KOMPRESIJA SLIKE

Postoji više razloga zašto se mora vršiti kompresija slike prije njenog memorisanja ili prenosa. Tri osnovna su:

- veliki zahtjevi za memorijskim prostorom,
- relativno spori uređaji koji nisu u mogućnosti da prikazuju nekomprimovane sliku u realnom vremenu,
- propusni opseg mreže koji ne dozvoljava prenos nekomprimovane slike u realnom vremenu.

Motivacija

- slika A4 stranice u CMYK modelu rezolucije 300 dpi
  - dimenzije A4: 8,27in x 11,69in = 96,68in^2
  - ukupan broj piksela: 96,68in^2 x 300^2 = 8701200
  - četiri bajta po pikselu (CMYK)
  - ukupna veličina slike iznosi: 8701200 x 4 = 34804800 ≈ 33.2 MB

- 1 minut audio snimka CD kvaliteta
  - frekvencija uzorkovanja: 44.1 kHz – 44100 uzoraka u sekundi
  - 16-bitna kvantizacija: 2 bajta po uzorku
  - 2 kanala (stereo)
  - potrebno je 2 x 44100 x 2 x 60 = 10584000 bajta ≈ 10 MB

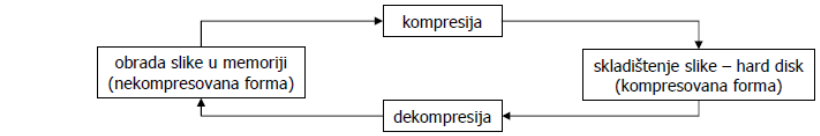
- 1 minut video snimka u PAL formatu
  - rezolucija: 768x576
  - RGB model: 3 bajta po pikselu
  - 25 frejmova u sekundi
  - potrebno je 768 x 576 x 3 x 25 x 60 ≈ 1.85 GB

Primjene

- skladištenje podataka
  - hard disk, razni optički / prenosni mediji
- prenos podataka
  - web, email, itd

Moderne tehnike za kompresiju slike nude rješenje za ove probleme tako da je moguće izvršiti kompresiju audio signala do 10 puta, mirne slike čak i do 50 puta bez znatnog uticaja na kvalitet reprodukovane slike. U slučaju sekvence slika (video signala) stepen kompresije može biti znatno veći, čak i preko 200 puta.

Potrebno je napomenuti da se prije obrade podaci moraju vratiti u originalnu formu.



Klasifikacija tehnika za kompresiju slike

Kompresija digitalnih podataka, pa prema tome i slike, se zasniva na različitim algoritmima koji koriste prostornu redundatnot informacija koje sadrži digitalna slika. Redundantnost informacija u slici postoji zbog znatne prostorne korelacije piksela. Statistička redundantnost je povezana sa raspodelom intenziteta piksela slike i može se eliminisati korišćenjem entropijskih tehnika. Eliminacijom statističke redundantnosti postiže se kompresija bez gubitaka. U slučaju video sekvenci susjedne slike u sekvenci se malo razlikuju (osim u slučaju vrlo brzih pokreta) pa kažemo da pored prostorne redundantnosti postoji i vremenska redundantnost. Vremenska redundantnost se otklanja metodama kompenzacijom pokreta, čime se postiže još veći stepen kompresije video sekvenci nego mirnih slika.
Tehnike kompresije se mogu podijeliti u dvije kategorije:

- kompresija bez gubitaka;
- kompresija sa gubicima.

Tehnike bez gubitaka omogućavaju perfektnu rekonstrukciju originalne slike, dok je kod tehnika sa gubicima rekonstruisana slika slična originalu, ali ne i u potpuosti jednaka.

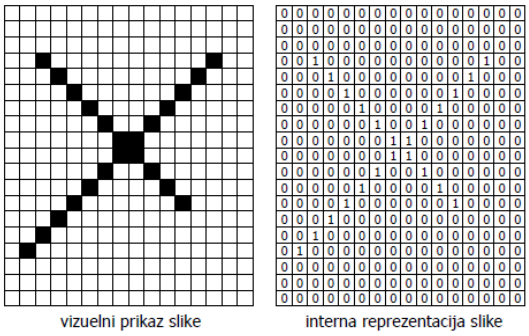
Tehnike sa gubicima omogućavaju veći stepen kompresije i zbog toga se mnogo više primjenjuju. Kompresija bez gubitaka se koristi u onim slučajevima kada je neophodno sačuvati originalnu sliku, da li zbog nemogućnosti njenog ponovnog snimanja ili važnih informacija koje bi, eventualno, mogle biti izgubljene u postupku kompresije (na primer, medicinski snimci). Ako je lako ponoviti proces snimanja ili se može tolerisati izvestan stepen gubitka informacija (video telefonija, televizija, multimedijalni sistemi, itd.) koriste se postupci kompresije sa gubicima.

KOMPRESIJA BEZ GUBITAKA

Posmatrajmo sivu sliku od $M \times N$ piksela sa mogućih 2^B nivoa svjetline. Ako se za zapis svakog nivoa svjetline koristi B bita, za memorisanje slike u ovoj formi je neophodno MNB bita. Ovaj način zapisa digitalnih slika je poznat kao impulsna kodna modulacija (Pulse Code Modulation - PCM).

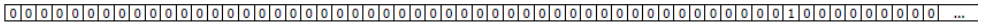
Kodovanje dužina nizova

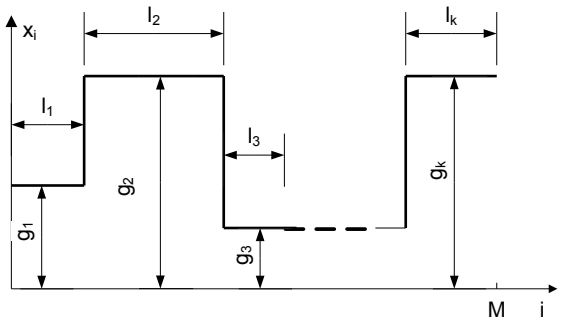
U slučajevima kada imamo mali broj nivoa svjetlina, dešava se da određen broj susjednih piksela u jednoj liniji ima istu vrijednost. Takvu grupu piksela dužine l_i sa nivoom sivog g_i, nazivamo niz. Ako se u jednoj liniji nalazi k takvih segmenata, sadržaj linije slike se umjesto pojedinačnim vrijednostima piksela može predstaviti uređenim parovima (g_i, l_i), $1 \le i \le k$, i umjesto kodovanja pojedinačnih piksela može se kodovati cijela grupa. Oaj metod kodovanja se naziva metod kodovanja dužina nizova (run-length coding - RLC).



vizuelni prikaz slike | interna reprezentacija slike

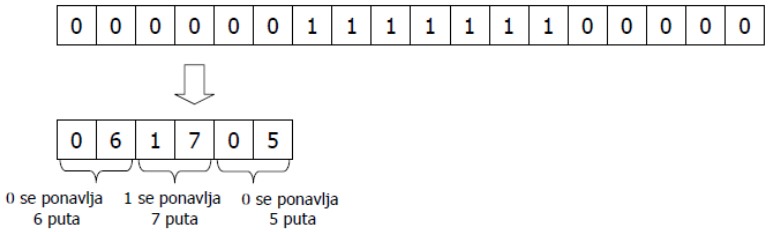
– internu reprezentaciju slike predstavimo kao niz, vrstu po vrstu





Slika 108. Grafička prikaz sadržaja jedne linije slike

Primjer



Svaki par (g_i, l_i) naziva se niz sivog (gray-level run). Veliki stepen kompresije se postiže ako su dužine nizova sivog velike, a posebno u slučajevima binarnih slika, kada postoje samo dvije vrste piksela, vrijednosti 0 i 1. Kod binarnih slika je često veća vjerovatnoća pojave jednog od nizova, bijelog ili crnog. Ako se pretpostavi da linija počinje nizom sa većom vjerovatnoćom, onda se koduju samo dužine nizova, a ne i njihovi nivoi za koje se zna da se smenjuju. Kraj svake linije se označava posebnim kodnim simbolom (end of line – EOL). Kraj slike se označava ponavljanjem EOL simbola, najčešće 6 puta.

RLC kodovanje je standardizovano od strane Međunarodne telekomunikacione unije ITU-T za prenos binarnih slika po javnoj telefonskoj mreži. Osnovna primjena je prenos faksimila.

