i jednostavnost izvedbe u sklopovlju. Navedeni zahtjevi su najčešće u recipročnom odnosu pa treba napraviti kompromis. Postoje grupe znanstvenika u svijetu (i veće i manje) koje proučavaju i predlažu nove metode koje bi što bolje zadovoljile njihove potrebe. Ipak, to je još uvijek novo područje te nije prisutna standardizacija.

U okviru ovog rada proučeni su kompresijski postupci pogodni za implementaciju u bežičnim senzorskim mrežama. Izabran je algoritam SPIHT, temeljen na DWT (diskretnoj wavelet transformaciji). Zbog svoje jednostavnosti izabran je Haarov wavelet. Postupak je implementiran i testiran u programskom paketu Matlab.

2. Kompresijski postupci – zahtjevi i ograničenja kod primjene u bežičnim senzorskim mrežama

U kompresiji slike danas su u svijetu dominantna dva standarda – JPEG i JPEG2000. JPEG se temelji na diskretnoj kosinusnoj transformaciji (DCT), a JPEG2000 je njegova mlađa verzija temeljena na diskretnoj wavelet transformaciji (DWT). Zbog brojnih prednosti sve više se koristi JPEG2000 i možemo reći da je u većini primjena gotovo istisnuo JPEG. No, i JPEG i JPEG2000 zahtijevaju mnogo energije i procesorske moći da bi brzo i efikasno komprimirali i dekomprimirali sliku. Zato to nisu prihvatljiva rješenja za sustave koji nemaju moćne procesore i koji su baterijski napajani. Jedan od takvih sustava je bežična senzorska mreža.

2.1. Akvizicija i obrada slike u bežičnim senzorskim mrežama

Bežične senzorske mreže su multidisciplinarna tehnologija u koju se danas mnogo ulaže i od koje se očekuju koristi u mnogim dijelovima ljudske djelatnosti. Smatraju se jednom od najvažnijih tehnologija 21. stoljeća. Njihova trenutna primjena i istraživanja uključuju: vojno senziranje, nadgledanje prometa, nadzor zgrada, automatizaciju u proizvodnji i industriji, robotiku, promatranje okoliša i staništa...

U posljednje vrijeme se sve više javlja potreba i želja za prijenosom informacije iz okoliša u obliku slike ili čak i videa. To stvara dodatne probleme zbog vrlo velikog prijenosa podataka koji troši jako puno energije i frekvencijskog pojasa i smanjuje životni vijek uređaja.

Na Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu u tijeku je razvojno-istraživački projekt MasliNet u okviru kojega se razvija bežična senzorska mreža za primjenu u uzgoju maslina. Mreža se sastoji od nekoliko bežičnih osjetilnih čvorova smještenih na tlu i u krošnji koji senzorima prikupljaju podatke iz okoline i omogućuju udaljenom promatraču (agronomu ili vlasniku maslinika) praćenje mikroklimatskih uvjeta. Osim senzora za temperaturu, tlak i vlagu, neki čvorovi imat će i kameru. Postoji potreba za snimanjem maslinine muhe (najčešćeg nametnika na maslini za kojeg se rade posebne lovke), za što se razvija modul s kamerom razlučivosti 2 megapiksela, te za snimanjem stanja stabla masline (odnosno fenofaze – da li maslina pupa, cvjeta i sl.), za što se razvija modul s kamerom VGA razlučivosti (480x640 piksela).

Čvorovi su u masliniku međusobno povezani u lokalnu mrežu bežičnim protokolom kratkog dometa (ZigBee). Jedan čvor je jače procesorske snage i predstavlja 'lokalni server' te je na sljedećem komunikacijskom sloju putem javne mreže (GPRS/GSM) povezan sa središnjim serverom gdje se podaci prikupljaju, pohranjuju i obrađuju. U slučaju pojave nametnika, na osnovi prikupljenih informacija pokreću se alarmi ili se pak pojava nametnika može unaprijed predvidjeti na nekoliko lokacija te na temelju njih donijeti odluku o tome treba li, koje i koliko pesticida koristiti na nekom mjestu. [3]

ZigBee protokol, koji se koristi za slanje podataka sa senzora na server, pogodan je za slanje male količine podataka uz malu potrošnju. Nije zamišljen za slanje velikih datoteka kao što je slika dobivena na senzorima slike visoke razlučivosti. Zato je osmišljeno razlaganje velike datoteke (tj. slike) na više manjih paketa koji se onda šalju ZigBee protokolom.

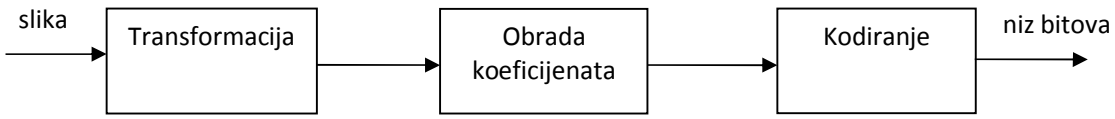
Mnoga su istraživanja napravljena u ovom području da bi se pronašle optimalne kombinacije obrade podataka dobivenih na senzorima (energija potrošena na instrukcije mikrokontrolera) i slanja podataka bežičnim putem (energija potrošena na emitiranje podataka). Istraživanja pokazuju da je energija potrebna za bežičnu komunikaciju mnogo veća od energije potrebne za izvođenje instrukcije na samom čvoru (obrada podatka).

Budući da slika po prirodi obiluje redundancijom, nju je potrebno komprimirati i tako komprimiranu slati je na server. Ipak, složeni postupci kompresije zahtijevaju izvođenje složenih algoritama što nije pogodno za implementaciju na čvoru napajanom baterijom, sa zahtjevom na što veću autonomiju, koji bi, uz to, u konačnici trebao biti i jeftin. U projektu MasliNet zamišljeno je implementirati kompresijski postupak na FPGA skolopovlju.

Zadatak je pronaći što jednostavniji kompresijski postupak koji ne troši puno energije, a ipak daje dekomprimiranu sliku koja zadovoljava naše zahtjeve.

