VEŽBA 6 – Decimacija i interpolacija

Potrebno predznanje

- Poznavanje programskog jezika C
- FIR filtri

Šta će biti naučeno tokom izrade vežbe

U okviru ove vežbe naučićete:

- Kako se vrši promena učestanosti odabiranja već odabranog signala
- Kako implementirati operacije decimacije i interpolacije digitalnih signala
- Čemu služe decimacioni odnosno interpolacioni filtri
- Kako izvršiti promenu učestanosti odabiranja digitalnog signala sa racionalnim faktorom L/M.
- Kakva je prednost obrade signala na različitim učestanostima odabiranja

Motivacija

Promena učestanosti odabiranja predstavlja veoma bitan process u obradi signala, koji nam omogućava da menjamo vremensku, odnosno prostornu rezoluciju već odabranog signala. Kod audio signala, promena učestanosti se koristi za prilagođavanje signala sistemima za reprodukciju koji koriste različitu izlaznu učestanost odabiranja od učestanosti signala. U obradi slike i video signala, ove operacije omogućavaju promenu rezolucije slike, odnosno skaliranje. Promena učestanosti odabiranja omogućava obradu signala na različitim učestanostima. Ovakva obrada podrazumeva promenu učestanosti odabiranja, primenu algoritama obrade signala i potom vraćanje učestanosti odabiranja na inicijalnu. Ovakva obrada omogućava efikasniju implementaciju algoritama, s obzirom da obrada signala predstavljenog sa manjim brojem odbiraka zahteva manje resursa.

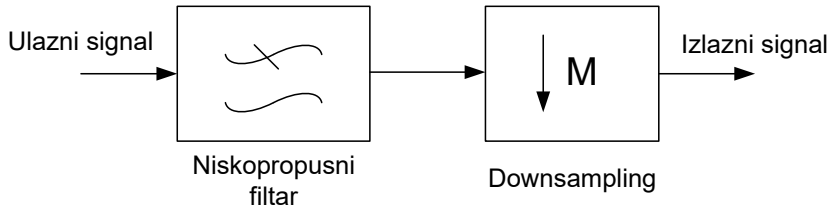
1 Decimacija

Uopšteno govoreći, decimacija je proces redukcije frekvencije odabiranja ulaznog signala za neki celobrojni faktor. Faktor decimacije predstavlja odnos frekvencije odabiranja ulaznog signala i frekvencije odabiranja signala nastalog decimacijom. On se najčešće obeležava sa M i predstavlja sledeći količnik:

$$M = \frac{f_{si}}{f_{so}}$$

Decimacija se u praksi obično realizuje filtriranjem ulaznog signala niskopropusnim filtrom i potom odbacivanjem određenog broja odbiraka (Slika 1). Odbacivanje odbiraka (engl. downsampling) odnosi se na proces prostog odbacivanja odbiraka ulaznog signala bez filtriranja. Ulazni signal može biti decimiran na ovaj način ukoliko je signal odabiran sa frekvencijom odabiranja dovoljno većom od one koja je potrebna da bi se signal mogao rekonstruisati (Nikvistov uslov). Ukoliko ovaj uslov nije ispunjen, doći ce do pojave preslikavanja spektra (engl. aliasing). Zbog toga je u takvim slučajevima neophodno filtrirati signal niskopropusnim filtrom pre odbacivanja odbiraka. Granična frekvencija ovog filtra treba da odgovara polovini frekvencije odabiranja decimiranog signala:

$$f_c = \frac{f_{so}}{2} = \frac{f_{si}}{2M}$$



Slika 1 - Decimacija signala za faktor M

Najočigledniji razlog upotrebe decimacije je da se smanji frekvencija odabiranja na izlazu jednog sistema, da bi se taj izlaz mogao proslediti sistemu koji radi na nižoj frekvenciji odabiranja. Mnogo značajnija motivacija za primenu decimacije je smanjenje cene obrade (po broju instrukcija i količini memorije) implementacije na DSP sistemima. Broj instrukcija i količina memorije potrebna za obradu obično je proporcionalna frekvenciji odabiranja ulaznog signala, tako da implementacija na nižim frekvencijama odabiranja rezultuje jeftinijom implementacijom. Niskopropusni fitar može biti bilo FIR, bilo IIR tipa. Odbacivanje odbiraka se svodi na zadržavanje svakog M-tog odbirka filtriranog signala.