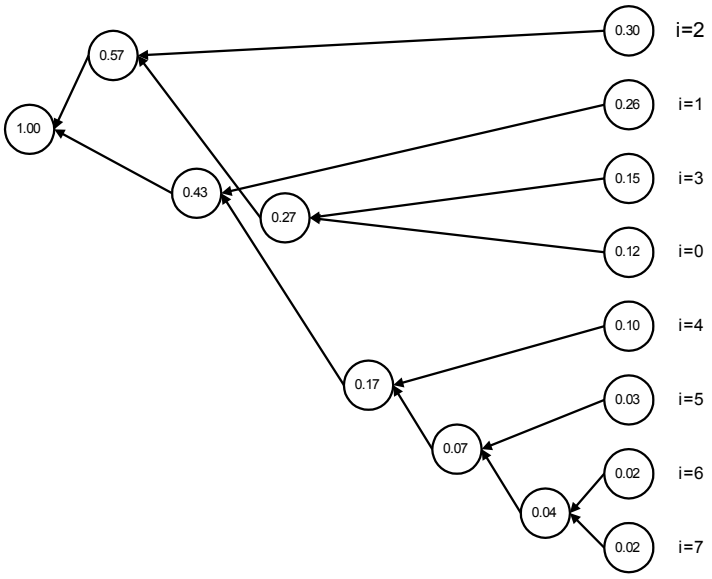
Hafmanovo kodovanje

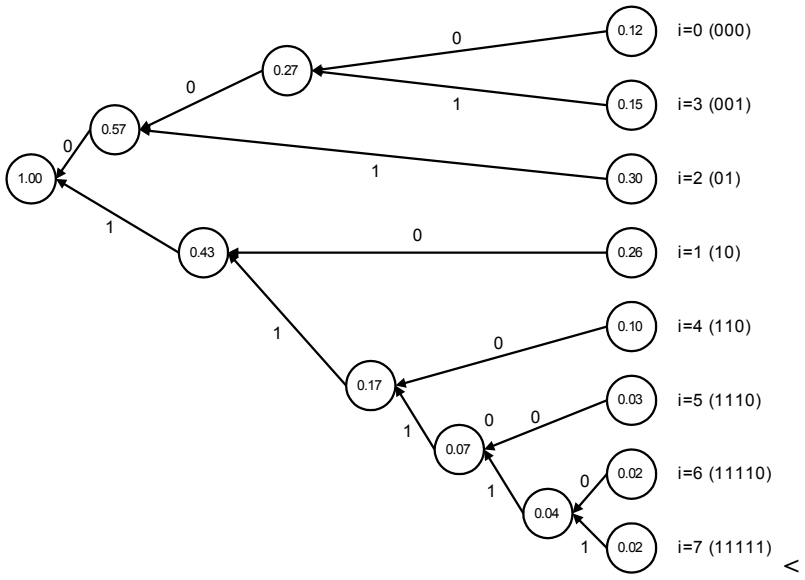
Suština Hafmanovog (Huffman) kodovanja slike leži u tome da se nivoima svjetline koji se češće pojavljuju pridružuju kodne riječi sa manje bita, dok se nivoima svjetline koji se ređe pojavljuju pridružuju kodne riječi sa više bita. Na taj način se postiže da prosječni broj bita po pikselu bude manji od B. Tabela kodnih riječi koja povezuje nivoe svjetline sa kodnim riječima se formira tako da se minimizira prosječna dužina kodne reči. Pri tome su dužine kodnih riječi različite.

Prilikom zapisa slike, sve kodne riječi se spajaju u neprekidan niz bitova, iz koga je na kraju neophodno dešifrovati pojedinačne kodne riječi. Zbog toga nijedna kraća kodna reč ne smije predstavljati prefiks (početak) duže kodne reči.

Primjer:

Pretpostavimo jednostavan slučaj slike sa osam nivoa intenziteta ($B = 3$). Vjerovatnoće pojavljivanja svakog od nivoa svjetline $p(i)$ se mogu procijeniti na osnovu histograma slike. Ove vjerovatnoće se poredaju opadajućim redoslijedom i pridruže se krajnjim desnim čvorovima grafa. Zatim se dva čvora sa najmanjim vjerovatnoćama spajaju i novom čvoru se dodjeljuje zbir njihovih vjerovatnoća. Postupak se nastavlja sve dok zbir vjerovatnoća čvorova koji se spajaju ne postane jednak jedinici, Slika 107. Tako se formira kodno stablo. Ono se zatim preuredi tako da se eliminišu presjeci grana. Granama stabla koje se stiču u korijen pridružuje se vrijednosti 0 (gornja grana) i 1 (donja grana). Ista procedura se nastavlja do krajnjih desnih čvorova. Kodna riječ koja odgovara nekoj svjetlini se dobiva očitavanjem nula i jedinica koje se nalaze na putu od korijena do čvora koji odgovara toj svjetlini.





Slika 107. Konstrukcija Hafmanovog koda

Hafmanova kodna knjiga predstavlja tabelu preslikavanja svih mogućih nivoa svjetline u odgovarajuće kodne riječi. Hafmanova kodna knjiga za prethodi primjer je data sljedećom tabelom.

Hafmanova kodna knjiga za navedeni primjer ima oblik:

i	**Hafmanov kod**
0	000
1	10
2	01
3	001
4	110
5	1110
6	11110
7	11111

U navedenom primjeru prosječna dužina kodne riječi je $L = 2.55$, pa je realizovani stepen kompresije 3/2.55= 1.176.

Primjenom Hafmanovog koda se ne može postići veliki stepen kompresije kod sivih slika sa velikim brojem nivoa sivog jer je histogram video signala u dužem vremenskom periodu prilično uniforman, te se ovaj kod koristi u kombinaciji sa algoritmima za prediktivno ili transformaciono kodovanje.

LZW postupak kompresije

Za kompresiju slike se mogu iskoristiti i postupci namijenjeni kompresiji binarnih datoteka, koji su široko rasprostranjeni u računarstvu. Jedan od takvih postupaka je poznati LZW algoritam, nazvan po inicijalima njegovih autora (Lempel, Ziv, Welch).

Slčno prethodno opisanim algoritmima LZW algoritam koduje nizove simbola, ali za razliku od njih generiše kodnu tabelu u toku postupka kodovanja.

Konstrukcija kodne tabele se vrši i prilikom kodovanja i prilikom dekodovanja slike. Digitalna slika se posmatra kao dugačak jednodimenzionalni niz, sastavljen od podnizova čija dužina može biti različita, zavisno od primjene algoritma (npr. jedan bajt).

Kodovanje

Na početku izvršavanja algoritma se vrši inicijalizacija tokom koje se kodna tabela popuni svim pojedinačnim podnizovima koji se mogu pojaviti na ulazu (npr., ako su dužine podnizova jedan bajt, kodna tabela u početku sadrži brojeve od 0 do 255). Takođe se vrši inicijalizacija promjenljive koja sadrži prefiks niz P tako što se ona napuni nizom dužine 0. Zatim se u svakom koraku učitava po jedan podniz slike, smješta u promjenljivu Z, i u tekućoj promjenljivoj S se vrši spajanje prefiks niza i sadržaja promenljive Z, $S = P + Z$. Ako se sadržaj tekuće promjenljive S poklapa sa nekim od postojećih sadržaja u kodnoj tabeli, prefiks niz se ažurira dodavanjem tekućeg niza ($P = S$) i postupak se ponavlja da bi se pronašla najduža moguća sekvenca podnizova iz ulaznog niza koja se već nalazi u kodnoj tabeli. U trenutku kad se desi da nakon dodavanja novog podniza Z iz ulaznog niza u tekuću promjenljivu njen sadržaj ne postoji u kodnoj tabeli, (što znači da se samo prefiks sadrži u kodnoj tabeli), kodna riječ prefiksa se šalje u izlazni niz, kodna tabela se dopunjava dodavanjem tekuće promenljive $S = P + Z$ na kraj kodne tabele, a prefiks se postavlja na vrijednost posljednjeg pročitanog niza ($P = Z$). Kada se dođe do kraja slike, u izlazni niz se ubaci kodna reč za kraj datoteke EOI.

Algoritam za kodovanje

1. Na početku kodna tabela sadrži sve pojedinačne podnizove koji se mogu pojaviti na ulazu, a prefiks P je prazan;
2. Z := sljedeći podniz iz ulaznog niza;
3. Postoji li P+Z u kodnoj tabeli?
   - ako postoji, P := P+Z (P proširi Z-om);
   - ako ne postoji,
     I. u izlazni niz pošalji kodnu riječ koja odgovara P-u (pozicija na kojoj je u kodnoj tabeli upisan P);
     II. u kodnu tabelu dodaj P+Z;
     III. isprazni P i u njega upiši samo Z (P := Z);
4. ima li još znakova u ulaznom nizu?
   - ako ima, vrati se na korak 2;
   - ako nema:
     I. u izlazni niz pošalji kodnu riječ koja odgovara P-u;
     II. KRAJ.

Primjer kodovanja:

Sadržaj kodne tabele na početku kodovanja/dekodovanja:

(1) A
(2) B
(3) C

Ulazni niz za kodovanje:

Pozicija	1	2	3	4	5	6	7	8	9
Podniz	A	B	B	A	B	A	B	A	C

Postupak kodovanja

Korak	Poz	Kodna tabela	Izlaz
1.	1	(4) A B	(1)
2.	2	(5) B B	(2)
3.	3	(6) B A	(2)
4.	4	(7) A B A	(4)
5.	6	(8) A B A C	(7)
6.	--	--	(3)

Važno je primjetiti da je poslednji znak podniza koji je dodat u kodnu tabelu u nekom koraku istovremeno i prvi znak sljedećeg podniza koji se koduje.

Dekodovanje

I prilikom dekodovanja LZW algoritmom kodna tabela se progresivno rekonstruiše tako da nije potrebno pamtiti niti prenositi kodnu tabelu. Prije početka dekodovanja kodna tabela izgleda isto kao i prije kodovanja - sadrži sve pojedinačne znakove koji se mogu pojaviti na ulazu.

Radi objašnjenja uzmimo neki trenutak u toku dekodovanja, kad kodna tabela već sadrži i znakove veće dužine. Algoritam zapamti prethodnu kodnu riječ (pK), a zatim učita trenutnu (tK), pronađe u kodnoj tabeli niz.tK (sadržaj kodne tabele koji odgovara kodnoj riječi tK) i pošalje ga u izlazni niz. U kodnu tabelu se zatim doda niz.pK proširen prvim znakom niza.tK. Budući da moramo učitati sljedeću kodnu riječ da bismo znali kojim znakom da proširimo trenutnu, nizove dodajemo u kodnu tabelu sa zakašnjenjem od jednog koraka u odnosu na kodovanje.