2.3. Kompresijski postupci – prednosti i nedostaci

Općenito se kompresijski postupak sastoji od tri dijela: transformacije, metode obrade koeficijenata (koji najčešće obuhvaća kvantizaciju koja je uzrok gubitaka) i kodiranja (Slika 1). Postoje mnoge inačice svakog od njih, a s time i mnogo kombinacija dizajniranja kompresijskih postupaka koji zadovoljavaju razne zahtjeve.



Slika 1: Blok shema kompresijskog postupka

Proučavanjem literature pronađene su neke metode kompresije temeljene na DCT-u ili DWT-u koje su jednostavnije od JPEG-a i JPEG2000, te njihove preinake namijenjene implementaciji na sklopovlju. Uspoređivanjem njihovih dobrih i loših strana, kao i zahtjeva u našoj primjeni, donesen je zaključak da bi kompresija temeljena na wavelet transformaciji bila bolji izbor od kosinusne transformacije. Wavelet transformacija daje bolje rezultate za ljudsko oko – osigurava veću vremensku razlučivost za visoke frekvencije, a visoku frekvencijsku razlučivost za niske frekvencije. Omogućava i progresivno kodiranje. To znači da se svim koeficijentima slike prvo kodiraju najviši bitovi pa se onda postupno kodiraju niži bitovi čime se povećava kvaliteta rekonstruirane slike. Upravo to omogućava i skalabilnost, što je vrlo korisno jer se u bilo kojem trenutku može prekinuti s dekodiranjem, a ipak će se rekonstruirati cijela slika (doduše, ne cijela s jednakom kvalitetom, ali i to ponekad bude dovoljno za određene primjene). Važna prednost je i multirezolucija, tj. moguće je rekonstruirati sliku za razne vrijednosti rezolucije manje od originalne. [8]

Postoje mnogi filtri korišteni u wavelet transformacijama, ali je zbog sklopovskih ograničenja odabran Haarov wavelet. On je najjednostavniji od svih waveleta i za njegovu implementaciju potrebno je samo zbrajanje, oduzimanje i množenje (tj. dijeljenje) s 2 koje se izvodi kao pomak (*shift*), što je vrlo korisno jer je poznato da operacija množenja troši puno više energije od zbrajanja.

Nakon izbora wavelet transformacije trebalo je izabrati tehniku obrade koeficijenata. To je algoritam, tj. način na koji se prolazi kroz matricu nastalu DWT-om i obrađuju koeficijenti i, na kraju, formira izlazni niz bitova koji predstavlja komprimiranu sliku.

Već je gore spomenuto da je JPEG2000 presložen za naše potrebe. On koristi EBCOT kompresijski postupak koji dosta složeno obrađuje koeficijente slike po razinama bitova (*bit-plane*).

Prekretnicu u tvrdnji da samo jako složeni kompresijski postupci mogu dati kvalitetnu rekonstruiranu sliku napravio je Shapiro 1993. godine sa svojim EZW (*Embedded Zerotree Algorithm*) temeljenim na *zerotree* tehnici (pronalaženje beznačajnih skupova koeficijenata). Poboljšanu verziju EZW-a donose Said i Pearlman 1996. god., pod nazivom SPIHT (*Set Partitioning in Hierarchical Trees*). Proučavanjem

SPIHT-a otkriveno je da posljednja etapa u kompresiji (kodiranje) kod SPIHT-a ne donosi značajna poboljšanja u rezultatu te se može slobodno, pogotovo u primjeni ograničenoj energijom, memorijom i složenošću (a naša to definitivno jest), izostaviti. [4]

3. Diskretna wavelet transformacija – Haarov wavelet

Kao što je već prije rečeno, pri izboru metode kompresije slike treba uzeti u obzir ograničenja pri potrošnji memorije i energije. Kao transformaciju smo odabrali Haarov wavelet, i to u ljestvičastoj (*lifting)* izvedbi jer se sastoji od jednostavnih instrukcija – zbrajanja, oduzimanja i množenja s 2 (što je zapravo pomicanje ulijevo). Jednostavne instrukcije troše manje procesorske moći i vremena, a time i manje energije. Također, ova metoda je *in place* što znači da ne zahtijeva dodatnu memoriju jer se nakon zbrajanja i oduzimanja elemenata matrice rezultat zapisuje na isto mjesto u memoriji.

3.1. Haarova 1-D wavelet transformacija

Da pojasnimo kako rade waveleti, uzet ćemo jedan jednostavan primjer. Kao što smo već više puta naglasili, Haarov wavelet odlikuje se iznimnom jednostavnošću i zato ga je moguće realizirati na više načina. Najčešća izvedba koristi usrednjavanje piksela za računanje aproksimacijskih koeficijenata. [7], [9]

Zbog potrebe implementacije transformacije na jednostavnom sklopovlju, želimo izbjeći dijeljenje s 2 i frakcionalnu aritmetiku u postupku dekompozicije, te koristimo malo izmjenjenu izvedbu.

Pretpostavimo da imamo jednodimenzionalnu sliku rezolucije 4 piksela s vrijednostima [9 7 3 5]. Na ovu sliku ćemo sada primijeniti Haarovu transformaciju. Podijelimo početni signal na parne i neparne koeficijente. Parne nazovimo *a* (u ovom slučaju to su 9 i 3), a neparne *b* (7 i 5).

Zbrojimo susjedne parove koeficijenata:

$$s = a + b\,.$$

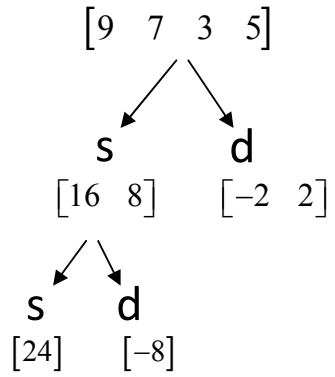
s nazivamo aproksimacijskim koeficijentom, i on predstavlja niske frekvencije slike.

Kada to napravimo, novi vektor će izgledati ovako: [16 8]. Kao što vidimo, dio informacije je izgubljen pa moramo uvesti koeficijente koji će nam pomoći pri rekonstrukciji ove slike i koje ćemo nazvati detaljima. Detalje ćemo izračunati tako da oduzmemo parne koeficijente od neparnih:

$$d = b - a\,,$$

te oni predstavljaju visoke frekvencije slike. Da bi kompresija bila što bolja, *d* treba biti što manji.