Poseban je slučaj ako tK odgovara prvom slobodnom mjestu u kodnoj tabeli (kojem još nije pridružen niz). To se može dogoditi upravo zbog navedenog kašnjenja s dodavanjem nizova u kodnu tabelu. Događa se kad prilikom kodovanja naiđemo ponovo na niz koji smo upravo dodali u kodnu tabelu u prethodnom koraku. Prilikom dekodovanja taj niz još ne postoji u kodnoj tabeli. Budući da se prilikom kodovanja, odmah nakon upisa u kodnu tabelu, vrši izjendačavanje $P = Z$ sljedeći podniz koji kodujemo uvijek počinje sa zadnjim znakom prethodnog. Prema tome, uslov da podniz na ulazu kodera može biti jednak upravo dodatom nizu u kodnu tabelu je da su mu prvi i zadnji znak jednaki. Iz toga izvodimo sljedeće pravilo za dekodovanje: niz.pK proširujemo prvim znakom njega samog te taj niz pošaljemo na izlaz dekodera i dodamo u kodnu tabelu.

Algoritam za dekodovanje

1. Na početku kodna tabela sadrži sve moguće znakove koji se mogu pojaviti na ulazu;
2. tK := prva kodna riječ u ulaznom nizu (odgovara jednom od postijećih nizova u kodnoj tabeli);

3. U izlazni niz pošalji niz.tK (niz iz kodne tabele koji odgovara kodnoj riječi tK);
4. pK := tK;
5. tK := sljedeća kodna riječ u ulaznom nizu;
6. Postoji li tK u kodnoj tabeli?
    - ako postoji,
        1. u izlazni niz pošaljemo niz.tK;
        2. P := niz.pK;
        3. Z := prvi znak niz.tK;
        4. na prvo slobodno mjesto u rječniku dodamo niz P+Z;
    - ako ne postoji,
        1. P := niz.pK;
        2. Z := prvi znak niz.pK;
        3. niz P+Z pošaljemo u izlazni niz i dodamo ga u kodnu tabelu;
7. Ima li još kodova u ulaznom nizu?
    - ako ima, vrati se na korak 4;
    - ako nema, KRAJ.

Primjer dekodovanja:

Postupak dekodovanja

Korak	Kod	Izlaz	Kodna tabela
1.	(1)	A	--
2.	(2)	B	(4) A B
3.	(2)	B	(5) B B
4.	(4)	A B	(6) B A
5.	(7)	A B A	(7) A B A
6.	(3)	C	(8) A B A C

Analizirajmo korak **4.** Prethodnu kodnu riječ (2) smo spremili u pK, a tK je (4). Na izlaz šaljemo niz.tK, odnosno "A B". Nizu.pK ("B") dodajemo prvi znak niza.tK ("A") i to dodajemo kao novi string u kodnoj tabeli ("B A").

Sad slijedi korak **5.** U pK spremamo (4) i pročitamo novi tK = (7). To mjesto u kodnoj tabeli je još prazno. Stoga na niz.pK ("A B") dodajemo prvi znak njega samog, dakle "A" i taj niz ("A B A") dodajemo u kodnu tabelu pod brojem (7), dakle upravo onim koji smo pročitali. To znači da taj isti niz moramo i poslati na izlaz.

Ova situacija se javlja zbog toga što dekoder kasni u odnosu na koder. Kašnjenje se manifestuje u situaciji u kojoj se podniz koji je tek dodat u kodnu tabelu na strani kodera pojavi u ulaznom nizu neposredno nakon dodavanja u kodnu tabelu. U tom slučaju dekoder će susresti nepoznatu kodnu riječ. Međutim, prethodno kodovani podniz je prefiks podniza koji odgovara nepoznatoj kodnoj riječi, a prvi i poslednji znak tog podniza su jednaki. Dekoder jednostavno kopira prvi znak prethodno dekodiranog podniza i dobijenim podnizom proširuje kodnu tabelu te ga šalje u izlazni niz.

Faktor kompresije LZW algoritma primijenjenog na sive slike zavisi od broja nivoa svjetline u originalnoj slici i kreće se od 1.5 do 3. Najbolji faktori kompresije se postižu

kod slika sa malim brojem nivoa svjetline, posebno kod binarnih slika, kod kojih faktor kompresije može dostići 5.

Ova metoda je vrlo popularna u praksi. Daljnje razrade dodaju promjenljivu dužinu kodne riječi niza (ovisno o trenutnoj veličini kodne tabele), brisanje starih nizova iz kodne tabele i sl. LZW algoritam i njegove brojne modifikacije osnova su mnogih praktičnih realizacija programa za kompresiju i dekompresiju binarnih datoteka, kao što su ZIP, ARJ i drugi. LZW algoritam je takođe inkorporiran i u mnoge formate za zapis slike sa kompresijom, kao što su TIFF i GIF format. Razlozi za ovako široku primenu LZW algoritma i njegovih modifikacija su što je kompresija brza, bez gubitaka, ima zadovoljavajući stepen kompresije, a može se primijeniti na slike sa bilo kojim brojem bita po pikselu.

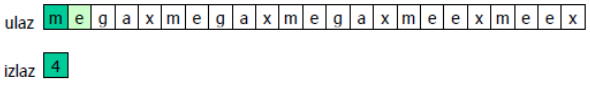
Primjer:

- alfabet: {a,e,g,m,x}
- tekst za kompresiju: megaxmegaxmegaxmeexmee
- početni sadržaj rečnika čine simboli alfabeta:

indeks	simbol
1	a
2	e
3	g
4	m
5	x

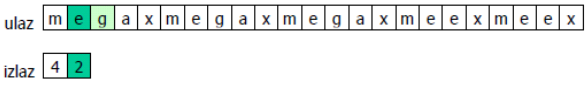
- Polazimo od početka teksta
- Posmatramo tekući simbol
  - postoji li *m* u rečniku? – postoji, sa indeksom 4
  - na tekući simbol dodamo naredni – *me*
    - postoji li sekvenca *me* u rečniku?
      - ne postoji
        - » dodajemo je na kraj rečnika
        - » na izlaz pišemo onaj kod koji smo poslednje pronašli, a to je indeks 4 za simbol *m*
        - » naredni simbol (*e*) postaje tekući

indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me



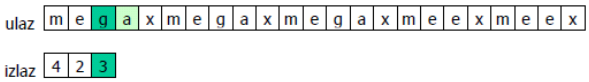
- Posmatramo tekući simbol
  - postoji li *e* u rečniku? – postoji, sa indeksom 2
  - na tekući simbol dodamo naredni – *eg*
    - postoji li sekvenca *eg* u rečniku?
      - ne postoji
        - » dodajemo je na kraj rečnika
        - » na izlaz pišemo onaj kod koji smo poslednje pronašli, a to je indeks 2 za simbol *e*
        - » naredni simbol (*g*) postaje tekući

indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me
7	eg



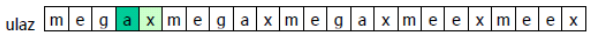
- Posmatramo tekući simbol
  - postoji li *g* u rečniku? – postoji, sa indeksom 3
  - na tekući simbol dodamo naredni – *ga*
    - postoji li sekvenca *ga* u rečniku?
      - ne postoji
        - » dodajemo je na kraj rečnika
        - » na izlaz pišemo onaj kod koji smo poslednje pronašli, a to je indeks 3 za simbol *g*
        - » naredni simbol (*a*) postaje tekući

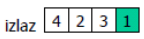
indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me
7	eg
8	ga



- Posmatramo tekući simbol
  - postoji li *a* u rečniku? – postoji, sa indeksom 1
  - na tekući simbol dodamo naredni – *ax*
    - postoji li sekvenca *ax* u rečniku?
      - ne postoji
        - » dodajemo je na kraj rečnika
        - » na izlaz pišemo onaj kod koji smo poslednje pronašli, a to je indeks 1 za simbol *a*
        - » naredni simbol (*x*) postaje tekući

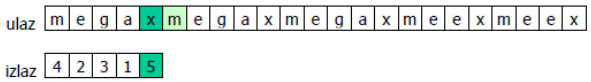
indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me
7	eg
8	ga
9	ax





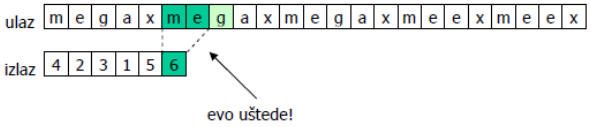
- Posmatramo tekući simbol
  - postoji li *x* u rečniku? – postoji, sa indeksom 5
  - na tekući simbol dodamo naredni – *xm*
    - postoji li sekvenca *xm* u rečniku?
      - ne postoji
        - » dodajemo je na kraj rečnika
        - » na izlaz pišemo onaj kod koji smo poslednje pronašli, a to je indeks 5 za simbol *x*
        - » naredni simbol (*m*) postaje tekući

indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me
7	eg
8	ga
9	ax
10	xm



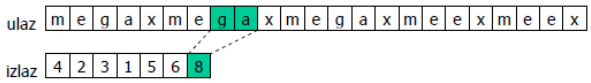
- Posmatramo tekući simbol
  - postoji li *m* u rečniku? – postoji, sa indeksom 4
  - na tekući simbol dodamo naredni – *me*
    - postoji li sekvenca *me* u rečniku? – postoji, sa indeksom 6
  - na tekuću sekvencu dodamo naredni simbol – *meg*
    - postoji li sekvenca *meg* u rečniku?
      - ne postoji
        - » dodajemo je na kraj rečnika
        - » na izlaz pišemo onaj kod koji smo poslednje pronašli, a to je indeks 6 za sekvencu *me*
        - » naredni simbol (*g*) postaje tekući

indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me
7	eg
8	ga
9	ax
10	xm
11	meg

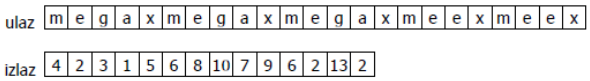


- Posmatramo tekući simbol
  - postoji li *g* u rečniku? – postoji, sa indeksom 3
  - na tekući simbol dodamo naredni – *ga*
    - postoji li sekvenca *ga* u rečniku? – postoji, sa indeksom 8
  - na tekuću sekvencu dodamo naredni simbol – *gax*
    - postoji li sekvenca *gax* u rečniku?
      - ne postoji
        - » dodajemo je na kraj rečnika
        - » na izlaz pišemo onaj kod koji smo poslednje pronašli, a to je indeks 8 za sekvencu *ga*
        - » naredni simbol (*x*) postaje tekući

indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me
7	eg
8	ga
9	ax
10	xm
11	meg
12	gax



- Nekoliko sati kasnije...



indeks	simbol
1	a
2	e
3	g
4	m
5	x
6	me
7	eg
8	ga
9	ax
10	xm
11	meg
12	gax
13	xme
14	ega
15	axm
16	mee
17	ex
18	xmee

Prediktivno kodovanje bez gubitaka

Kako postoji visok stepen korelacije susjednih piksela u slici, vrijednost nekog piksela je moguće uspješno procijeniti na osnovu vrijednosti susjednih piksela. Histogram razlike između originalne i procijenjene slike imaće vrlo mali broj popunjenih nivoa i biće koncentrisan u oblasti oko nule. Za kodovanje ovih razlika umjesto originalne moguće je koristiti kratke kodne riječi, a zbog vrlo neravnomjernog histograma može se efikasnije primjeniti i Hafmanovo kodovanje.

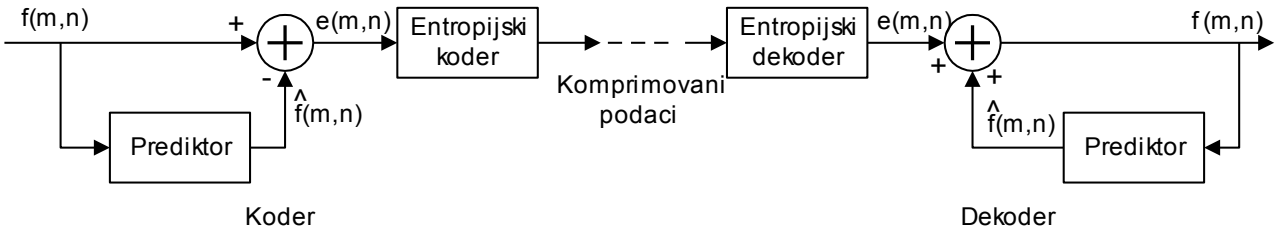
Prediktivne tehnike se najčešće primjenjuju za kompresiju slike sa gubicima, ali se mogu primijeniti i za kompresiju slike bez gubitaka, Slika 109.

Blok za predikciju vrši procjenu vrijednosti tekućeg piksela na osnovu vrijednosti susjednih piksela, koji se nalaze unutar nekog prozora W:

$$\hat{f}(m,n) = P\{f(m-i,n-j)\},\ i,j \in W$$

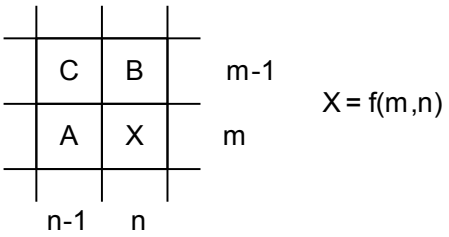
Zatim se izračunava greška predikcije:

$$e(m,n) = f(m,n) - \hat{f}(m,n)$$



Slika 109. Prediktivno kodovanja slike bez gubitaka

Raspored piksela u dvodimenzionalnom prozoru W koji se koristi za predikciju najčešće izgleda ovako:



Histogram greške predikcije koja se dobije oduzimanjem procijenjene vrijednosti svakog piksela od odgovarajuće stvarne vrijednosti ima vrlo mali broj nivoa i vrlo izražen maksimum oko nulte vrijednosti, te se može kodovati Hafmanovim koderom, čime se ostvaruje kompresija podataka.

Rekonstruisana slika se dobije inverznim operacijama. Dekodovanjem se dobija rekonstruisana slika greške predikcije. Iz greške predikcije i već rekonstruisanih podataka o pikselima u istoj i prethodnoj liniji, primjenom istovjetnog prediktora se rekonstruiše cijela slika.

Ovom metodom se postižu kompresije sa faktorima vrijednosti oko 2. Zbog svoje jednostavnosti, ova metoda prediktivnog kodovanja bez gubitaka je postala dio JPEG standarda za kompresiju slika.

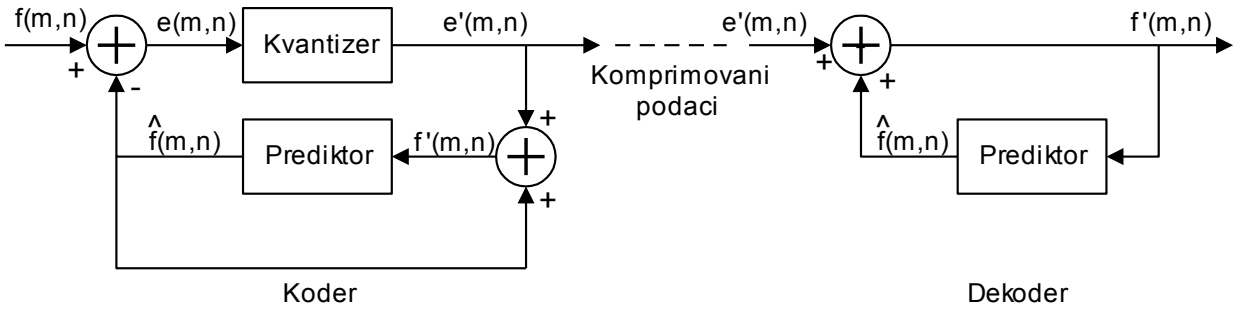
KOMPRESIJA SA GUBICIMA

Prediktivno kodovanje sa gubicima

Ako u blok šemu prediktivnog kodera slike bez gubitaka dodamo kvantizer za kvantovanje greške predikcije, dobićemo osnovnu šemu prediktivnog kodera slike sa gubicima, Slika 110.

Na taj način neizbježno unosimo greške u slici dobivenoj poslije rekonstrukcije. Ovakav način kompresije je pogodniji za slike sa velikim brojem nivoa svjetline.

Prediktivni metod kodovanja slike sa gubicima je rekurzivan, a naziva se diferencijalna impulsna kodna modulacija (Differential Pulse Code Modulation – DPCM).



Slika 110. Blok šema kodera i dekodera za postupak diferencijalne impulsne kodne modulacije (DPCM) slike.

Jednačine kojima je opisan rad prediktivnog kodera slike sa gubicima su:

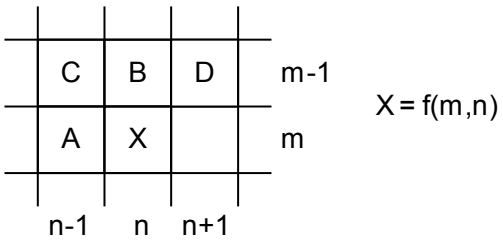
$$\hat{f}(m,n) = P\{f'(m-i,n-j)\},\ i,\ j \in W$$

$$e(m,n) = f(m,n) - \hat{f}(m,n)$$

$$e'(m,n) = Q[e(m,n)]$$

$$f'(m,n) = e'(m,n) + \hat{f}(m,n)$$

gde je W prozor koji obuhvata piksele iz iste i prethodne linije slike:



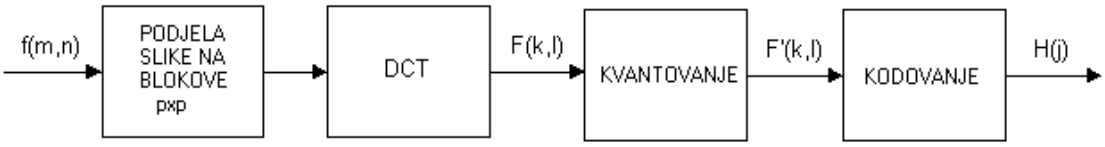
Transformacione tehnike za kompresiju slike

Transformacione tehnike kodovanja spadaju u klasu tehnika kodovanja sa gubicima. Nakon izvršene transformacije energija slike je sadržana u malom broju transformacionih koeficijenata, koje treba kvantovati i kodovati. Princip kodovanja se zasniva na umanjenju uticaja određenih transformacionih koeficijenata. Primjenom transformacionih tehnika se mogu ostvariti znatno bolji rezultati nego prediktivnim tehnikama, ali po cijenu veće računske složenosti algoritama.