Prema tome, originalna slika je smanjila rezoluciju na dva piksela i na par koeficijenata detalja. Kada se ovaj proces rekurzivno ponavlja, dobivamo punu dekompoziciju slike prikazanu shemom:

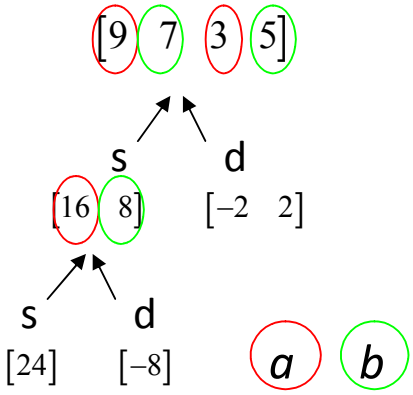


Dakle, wavelet transformacija ove slike bit će dana vektorom [24 –8].

Slika se može rekonstruirati na bilo koju rezoluciju tako da koeficijentima aproksimacije iz nižih rezolucija rekurzivno dodajemo i oduzimamo detalje i tako se krećemo prema višim rezolucijama. Koeficijente *a* i *b* "vratit" ćemo izrazima:

$$a = \frac{s-d}{2},$$
$$b = \frac{s+d}{2}.$$

Ova inverzna transformacija također se može prikazati shemom:



Ove operacije možemo izvesti u takozvanoj ljestvičastoj izvedbi koju karakterizira jako mala potreba za memorijom (upravo onoliko mjesta koliko zauzima sama slika).

Da je ova metoda zaista *in place*, pokazuju sljedeći izrazi. Poznato nam je:

$$s = a + b$$
$$d = b - a$$

Kada promijenimo redoslijed varijabli, dobijemo:

$$d = b - a$$
$$s = 2 \cdot a + d.$$

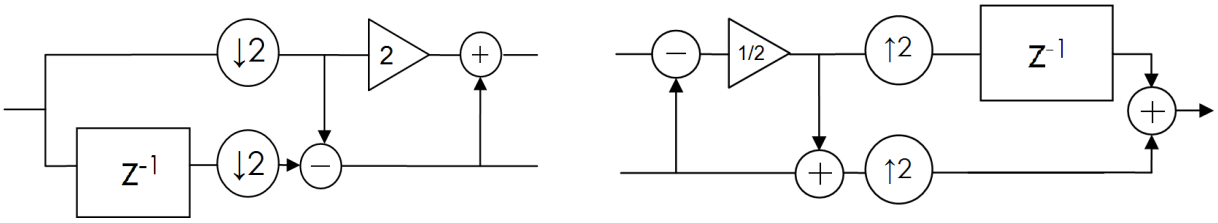
Uočavamo da za računanje koeficijenta *s* ne trebamo koeficijent *b*, te koeficijent *d* može zapisati na njegovo mjesto u memoriji. Također, izračunati koeficijent *s* možemo zapisati na mjesto koeficijenta *a*.

U sljedećoj razini, *d* postaje koeficijent *b*, a *s* postaje koeficijent *a* pa možemo ilustrativno pisati:

$$b = b - a$$
$$a = 2 \cdot a + d$$

Primjećujemo da nam varijable *s* i *d* više ne trebaju jer se na njihovo mjesto zapisuju novi koeficijenti *a* i *b*. To znači da nam ne treba niti dodatna memorija za zapisivanje međurezultata.

Ljestvičasta struktura Haarovog sloga korištenog u ovom projektu prikazana je na slici 2.

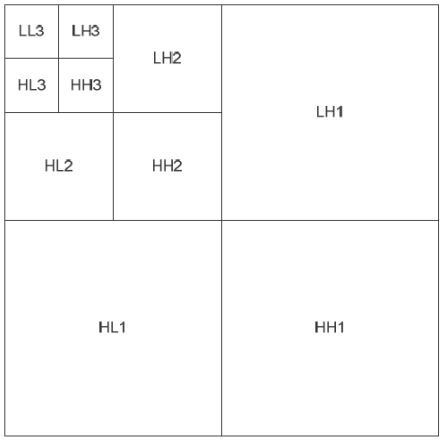


Slika 2: Ljestvičasta struktura Haarovog sloga

3.2. Haarova 2-D wavelet transformacija

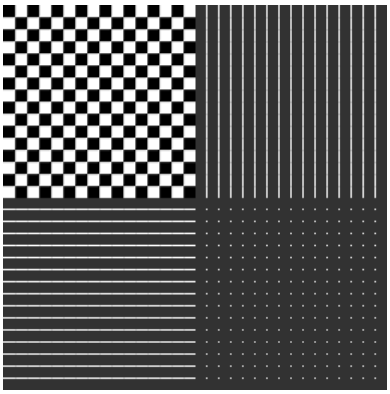
Transformacija 2-D slike je generalizacija već spomenute 1-D wavelet transformacije na dvije dimenzije. U ovom slučaju, 1-D transformacija se primjenjuje na svaki redak slike. Ovom operacijom u svakom retku dobivamo usrednjeni član i koeficijente detalja. Dalje se transformacija provodi nad stupcima tako dobivene matrice dok se, nakon nekoliko koraka, kao rezultat ne dobije jedan usrednjeni element (aproksimacija) i detalji.

Tipična 2-D diskretna wavelet transformacija koja se upotrebljava u kompresiji slike generira hijerarhijsku strukturu kao na slici 3.



Slika 3: Hijerarhijska struktura nastala nakon tri koraka 2-D wavelet tranformacije

Slika 4 dobar je primjer DWT dekompozicije u jednoj razini razlaganja. Jasno se vide četiri dijela slike. Gornji lijevi dio je aproksimacijski dio slike (LL), a ostali dijelovi prikazuju detalje u horizontalnom smjeru (LH), vertikalnom smjeru (HL) i dijagonalno (HH).



Slika 4: Ilustracija 2-D wavelet transformacije (jedna razina razlaganja)

4. SPIHT – *Set Partitioning In Hierarchical Trees*

Nakon kompresije, sliku treba kodirati. U literaturi nalazimo dva uobičajena načina – EZW (*Embedded Zerotree Wavelet*) i SPIHT (*Set Partitioning In Hierarchical Trees*). Odabran je SPIHT zato što je u odnosu na EZW brži i može povećati PSNR (vršnog odnos signala i šuma) za 0.3 do 0.6 dB. [4]

Neka su pikseli slike označeni kao $p_{i,j}$, gdje su (*i,j*) koordinate piksela. Kodiranje slike možemo prikazati na ovaj način:

$$\mathbf{c} = \Omega(\mathbf{p}),$$

gdje je **p** originalna slika u obliku matrice, Ω(.) unitarna transformacija, a **c** transformirana slika istih dimenzija kao originalna, gdje $c_{i,j}$ predstavlja transformirani koeficijent slike **p** na mjestu (*i,j*). Za potrebe kodiranja pretpostavit ćemo da je koeficijent $c_{i,j}$ cjelobrojan te da u zapisu zauzima najviše 16 bitova.

Na početku transformacije, dekoder inicijalizira rekonstrukcijski vektor $\hat{\mathbf{c}}$ na nulu da bi u tijeku algoritma prema kodiranoj informaciji ažurirao njegove komponente. Nakon što primi vrijednosti nekih koeficijenata, dekoder može rekonstruirati sliku:

$$\hat{\mathbf{p}} = \Omega^{-1}(\hat{\mathbf{c}}).$$

Glavni cilj ovakve transformacije je da se prvo prenese najvažnija informacija, dakle ona koja će rezultirati najmanjom distorzijom. Kao mjeru distorzije uzet ćemo najmanju kvadratnu pogrešku (eng. MSE – *minimum square error*). Izraz za srednju kvadratnu pogrešku glasi:

$$D_{mse}(p-\hat{p}) = \frac{1}{N}\sum_i\sum_j (p_{ij}-\hat{p}_{ij})^2,$$

gdje je N broj piksela u slici koji iznosi i×j.

Koristit ćemo i činjenicu da je euklidska norma invarijantna na unitarnu transformaciju Ω, odnosno da vrijedi:

$$D_{mse}(p-\hat{p}) = D_{mse}(c-\hat{c}) = \frac{1}{N}\sum_i\sum_j (c_{ij}-\hat{c}_{ij})^2.$$

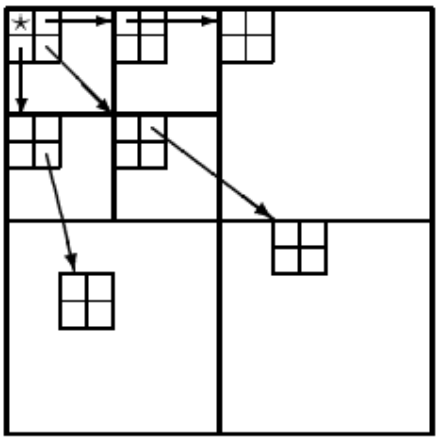
Iz posljednjeg izraza vidljivo je da, ako se dekoderu proslijedi prava vrijednost koeficijenta $c_{i,j}$, srednja kvadratna pogreška se smanji za $|c_{i,j}|^2/N$. To znači da se koeficijenti veće važnosti prenose prvi jer sadrže najviše informacije. Osim toga, pri prijenosu koeficijenata treba najznačajnije bitove prenijeti prve.

Glavno obilježje ovog algoritma je da se ne trebaju sortirati svi koeficijenti $c_{i,j}$, nego se oni odabiru tako da vrijedi $2^n \le |c_{i,j}| < 2^{n+1}$, gdje se *n* smanjuje u svakom prolazu. Za zadani *n*, ako je $|c_{i,j}| \ge 2^n$, kažemo da je koeficijent značajan (*significant*), inače je beznačajan (*insignificant*). U algoritmu se skupovi piksela dijele na podskupove T_m i provjerava veličinu koeficijenata pitanjem:

$$\max_{(i,j)\in\tau_m}\{|c_{i,j}|\} \ge 2^n ?$$

Ako je odgovor dekodera negativan, sve koeficijente u podskupu T_m tretirat će kao *beznačajne*, a ako je odgovor potvrdan, T_m se dijeli na nove dijelove koji se dalje ispituju na isti način sve dok se ne ispita svaki koeficijent.

Matrica piksela slike ima hijerarhijsku, piramidalnu strukturu prikazanu na slici 5 kojom su uređeni odnosi među pikselima (eng. *tree-structure*). Svaki čvor "stabla" odgovara pikselu koji je predstavljen svojim koordinatama. Njegovi direktni potomci, djeca, odgovaraju pikselima iste prostorne orijentacije u sljedećoj razini piramide. Ova struktura je definirana tako da svaki čvor ili nema potomka ili ih ima četiri koja uvijek formiraju grupu od 2×2 susjedna piksela. Na slici 5 strelice prikazuju djecu pojedinih čvorova. U skupu korijena (najviša razina piramide), u svakoj grupi od 2×2 susjedna piksela jedan od njih (na slici 5 označen zvjezdicom) neće imati više djece.



Slika 5: Hijerarhijska struktura matrice piksela

Da bismo mogli razlikovati skupove piksela, uvest ćemo neke oznake:

- $O(i, j)$: skup koordinata sve djece čvora (i, j),
- $D(i, j)$: skup koordinata svih potomaka čvora (i, j),
- H: skup koordinata svih čvorova na najvišoj razini (korijeni).

Također, vrijedi i:

$L(i, j) = D(i, j) - O(i, j)$.