Od svih poznatih unitarnih transformacija najbolje rezultate pri kodovanju daje diskretna kosinusna transformacija (DCT). DCT je realna i separabilna transformacija što umanjuje složenost izračunavanja transformacije slike. Moguće je koristiti brze algoritme kojima se broj operacija za računanje transformacije vektora od N elemenata smanjuje sa N^2 na $N \log_2 N$.

Kompresija slike pomoću DCT se realizuje u nekoliko koraka, Slika 111:

1. slika se podijeli na blokove dimenzija $p \times p$;
2. izračuna se DCT transformacija pojedininih blokova;
3. izvrši se kvantovanje DCT koeficijenata;
4. koduju se kvantovani DCT koeficijenti.



Slika 111. Blok šema kompresije slike pomoću diskretne kosinusne transformacije

Zbog velike složenosti izračunavanja DCT cijele slike dimenzija $M \times N$, slika se dijeli na blokove dimenzija $p \times p$. U praksi se najčešće koriste blokovi dimenzija 8×8, 16×16 i 32×32 piksela. Na taj način se smanjuje potrebna brza memorija za implementaciju transformacije sa $M \times N$ na p^2 i broj operacija sa $MN \log_2 MN$ na $2p^2 \log_2 p$. Za slike dimenzija 512×512 piksela podjelom na blokove od 16×16 piksela, potrebna memorija se smanji 1024 puta, a broj operacija 2.25 puta. Ako se računanje DCT realizuje hardverski ili

na višeprocesorskim sistemima, moguće je paralelno izračunavanje transformacija više blokova čime se još više smanjuje vrijeme potrebno za izračunavanje DCT. Međutim, podjela slike na blokove ima i svoje loše strane. Kao prvo, zasebnom kompresijom blokova nije moguće redukovati korelaciju piksela koji pripadaju različitim blokovima, te je stepen kompresije manji nego kada se vrši transformacija cijele slike. Osim toga, podjela slike na blokove izaziva tzv. blokovski efekat, tj. pojavu vidljivih granica između blokova u rekonstruisanoj slici.

Dakle, nakon podjele slike na blokove vrši se diskretna kosinusna transformacija pojedinačnih blokova definisana sa:

$$F[k,l]=\frac{\alpha[k]\alpha[l]}{p}\sum_{m=0}^{p-1}\sum_{n=0}^{p-1}f[m,n]\cos\frac{\pi k(2m+1)}{2p}\cos\frac{\pi l(2n+1)}{2p},\ 0\le k,l\le p-1$$

gde su $\alpha[j]$ definisani kao:

$$\alpha[j]=\begin{cases}1/\sqrt{2}, & j=0,\\ 1 & 1\le j\le p-1.\end{cases}$$

a p je određeno veličinom blokova i može biti p = 8, 16 ili 32.

Pri računanju DCT koriste se algoritmi za brzo računanje transformacije, a osobina separabilnosti omogućava računanje 2D DCT sukcesivnim 1D transformacijama vrsta i kolona slike, što još više smanjuje broj neophodnih operacija.

Transformacione metode kompresije se zasnivaju na odbacivanju transformacionih koeficijenata čija je vrijednost mala. Na taj način se postiže znatan stepen kompresije, jer se ti koeficijenti niti pamte niti prenose. DCT koeficijenti se kvantuju pomoću kvantizacionih tabela, koje redukuju amplitude koeficijenata koji malo ili nimalo utiču na kvalitet slike, s ciljem da se poveća broj koeficijenata sa nultom vrijednošću:

$$F_q(k,l)=Round\left[\frac{F(k,l)}{Q(k,l)}\right]$$

Jedna tipična kvantizaciona tabela za blok od 8×8 piksela je data na Slici 112:

8	6	5	8	12	20	26	30
6	6	7	10	13	29	30	28
7	7	8	12	20	29	35	28
7	9	11	15	26	44	40	31
9	11	19	28	34	55	52	39
12	18	28	32	41	52	57	46
25	32	39	44	52	61	60	51
36	46	48	49	56	50	52	50

Slika 112. Tipična kvantizaciona tabela za blok od 8×8 piksela

Uglavnom je potrebno prenositi samo one DCT koeficijente koji se nalaze u gornjem lijevom uglu matrice transformacionih koeficijenata. Međutim, i ti koeficijenti se mogu međusobno razlikovati i za više redova veličine. Da bi se još više povećao stepen kompresije, koristi se nejednak broj bita za njihovo predstavljanje.

Prije kodovanja se od kvantovanih DCT koeficijenata formira jednodimenzionalni niz tako što se vrijednosti DCT koeficijenata očitavaju po cik-cak redosljedu. Ovakav niz je pogodan za entropijsko kodovanje jer se koeficijenti veće vrijednosti nalaze na početku niza. Najčešće se koristi Hafmanovo kodovanje.

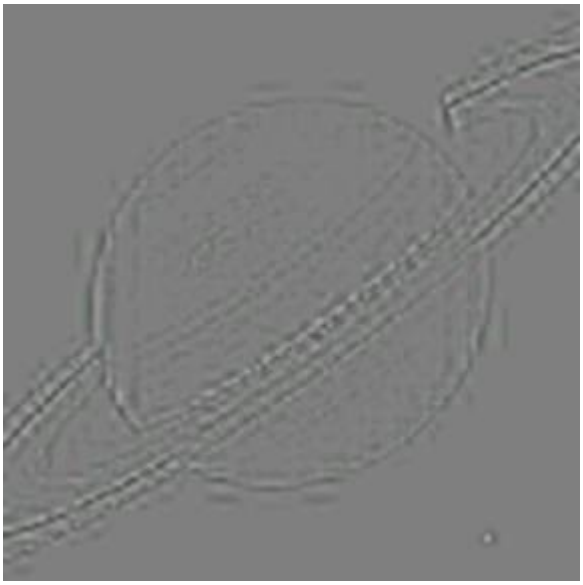
Transformacione tehnike kompresije unose nekoliko vrsta degradacija slike: gubitak detalja zbog eliminacije visokofrekventnih DCT koeficijenata, granularnost koja se manifestuje pojavom zrnaste strukture na površinama relativno uniformne osvjetljenosti kao posljedica grube kvantizacije DCT koeficijenata i blokovski efekat pri većim stepenima kompresije.

Ipak, i pored navedenih loših osobina, transformacione tehnike kompresije se mnogo koriste jer se uz prihvatljivi stepen degradacije u rekonstruisanoj slici može postići stepen kompresije u granicama od 4 do 8.

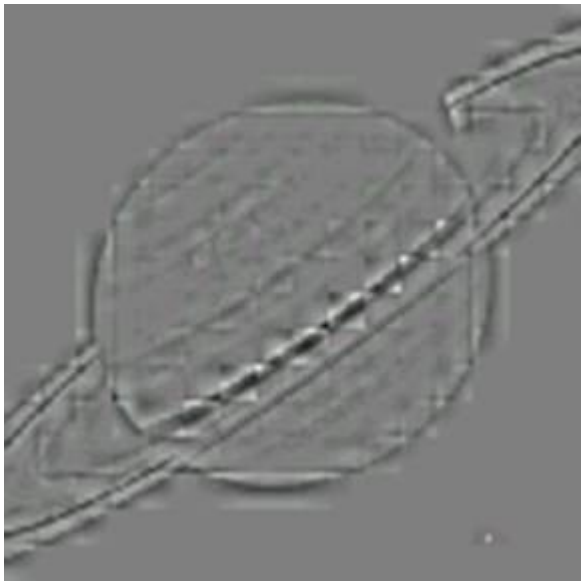
Primjeri rekonstrukcije slike na osnovu malog broja DCT dati su na slikama 113-115.

originalna slika Saturn (512x512 piksela),

rekonstruisana slika na osnovu 20 DCT koeficijenata (32x32 blok), slika greške



rekonstruisana slika na osnovu 12 DCT koeficijenata (32x32 blok), slika greške



rekonstruisana slika na osnovu 4 DCT koeficijenta (32x32 blok), slika greške

Vremenom se pokazalo da je neophodno izvršiti standardizaciju metoda za kompresiju slike što bi omogućilo komunikaciju između korisnika koji koriste opremu i softver različitih proizvođača. Pokušaji velikih proizvođača opreme i softvera da nametnu neka svoja rešenja kao standarde nisu uspjeli, iako se i ta rješenja koriste u praksi (npr. GIF i TIFF grafički formati). Međutim, njihova primjena na kompresiju prirodnih slika, gray-scale ili u boji, ne daje dobre rezultate, jer su oni prvenstveno namijenjeni kompresiji crteža i tekstualnih dokumenata.

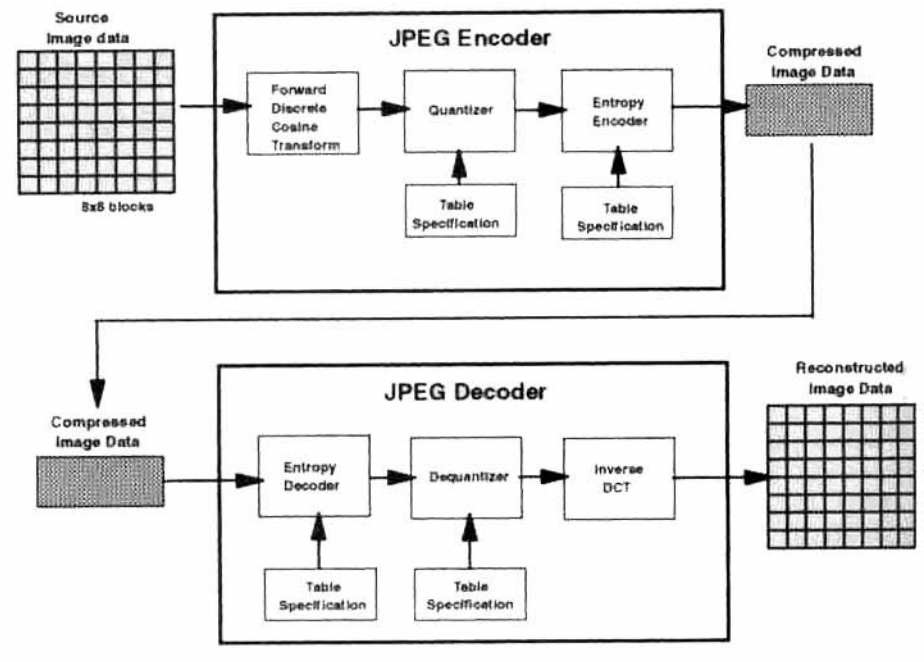
Internacionalna organizacija za standarde (ISO), Internacionalna elektrotehnička komisija (IEC) i Internacionalna telekomunikaciona unija (ITU) rade na standardizaciji metoda, hardvera i softvera za multimedijalne sisteme, videokonferencije, videotelefoniju i slične aplikacije. Do sada se pojavilo više standarda (JPEG za kompresiju mirnih slika, MPEG-1, MPEG-2, MPEG-4, H.261, H.263, za kompresiju video sekvenci, itd.).

JPEG standard

JPEG (Joint Photographic Experts Group) je standard za kompresiju mirnih slika sa više nivoa svjetline, kao i slika u boji. To je zajednički standard tri međunarodne organizacije: ISO, IEC i ITU. On omogućava kompresiju bilo kakve digitalne slike, sive ili u boji, sa gubicima ili bez gubitaka, nezavisno od njene rezolucije. Postoje četiri načina kompresije po JPEG standardu:

- *Kodovanje bez gubitaka*
- *Sekvencijalno DCT kodovanje*
- *Progresivno DCT kodovanje*
- *Hijerarhijsko kodovanje*

Mi ćemo pažnju posvetiti, najčešće korištenom, sekvencijalnom kodovanju. Blok dijagram sekvencijalnog kodera i dekodera je prikazan na Slici 116.



Slika 116. Blok šema JPEG sekvencijanog kodera i dekodera

JPEG koder

Koder se sastoji od tri osnovna bloka:

1. Blok za računanje direktne diskretne kosinusne transformacije (FDCT),
2. Kvantizer,
3. Entorpijski koder.

Na ulazu kodera, originalne vrijednosti piksela, koje su pozitivni cijeli brojevi iz opsega $[0, 2^p-1]$, se pomjeraju u opseg $[-2^{p-1}, 2^{p-1}-1]$. Na primjer, za gray-scale slike kod kojih je p=8, originalni odmjerci iz opsega [0, 255] se oduzimanjem 128 pomjeraju u opseg [-128, +127].

Zatim se slika dijeli na blokove dimenzija 8x8. Dimenzija blokova je odabrana na bazi opsežnih ispitivanja subjektivnog osjećaja kvaliteta komprimovane slike sa različitim dimenzijama blokova. Svaki od blokova se transformiše diskretnom kosinusnom transformacijom u skup od 64 DCT koeficijenata koji predstavljaju "dvodimenzionalni spektar" ulaznog signala:

$$F[k,l]=\frac{\alpha[k]\alpha[l]}{p}\sum_{m=0}^{p-1}\sum_{n=0}^{p-1}f[m,n]\cos\frac{\pi k(2m+1)}{2p}\cos\frac{\pi l(2n+1)}{2p},\ 0\le k,l\le p-1$$

gdje je $p = 8$ i

$$\alpha[j]=\begin{cases}1/\sqrt{2}, & j=0,\\ 1 & 1\le j\le p-1.\end{cases}$$

Diskretna kosinusna transformacija je diskretna funkcija dvije prostorne dimenzije, k i l, koje nazivamo prostornim frekvencijama. Vrijednosti ove funkcije nazivamo DCT koeficijentima. Koeficijent $F(0,0)$ se naziva DC koeficijent, dok se preostala 63 koeficijenta nazivaju AC koeficijenti. Za sive slike DCT koeficijenti su iz opsega vrijednosti [-1024, 1023], što znači da u poređenju sa originalnim vrijednostima piksela slike trebamo 3 bita više za njihov zapis.

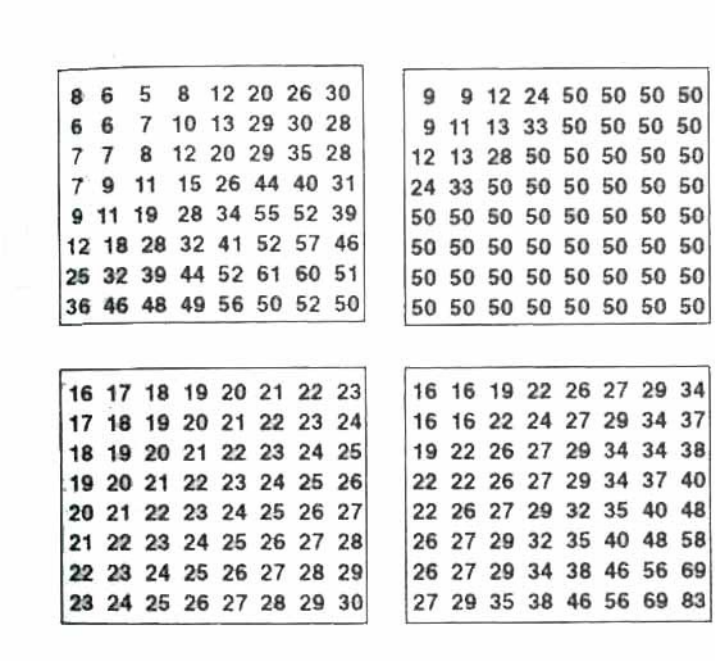
Za tipičan 8x8 blok većina DCT koeficijenata je jednaka nuli, ili ima vrijednost blisku nuli, odnosno znatno manju od koeficijenata na niskim prostornim frekvencijama, Slika 117, te ih ne treba kodovati. Ova osobina se koristi za postizanje kompresije slike.

140	144	147	140	140	155	179	175	185	-17	14	-8	23	-9	-13	-18
144	152	140	147	140	148	167	179	20	-34	26	-9	-10	10	13	6
152	155	136	167	163	162	152	172	-10	-23	-1	6	-18	3	-20	0
168	145	156	160	152	155	136	160	-8	-5	14	-14	-8	-2	-3	8
162	148	156	148	140	136	147	162	-3	9	7	1	-11	17	18	15
147	167	140	155	155	140	136	162	3	-2	-18	8	8	-3	0	-6
136	156	123	167	162	144	140	147	8	0	-2	3	-1	-7	-1	-1
148	155	136	155	152	147	147	136	0	-7	-2	1	1	4	-6	0

Slika 117. Matrica vrijednosti originalne slike i DCT koeficijenti

U sljedećem bloku, kvantizeru, 64 DCT koeficijenta se kvantuju koristeći kvantizacione matrice (zavisno od aplikacije) koje redukuju amplitudu koeficijenata koji malo ili nimalo ne utiču na kvalitet slike, s ciljem da se poveća broj nultih vrijednost u matrici DCT koeficijenata. Kvantizacijom se gube informacije koje nisu vizualno od značaja. Kvantovanje se vrši dijeljenjem svakog DCT koeficijenta sa odgovarajućim elementom neke od kvantizacionih matrica (Slika 118) i zaokruživanjem na najbliži cio broj:

$$F_q(k,l) = Round\left[\frac{F(k,l)}{Q(k,l)}\right].$$

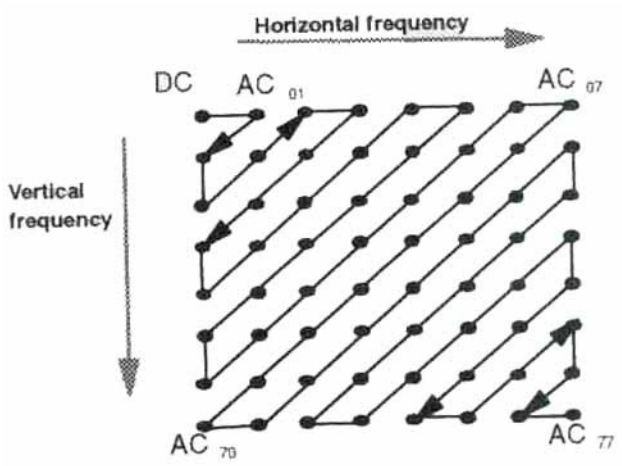


Slika 118. Standardne kvantizacione matrice

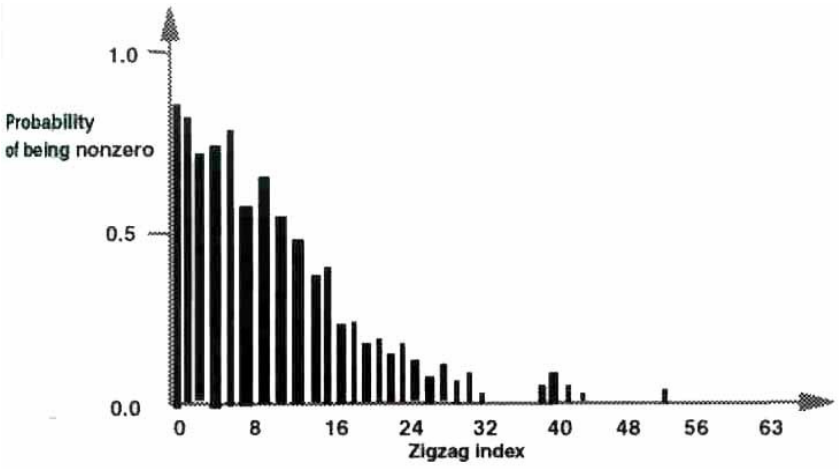
Kvantovanje se vrši sa različitim korakom kvantovanja za svaki DCT koeficijent. Za maksimalnu kompresiju slike bez vidljivih izobličenja, svaka vrijednost u kvantizacionoj matrici mora da bude izabrana tako da predstavlja prag percepcije odgovarajuće kosinusne bazisne funkcije. Sproveden je niz psihovizuelnih eksperimenata za određivanje najboljih vrednosti pragova. Sam standard ne propisuje sadržaj kvantizacionih matrica, ali daje određene tabele kao preporuke.

Za regulaciju stepena kompresije uveden je faktor kvaliteta Q, cio broj kojim se množe elementi kvantizacione matrice. Tako se omogućava jednostavno, ali grubo, zadavanje željenog stepena kompresije, pri čemu veće vrijednosti parametra Q odgovaraju grubljem kvantovanju, odnosno većem stepenu kompresije. Većem stepenu kompresije odgovara lošiji kvalitet rekonstruisane slike.

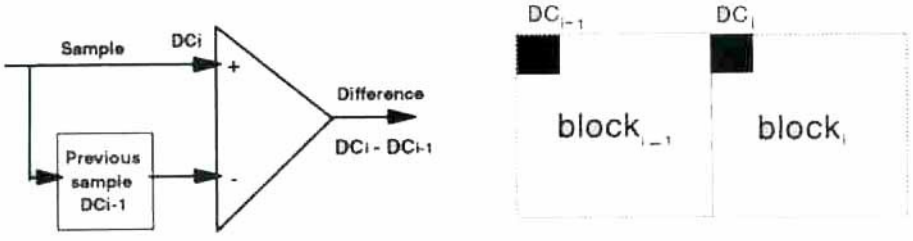
Nakon kvantizacije, od 63 AC koeficijenta se formira jednodimenzionalna sekvenca, cik-cak redoslijedom, kao na Slici 119. Ovakav poredak koeficijenata olakšava entropijsko kodovanje jer su niskofrekventni koeficijenti, čija je vjerovatnoća da su različiti od nule veća nego kod visokofrekventnih, smješteni na početak. Ova činjenica je potvrđena nizom eksperimentalnih ispitivanja. Dobijeni rezultati pokazuju da je, u slučaju cik-cak poretka DCT koeficijenata, vjerovatnoća da su koeficijenti različiti od nule monotono opadajuća funkcija indeksa, Slika 120. DC koeficijenti, koji predstavljaju srednju vrijednost piksela u bloku dimenzija 8x8, koduju se prediktivnim tehnikama, Slika 121, jer postoji velika prostorna korelacija između srednjih vrijednosti svjetline u susjednim blokovima. Zbog toga se kodovanjem razlike vrijednosti DC koeficijenata postiže veći stepen kompresije nego kodovanjem samih koeficijenata.



Slika 119. Cik-cak poredak AC koeficijenata



Slika 120. Vjerovatnoća da su DCT koeficijenti različiti od nule



Slika 121. Prediktivno kodovanje DC koeficijenata

Konačno, poslednji blok JPEG kodera je entropijski koder, kojim se postiže dodatna kompresija kodovanjem kvantovanih DCT koeficijenata. Nakon kvantovanja, Hafmanov koder konvertuje DCT koeficijente u kompaktnu binarnu sekvencu kroz dva koraka: (1)

formira se tabela simbola, (2) na osnovu Hafmanove tabele, simboli se konvertuju u binarnu sekvencu.

U tabeli simbola, svaki AC koeficijent se predstavlja parom simbola:

- (DUŽINA_NIZA, VELIČINA)
- (AMPLITUDA)

DUŽINA_NIZA je broj koji pokazuje koliko AC koeficijenata nultih vrijednosti prethodi AC koeficijentu nenulte vrijednosti. Vrijednost za DUŽINU_NIZA je u opsegu 0-15, te su potrebna 4 bita za njenu reprezentaciju.
VELIČINA je broj bita koji je potreban da se koduje AMPLITUDA. AMPLITUDA se koduje sa 0-10 bita, što znači da su neophodna 4 bita za kodovanje VELIČINE.
AMPLITUDA je vrijednost nenultih AC koeficijenata koja se kreće u opsegu [-1024, 1023], te je potrebno maksimalno 10 bita za njeno kodovanje.
Na primjer, ako je sekvenca AC koeficijenata:

0,0,0,0,0,0,476

simbol kojim se predstavlja nenulti koeficijent 476 je

(6,9)(476)

gdje je: DUŽINA_NIZA=6, VELIČINA=9 i AMPLITUDA=476.

Ako je DUŽINA_NIZA veća od 15, simbol (15,0) se interpretira kao da je DUŽINA_NIZA=16. Uzastopno se mogu pojaviti do tri ovakva simbola.

U sljedećem primjeru:

(15,0)(15,0)(7,4)(12)

DUŽINA_NIZA je jednaka 16+16+7=39, VELIČINA=4 i AMPLITUDA=12.

Simbolom (0,0) se završava svaki 8x8 blok.

Za DC koeficijente se tabela simbola sastoji od:

- VELIČINA
- AMPLITUDA

Kako se DC koeficijenti diferencijalno koduju, opseg vrijednosti je dvostruko veći nego kod AC koeficijenata, [-2048, 2047].

U sljedećem koraku se sekvenca simbola konvertuju u binarnu sekvencu. Simboli se zamjenjuju kodnim riječima promjenljive dužine, počevši od DC koeficijenta, a zatim AC koeficijenti u cik-cak poretku. Svaki simbol kojim je predstavljen AC ili DC koeficijent se koduje na osnovu Hafmanove tabele.

Na primjer, AC koeficijent predstavljen simbolom (1,4)(12) se koduje binarnom sekvencom (1111101101100), gdje je (111110110) kod za (1,4) očitan iz Hafmanove tabele (koja je preporučena JPEG standardom), slika 110, a (1100) je kod 12.

JPEG dekoder

Kod JPEG sekvencijalnog dekodovanja, svi koraci iz procesa kodovanja se implementiraju obrnutim redoslijedom, slika 103. Prvo se binarna sekvenca konvertuje u sekvencu simbola koristeći Hafmanovu tabelu, a zatim se simboli konvertuju u DCT koeficijente. Zatim se vrši dekvantizacija po sljedećoj formuli:

$$F_q^i(u,v) = F_q(u,v) \times Q(u,v)$$

gdje su $Q(u,v)$ kvantizacioni koeficijenti iz kvantizacione tabele.

Poslije toge se uradi inverzna diskretna kosinusna transformacija (IDCT), kojom se slika iz 2D frekventnog domena prevodi u prostorni domen. Inverzna diskretna kosinusna transformacija je definisana sa:

$$f[m,n] = \frac{1}{4}\sum_{k=0}^{7}\sum_{l=0}^{7}\alpha[k]\alpha[l]F[k,l]\cos\frac{\pi k(2m+1)}{2p}\cos\frac{\pi l(2n+1)}{2p}$$

gdje je:

$$\alpha[j] = \begin{cases} 1/\sqrt{2}, & j = 0, \\ 1 & 1 \le j \le p-1. \end{cases}$$

Posljednji korak predstavlja pomijeranje dekomprimovanih odmjeraka u opseg $[0, 2^p-1]$.