Osim na najvišoj i najnižoj razini, vrijede ovi odnosi između koordinata piksela:

$$O(i, j) = \{(2i, 2j), (2i, 2j+1), (2i+1, 2j), (2i+1, 2j+1)\}.$$

Pravila pri podjeli skupova:

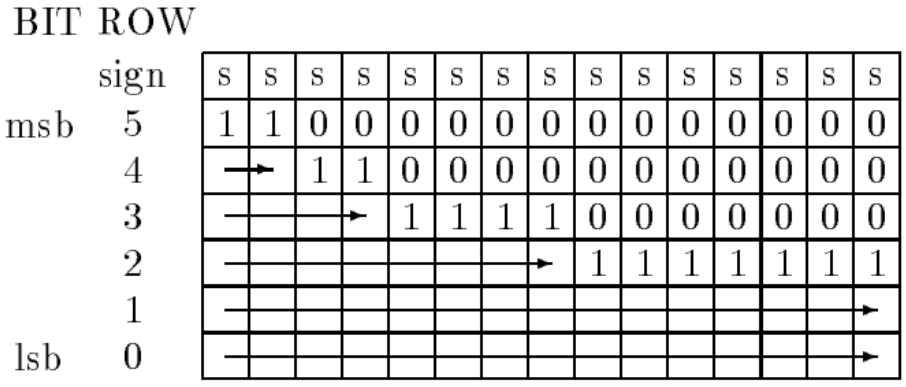
1. Početna podjela se formira skupom $\{(i, j)\}$ i $D(i, j)$, za sve $(i, j) \in H$.
2. Ako je $D(i, j)$ značajan, dijeli se na $L(i, j)$ i na četiri skupa s jednim elementom gdje je $(k, l) \in O(i, j)$.
3. Ako je $L(i, j)$ značajan, dijeli se u četiri skupa $D(k, l)$ gdje je $(k, l) \in O(i, j)$.

Poredak kojim se podskupovi ispituju prema značajnosti je važan pa se u praktičnoj primjeni *significance information* sprema u uređene liste koje se zovu:

- LIS (*list of insignificant sets*) – lista beznačajnih skupova,
- LIP (*list of insignificant pixels*) – lista beznačajnih piksela,
- LSP (*list of significant pixels*) – lista značajnih piksela.

U svakoj listi članovi se definiraju koordinatama (*i*, *j*). Liste LIP i LSP sadrže koordinate piksela. LIS predstavlja skup *D*(*i*, *j*) ili skup *L*(*i*, *j*). Da bismo ih mogli razlikovati, kažemo da je skup *D*(*i*, *j*) tipa A, a skup skup *L*(*i*, *j*) tipa B. Za vrijeme sortiranja u algoritmu, pikseli u listi LIP se ispituju i oni značajni se zapisuju u LSP. Na sličan način ispituju se i skupovi. Ako je koji postao značajan, briše se s liste i dijeli na podskupove.

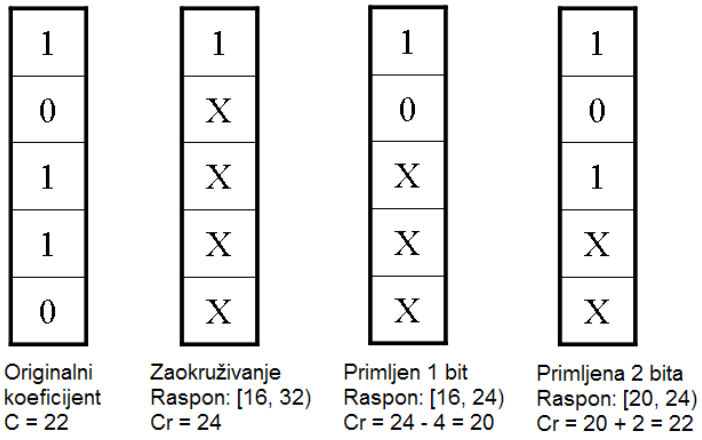
Na slici 6 shematski su prikazani pikseli slike u binarnom zapisu, po stupcima. Bitovi u gornjem retku prikazuju predznak koeficijenta. Ostali retci numerirani su odozdo prema gore, prema koraku *n*, a u najnižem retku su najmanje značajni bitovi. Dekoder prima bitove koji odgovaraju koeficijentima tako da vrijedi $2^n \le \left|c_{i,j}\right| < 2^{n+1}$.



Slika 6: Shematski prikaz piksela slike u binarnom zapisu

Rekonstrukciju možemo opisati i shematski (Slika 7). Neka je originalni koeficijent primljen iz matrice piksela C = 22 i neka je zapisan pomoću 5 bitova. Dekoder najprije primi najznačajniji bit, jedinicu, što znači da se koeficijent nalazi u rasponu [2^4, 2^5) ili [16, 32) i zaokružuje ga na sredinu raspona, što iznosi Cr = 24. U sljedećem koraku dekoder prima drugi bit, nulu, koja će malo povećati preciznost. Ona pokazuje da je broj manji od ($2^5 – 2^4$) pa sada znamo da je vrijednost koeficijenta u rasponu [16, 24) i njegovu vrijednost zaokružujemo na Cr = 20. Ovaj postupak odvija se u *refinement passu* prikazanom kasnije u algoritmu i može se ponavljati dok ne postanemo zadovoljni preciznošću s kojom rekonstruiramo dani koeficijent.

Slikom 6 to bismo mogli protumačiti ovako: ako prenesemo sve bitove u retcima 2, 3 i 4, rekonstrukcija je zadovoljavajuća; ako prenesemo sve bitove u retku 1, rekonstrukcija je visoke kvalitete; a ako prenesemo sve bitove u retku 0 rekonstrukcija je bez gubitaka (*lossless*).



Slika 7: Rekonstrukcija koeficijenata u dekoderu

Tijek algoritma

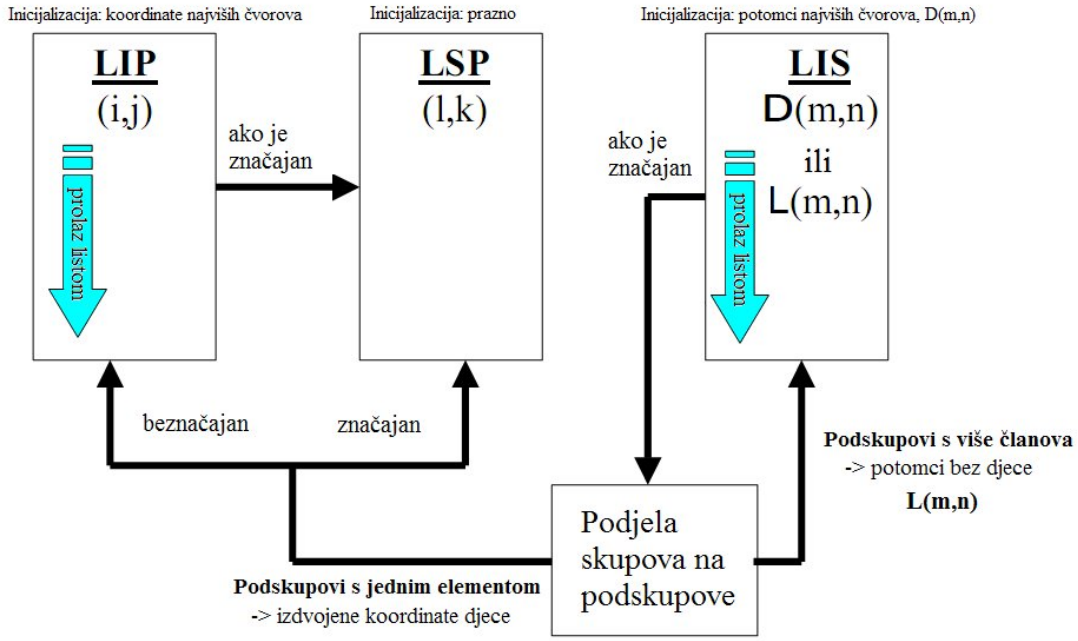
1. Inicijalizacija: $n = \left\lfloor \log_2 \left(\max_{i,j} \left\{ \left| c_{i,j} \right| \right\} \right) \right\rfloor$. Postaviti LSP kao praznu listu, dodati koordinate $(i, j) \in H$ u LIP, i to samo one kojima su potomci u listi LIS članovi tipa A.

2. *Sorting pass* (prolaz sortiranja):
   2.1. Za svaki član (*i*, *j*) u LIP napravi:
      2.1.1. odredi $S_n(i, j)$ i zapiši ga u izlazni niz bitova;
      2.1.2. ako je $S_n(i, j) = 1$, premjesti (*i*, *j*) u LSP i zapiši u izlazni niz bitova predznak od $c_{i,j}$
   2.2. Za svaki član u LIS napravi:
      2.2.1. ako je član liste tipa A, tada:
      - zapiši $S_n\{D(i, j)\}$ u izlazni niz bitova;
      - ako je $S_n\{D(i, j)\} = 1$, tada:
        ◊ za svaki $(k, l) \in O(i, j)$ napravi:
          ▪ odredi $S_n(k, l)$ i zapiši ga u izlazni niz bitova;
          ▪ ako je $S_n(k, l) = 1$, tada dodaj (*k*, *l*) na LSP i zapiši u izlazni niz bitova predznak od $c_{i,j;}$
        ◊ ako je $L(i, j) \neq 0$, pomakni (*i*, *j*) na kraj liste LIS kao član tipa B i idi na korak 2.2.2, inače briši (*i*, *j*) s liste LIS;
      2.2.2. Ako je član tipa B, tada:
      - odredi $S_n\{L(i, j)\}$ i zapiši ga u izlazni niz bitova
      - ako je $S_n\{L(i, j)\} = 1$, tada:
        ◊ dodaj svaki $(k, l) \in O(i, j)$ na kraj LIS-a kao član tipa A;
        ◊ briši (*i*, *j*) iz LIS-a;

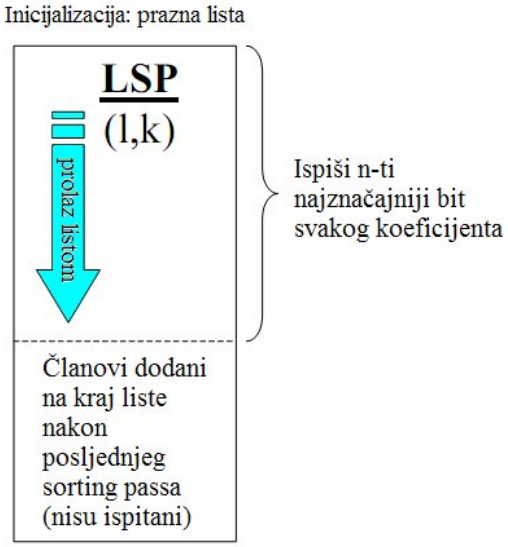
3. *Refinement pass*: za svaku stavku (*i*, *j*) u listi LSP, osim onih uključenih u posljednji *sorting pass* (s istim *n*-om), zapiši u izlazni niz bitova n-ti najznačajniji bit koeficijenta $| c_{i,j} |$;

4. Ažuriranje kvantizacijskog koraka: smanji *n* za 1 i idi na korak 2.

U 4. koraku može se provjeravati greška i prema tome odrediti kada se želi da algoritam stane.



Slika 8: ***Sorting pass***



Slika 9: ***Refinement pass***

5. Implementacija algoritma u Matlabu

Za projekt Maslinet razvijaju se dvije vrste kamera: kamera razlučivosti 1600x1200 piksela (UXGA) i kamera razlučivosti 480x640 piksela (VGA).
Za potrebe ovog rada korištena je slika dobivena VGA kamerom. Kamera na svom izlazu daje 2 bytea po pikselu u formatu YCbCr 4:2:0.