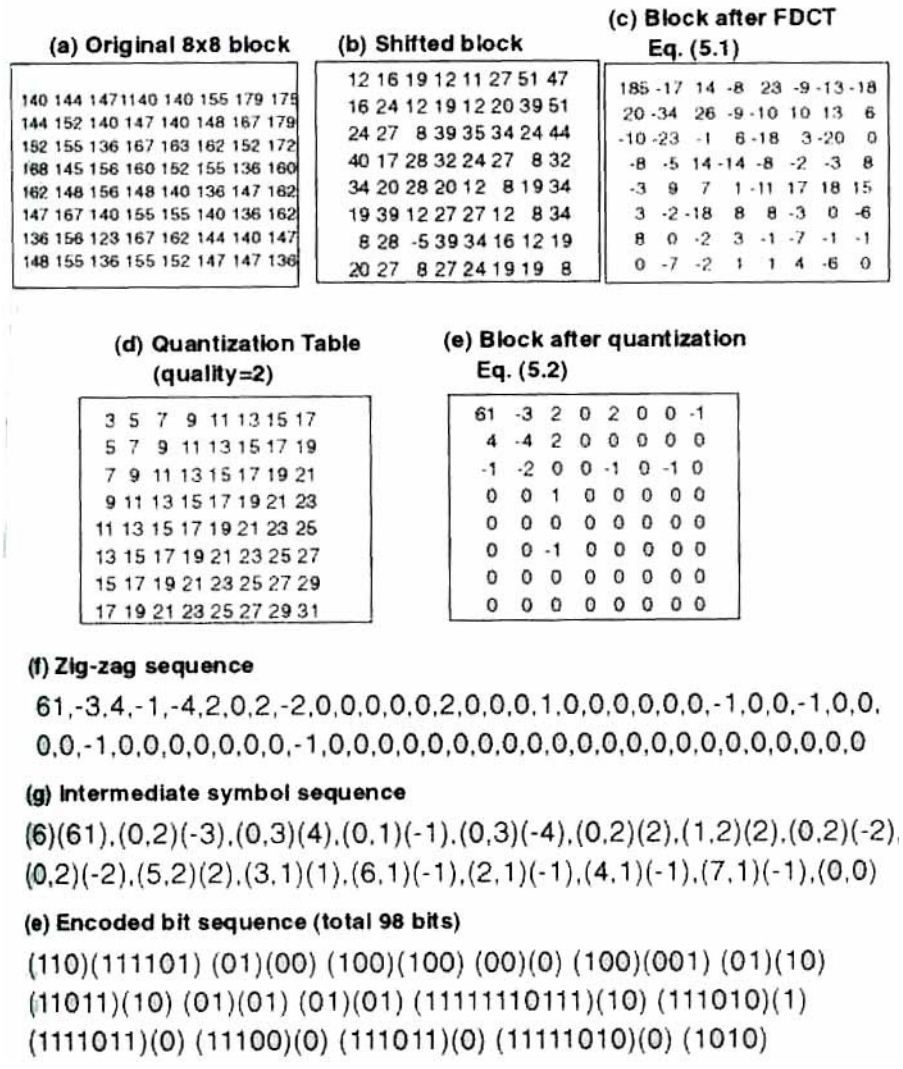
Primjer

Za ilustraciju principa sekvencijalnog JPEG kodovanja, prikazaćemo proceduru korak po korak na jednom 8x8 bloku 8-bitnih uzoraka. Na slici 109 (a) su prikazane vrijednosti originalnog bloka (opseg vrijednosti je [0, 255]), a na slici 109 (b) je isti blok nakon pomijeranja u opseg [-128, +127]. Primjenom diskretne kosinusne transformacije dobiju se DCT koeficijenti prikazani na slici 109 (c). Primijetimo da su skoro sve vrijednosti koeficijenata, izuzev onih na niskim frekvencijama, bliske nuli.

Za generisanje kvantizacione tabele se preporučuje sljedeći program:

```
for(i = 0; i <  N; i++)
    for(j = 0; j <  N; j++)
        Q(i,j) = 1+[(1+i+j) x Quality];
```

Pri tome faktor *Quality* specificira kvalitet rekonstruisane slike. *Quality* = 1 daje najbolji kalitet, ali i najniži stepen kompresije. U ovom primjeru je korišten faktor kvaliteta 2, i kvantizaciona tabela je prikazana na Slici 122 (d). Kvantovani DCT koeficijenti su prikazani na Slici 122 (e). Primijetimo da je većina AC koeficijenata na višim frekvencijama jednaka nuli. Sada se od DCT koeficijenata formira cik-cak sekvenca, prikazana na Slici 122 (f). Koeficijenti različiti od nule su grupisani na početku sekvence, a zatim slijede duge sekvence nula. Zbog načina generisanja simbola, postiže se veća kompresija nego kad bi imali kraće sekvence nula isprekidane vrijednostima različitim od nule. Tabela simbola je prikazana na Slici 122 (g). Konačno, nakon implementacije Hafmanovog kodera, binarna sekvenca je prikazana na Slici 122 (e).



Slika 122. [16] Ilustracija JPEG kodovanja

Dio Hafmanove tabele je prikazan na slici 123.

Zero Run	Category	Code length	Codeword
0	1	2	00
0	2	2	01
0	3	3	100
0	4	4	1011
0	5	5	11010
0	6	6	111000
0	7	7	1111000
.	.	.	.
.	.	.	
1	1	4	1100
1	2	6	111001
1	3	7	1111001
1	4	9	111110110
.	.	.	.
.	.	.	.
2	1	5	11011
2	2	8	11111000
.	.	.	.
.	.	.	.
3	1	6	111010
3	2	9	111110111
.	.	.	.
.	.	.	.
4	1	6	111011
5	1	7	1111010
6	1	7	1111011
7	1	8	11111001
8	1	8	11111010
9	1	9	111111000
10	1	9	111111001
11	1	9	111111010
.	.	.	.
.	.	.	.
EOB		4	1010

Slika 123. Hafmanova tabela za JPEG kodovanje

Napomenimo da je DC koeficijent kodovan kao koeficijent prvog bloka, što znači da nije korišteno prediktivno kodovanje kao za DC koeficijente iz ostalih blokova, već je kodovan direktno.

Za ovaj primjer, stepen kompresije je:

$$C_R = \frac{\text{originalan broj bita}}{\text{kodovani broj bita}} = \frac{64 \times 8}{98} = \frac{512}{98} = 5.22$$

a broj bita po pikselu iznosi:

$$N_b = \frac{\text{kodovani broj bita}}{\text{broj piksela}} = \frac{98}{64} = 1.53$$

Mjera kompresije

Kad govorimo o preciznosti slike, mislimo na broj nivoa svjetline koje je moguće prikazati. Pri tome se preciznost izražava kao broj bita/pikselu. Na osnovu ovog parametra, može se izvršiti klasifikacija originalne (nekomprimovane) slika na:

- *binarne slike* predstavljene sa 1 bit/pikselu,
- *računarsku grafiku* sa 4 bita/pikselu,
- *grayscale* slike sa 8 bita/pikselu,
- *slike u boji* sa 16, 24 ili više bita/pikselu.

Osnovna mjera pri poređenju različitih algoritama za kompresiju je:

$$C_R = \frac{\text{originalna kolicina podataka}}{\text{kolicina podataka nakon kompresije}}$$

Prilikom kompresije slike mora se tražiti kompromisno rješenje između stepena kompresije i kvaliteta slike. Pri tome, odnos kompresije i kvaliteta varira zavisno od karakteristika slike i od toga šta se nalazi na sceni. Kao mjera kvaliteta slike usvaja se broj bita po pikselu u komprimovanoj slici, definisan kao odnos ukupnog broja bita u komprimovanoj slici i broja piksela:

$$N_b = \frac{\text{broj bita nakon kodovanja}}{\text{broj piksela}}$$

Druga statistička mjera koja se može koristiti pri razvoju algoritama za kompresiju slike je srednjekvadratna greška:

$$RMS = \frac{1}{n}\sqrt{\sum_{i=1}^{n}\left(X_i - \hat{X}_i\right)^2}$$

gdje je:

X_i – originalna vrijednost piksela,

$\hat{X}_i$ – vrijednost piksela nakon kompresije,

n – ukupan broj piksela.

U većini slučajeva, ali ne i uvijek, kvalitet slike je bolji ako je srednjekvadratna greška manja.

Primjer:

Na Slici 124 (a) je prikazana gray-scale slika *Cvijeće*, dimenzija 500×362 piksela i kodovana sa 8 bita/pikselu. Nakon JPEG kompresije, na Slici 124 (b) je predstavljena slika komprimovana 4 puta, na Slici 124 (c) je stepen kompresije jednak 10, dok je na Slici 124 (d) prikazana slika koja je komprimovana 22 puta. Tek kad stepen kompresije pređe 20 mogu se uočiti vidljiva izobličenja. Kod slika sa manje detalja stepen kompresije može biti i veći od 20 a da se izobličenja ne primjećuju. Međutim, kod slika sa mnogo detalja, (npr. otiska prsta), izobličenja se primjećuju već kad stepen kompresije pređe vrijednost 10.

(a) **(b)**

(c) **(d)**

Slika 124. [15] JPEG kompresija sive slike *Cvijeće*: (a) Originalna slika. (b) Slika komprimovana 4 puta. (c) Slika komprimovana 10 puta. (d) Slika komprimovana 22 puta.

KOMPRESIJA SLIKA U BOJI

Pri kodovanju slika u boji, prvo se izvrši transformacija *RGB→YCbCr*. Slika luminentne komponente *Y* se dijeli na blokove od 8×8 piksela. Slike hrominentnih komponenti *Cb* i *Cr* se dijele na blokove od 16×16 piksela, koji se zatim, decimacijom sa 2, svode na dimenzije 8×8 piksela. Tako na svaka četiri bloka *Y* komponente dolazi po jedan blok hrominentnih komponenti *Cb* i *Cr*. Daljnja kompresija se radi nezavisno za svaku od tri komponente slike *Y*, *Cb* i *Cr*.

Kompresijom slike u boji se može postići veći stepen kompresije nego kompresijom gray-scale slika, uz jednak subjektivni osećaj izobličenja slike. Na Slici 216 je prikazana slika *Cvijeće* u boji sa tri različita stepena kompresije, koji su približno dva puta veći od odgovarajućih stepena kompresije gray-scale slike.

(a) **(b)**

(c) **(d)**

Slika 216. [15] JPEG kompresija slike u boji *Cvijeće*: (a) Originalna slika, (b) Slika komprimovana 10 puta, (c) Slika komprimovana 20 puta, (d) Slika komprimovana 43 puta.