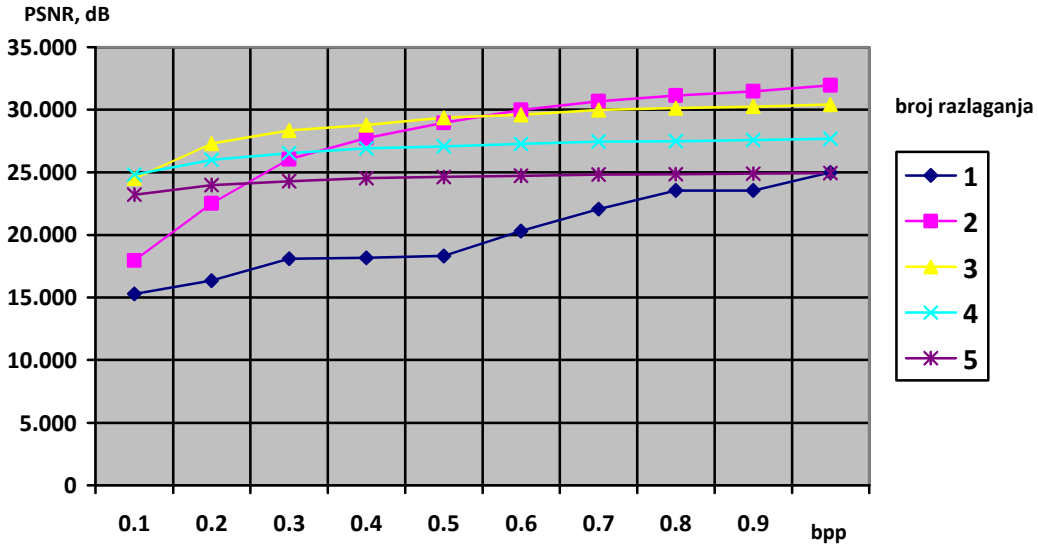
Mode	1st Byte	2nd Byte	3rd Byte	4th Byte
Default (no swap)	Cb_i	Y_i	Cr_i	Y_{i+1}

Tabela 1: Format izlaznog signala kamere [[1]]

U Matlabu je napravljena funkcija *SPIHT_main* koja poziva podfunkcije koje nad zadanom matricom (slikom) redom obavljaju wavelet dekompoziciju, kodiranje SPIHT metodom, dekodiranje SPIHT metodom te wavelet rekonstrukciju.
Od slike dobivene kamerom formirane su 3 komponente – Y (veličine 256x256 piksela,), te Cb i Cr (svaka veličine 128x128 piksela). Za daljnje testiranje algoritama korištena je slika Y komponente umanjena za 128, radi postizanja simetrije koeficijenata SPIHT-a.
Naravno, također je moguće bilo koju sliku učitati funkcijom *imread* u matricu te tu matricu predati funkciji *SPIHT_main*.
Parametri koje zadaje korisnik su broj razina razlaganja wavelet dekompozicijom te broj bitova po pikselu (tj. elementu matrice), čime ograničavamo broj bitova u izlaznom nizu koji predstavlja kodiranu sliku. Kad se postigne zadani broj bitova, obrada matrice SPIHT metodom prestaje i signal koji predstavlja komprimiranu sliku spreman je za prijenos ili pohranu.
Dekoder se sastoji od algoritma idejno jednakog koderu te se prolaženjem kroz niz bitova pronalaze odgovori na postavljena pitanja i rekonstruira početni signal (slika).

5.1. Rezultati testiranja

Sliku veličine 256x256 piksela dekomponiramo funkcijom *DWT_decomposition* te dobivenu matricu prosljeđujemo funkciji *SPIHT_coder*. Izlaz te funkcije je niz bitova koji predstavlja komprimiranu sliku. Taj niz bitova predaje se funkciji *SPIHT_decoder* koja iz njega rekonstruira DWT matricu. Dobivena matrica se prosljeđuje funkciji *DWT_reconstruction* koja vraća rekonstruiranu sliku.
Ovisno o broju bitova koje zadamo, dobijemo rekonstruirane slike različite kvalitete (Slika 10).



Slika 10: Odnos PSNR i bpp za različiti broj razina razlaganja

Cilj nam je uz što manji bpp (manji broj potrebnih bitova za pohranu komprimirane slike) postići što veći PSNR (veća kvaliteta slike). Iz grafa na slici 10 vidi se da bi dobar izbor bio slučaj za tri razine razlaganja i 0.3 bpp. Iako je s drugim kombinacijama moguće postići i veći PSNR, poboljšanje nije značajno u odnosu na broj dodatno potrošenih bitova.

DWT dekompozicija, Haar, broj razina razlaganja= 3

Slika 11: Rezultat DWT dekompozije s 3 razine razlaganja

Treba samo naglasiti da su većina detalja jako mali ili nula (dominira siva boja koja označava nulu, jer smo prethodno od slike oduzeli 128).

Original 256x256px, veličina 524288B

Rekonstruirana slika, bpp=0.3

Kompresija= 27 puta, PSNR= 28.3555dB

Pogreška

Slika 12: Kompresija slike s 3 razine razlaganja i 0.3 bpp – originalna slika, rekonstruirana slika i slika pogreške

Crna boja na slici označava koeficijente vrijednosti nula.

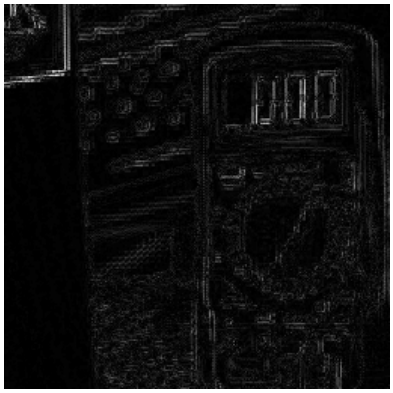
Pokažimo i slučaj za dvije razine dekompozije, uz bpp=1. Vidimo da rezultat jest bolji, ali treba uzeti u obzir da u ovom slučaju trebamo 65 636 bitova za pohranu (ili slanje) slike, dok smo u prethodnom trebali 19 661 bit.

Original 256x256px, veličina 524288B

Rekonstruirana slika, bpp=1

Kompresija= 8 puta, PSNR= 31.961dB

Pogreška



Slika 13: Kompresija slike s 2 razine razlaganja i 1 bpp – originalna slika, rekonstruirana slika i slika pogreške

Budući da korištena VGA kamera daje sliku u boji, pokažimo kompresiju i takve slike. Kao što smo gore komprimirali sliku Y komponente, na isti način komprimiramo i preostale dvije komponente (Cb i Cr). Budući sa su te komponente decimirane, slike koje komprimiramo su upola manjih dimenzija, te ih nakon rekonstrukcije pomoću funkcije *imresize* linearno interpoliramo. Zatim ih sve tri zapišemo u jednu matricu koju pretvorimo u RGB matricu. Na slici 14 prikazane su originalna i rekonstruirana slika u boji, uz 3 razine razlaganja i 0.3 bpp. Vidimo degradaciju u kvaliteti, no uzevši u obzir da smo sliku veličine 1 048 576 bitova (256x256x2x8) komprimirali na samo 29 502 bita (dakle, više od 35 puta), rezultat je zadovoljavajuć.

Slika 14: Kompresija slike u boji s 3 razine razlaganja i 0.3 bpp – originalna slika i rekonstruirana slika

U literaturi se spominju i neke posebne metode komprimiranja slika u boji koje se temelje na sličnostima među različitim komponentama boje (tj. osvjetljenja).

Istaknimo još da obrada ovako velike slike može predstavljati problem ako se izvodi na sklopovlju poput FPGA koji ima ograničenu količinu memorije te nije zgodno imati toliko velike *buffere*. Zato postoji mogućnosti podjele slike na blokove (tzv. *partitioning* metoda), koji se onda zasebno komprimiraju. Postoje razne kombinacije. Ovdje je izabrana veličina bloka 64x64 piksela, dakle slika je podijeljena na ukupno 16 blokova. Rezultat za opciju koja se pokazala optimalnom za cijelu sliku: 3 razine razlaganja, 0.3 bpp prikazan je na slici 15.

Original 256x256px, veličina 524288B

Rek. slika, bpp=0.3, broj razlaganja=3

Veličina blokova= 64, PSNR= 20.5866dB

Pogreška (partitioned)

Slika 15: Kompresija slike rastavljanjem na blokove, s 3 razine razlaganja i 0.3 bpp – originalna slika, rekonstruirana slika i slika pogreške

Nije napravljena detaljna analiza odnosa PSNR i bpp za različite brojeve razina razlaganja, ali iz par raznih pokušaja primijeti se da ova opcija ne daje najbolje rješenje. To dokazuje da kvaliteta rekonstruirane slike ovisi i o veličini slike.

Primjećuje se da PSNR u ovom slučaju ne odražava stvarnu kvalitetu rekonstruirane slike. Unatoč tome što naše oko primjećuje da je slika dosta dobro rekonstruirana, PSNR je dosta mali zbog dijela slike koji nije dobro rekonstruiran. U našem slučaju taj dio je, mogli bismo reći, nebitan za opće razumijevanje slike, no u nekom drugom slučaju mogao bi biti od ključne važnosti.

Dakle, također i o samoj slici ovisi kakva će biti kvaliteta rekonstrukcije.
Kod obrade slike podjelom na blokove često se javljaju tzv. *blocking artefacts*, odnosno nepravilnosti nastale upravo zbog prirode samog algoritma – blokovi se obrađuju nezavisno jedan od drugoga te prijelazi s jednog na drugog u rekonstruiranoj slici znaju biti dosta oštri. Da bi se oni uklonili, algoritam se modificira tako da se pri obradi jednog bloka DWT dekompozicijom uzmu u obzir i koeficijenti njemu susjednih blokova te se tako prijelazi ublaže.

6. Zaključak

U bežičnim senzorskim mrežama sve je učestalija potreba za akvizicijom slika visoke razlučivosti. To predstavlja veliki izazov u projektiranju uzevši u obzir ograničenja čvorova: energija, male procesorske mogućnosti, memorija. Zato je potrebno naći kompromis između lokalne obrade informacije (kompresije) i slanja prema korisniku.

Jedan od kompresijskih postupaka koji je dovoljno jednostavan, a daje dosta dobre rezultate i time je pogodan za implementaciju u takvim projektima, je SPIHT.

Kôd napisan za ovaj projekt dao je zadovoljavajuće rezultate. Ipak, potrebno je još proučiti SPIHT algoritam i ispraviti eventualne pogreške. Iz literature se može zaključiti da je moguće postići još bolje rezultate. Također, treba proučiti da li se može pojednostavniti i ubrzati rad kodera (za dekoder su zahtjevi manje strogi jer se on ne implementira na čvoru, već na serveru koji nema značajna ograničenja u energiji i procesorskoj moći). Trenutno je dekoder značajno brži i jednostavniji od kodera jer ne pretražuje više puta matricu radi ispitivanja uvjeta.

Slika na kojoj je obavljeno testiranje snimljena je u laboratoriju i prikazuje objekt na kojemu su prisutne razne boje i debljine linija. No, kvaliteta rezultata dobivenih SPIHT-om ovisi i o samoj slici. Treba snimiti sliku koja (barem približno) odgovara slici koja će se snimati u projektu MasliNet – sliku stabla, zelenila, te za nju ispitati optimalne parametre.

Za implementaciju u hardveru, osim što se radi rastavljanje slike na blokove koji se pojedinačno obrađuju, postoje modifikacije SPIHT algoritma koje su pogodnije s obzirom na već spomenuta ograničenja čvorova u bežičnim senzorskim mrežama.

7. Literatura

[1] 1/4-Inch SOC VGA CMOS Digital Image Sensor, MT9V131, Micron Datasheet

[2] Maly, J.; Rajmi, P.: DWT-SPIHT image codec implementation, http://wavelets.triablo.net/spiht/

[3] MasliNET. URL: http://www.maslinet.com/ (2009-01-15)

[4] Pearlman, William A., Said, Amir: New fast efficient image codec based on SPIHT, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, lipanj 1996.

[5] Predavanja iz Naprednih metoda digitalne obradbe signala. URL: http://nmdos.zesoi.fer.hr/predavanja.html

[6] Ritter: Wavelet based image compression using FPGAs, 2002.

[7] Talukder, K.H., Harada, K.: Haar wavelet based approach for image compression and quality assessment of compressed image, IAENG International Journal of Applied Mathematics, 36:1, IJAM_36_1_9, veljača 2007.

[8] Tran; Scalable coding, *thanglong.ece.jhu.edu/Course/443/Lectures/520.443-scalable.ppt* (2009-01-15)

[9] Van Fleet, P. J. : The discrete haar wavelet transformation, 2007., http://cam.mathlab.stthomas.edu/wavelets/pdffiles/NewOrleans07/HaarTransform.pdf