Ukoliko faktor decimacije nije prost broj, decimacija se može izvršiti u nekoliko uzastopnih faza. Npr. ukoliko je faktor decimacije 24, decimacija se može izvršiti na sledeće načine:

- jedna faza: M = 24

- dve faze: M = 6 i M = 4, ili M = 8 i M = 3
- tri faze: M = 4, M = 3 i M = 2
- četiri faze: M = 3, M = 2, M = 2, M = 2

Implementacija odbacivanja odbiraka u programskom jeziku C, može se na jednostavan način realizovati koristeći jednu petlju, koja prolazi kroz čitav ulazni niz sa korakom koji odgovara faktoru decimacije i pročitane odbirke upisuje redom u izlazni niz. U sledećem primeru data je funkcija koja obavlja odbacivanje odbiraka. Niz sa ulaznim signalom nalazi se u promenljivoj *input*, niz za izlazni signal u promenljivoj *output*, dok *M* i *N* predstavljaju redom faktor decimacije i veličinu ulaznog niza.

```
void downsample(Int16 *input, Int16 *output, int M, int N)
{
	Int16 i, j;

	for(i = 0, j = 0; i< N; i+=M, j++)
	{
		output[j] = input[i];
	}
}
```

Kao što je već rečeno, celokupan postupak decimacije sastoji se iz filtriranja signala odgovarajućim niskopropusnim filterom i primene operacije odbacivanja odbiraka. Primer:

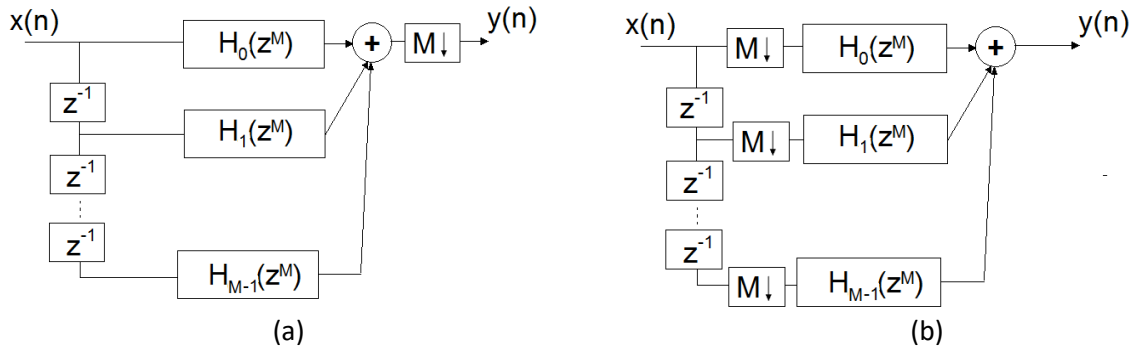
```
	for(i = 0; i< N; i++)
	{
		filtered[i] = fir_circular(input[i], lp_filter, history, N_COEFF,
&state);
	}

	downsample(filtered, output, M, N);
```

S obzirom da se nakon filtriranja niskopropusnim filterom vrši odbacivanje svih odbiraka osim svakog M-tog, zaključuje se da kada se primeni operacija decimacije na ovakav način, za signal dužine N i faktor decimacije M, samo L/M filtriranih odbiraka se zadržava, dok se ostatak odbiraka odbacuje, odnosnu njihovo računanje je bilo nepotrebno. Kako bi se izbeglo nepotrebno računanje, moguće je primeniti tzv. polifaznu implementaciju FIR filtra za decimaciju. Polifazno filtriranje podrazumeva razlaganje niskopropusnog FIR filtera na M filtera nižeg reda (slika 2a), tako da svaki od filtera sadrži koeficijente:

$$h_n(i) = h(iM + n)$$

Na ulaz u svaki od filtera dovodi se signal zakašnjen za jedan odbirak u odnosu na ulazu u prethodni filter. Kod ovakvog filtera moguće je operaciju odbacivanja odbiraka izmestiti pre filtriranja samog signala a da se pri tome ne unesu izobličenja u signal (slika 2b).



Slika 1 – Polifazno filtriranje kod decimacije

Implementacija operacije decimacije koristeći polifazno filtriranje, na osnovu slike 2b može se izvršiti na sledeći način:

```
	output[0] = fir_circular(input[i], lp_filter_poly[0], history[0], N_COEFF_P,
&state[0]);

	for(i = M, j = 1; i< N; i+=M, j++)
	{
		output[j] = fir_circular(input[i], lp_filter_poly[0], history[0],
N_COEFF_P, &state[0]);
		for(k = 1; k < M; k++ )
			output[j] += fir_circular(input[i-k], lp_filter_poly[k],
history[k], N_COEFF_P, &state[k]);
	}
```

Za filtriranjeg prvog odbirka signala prethodnih M-1 odbiraka nije prisutno, tako da se za njih računaju vrednosti 0, odnosno filtriranje se ne vrši. Koeficijente za polifaznu filter banku potrebno je izračnati pre poziva funkcije za filtriranje, na sledeći način:

```
for(i = 0, j=0; i < N_COEFF; i+=M, j++)
	{
		for(k = 0; k < M; k++)
		{
			lp_filter_poly[k][j] = lp_filter[i+k];
		}
	}
```

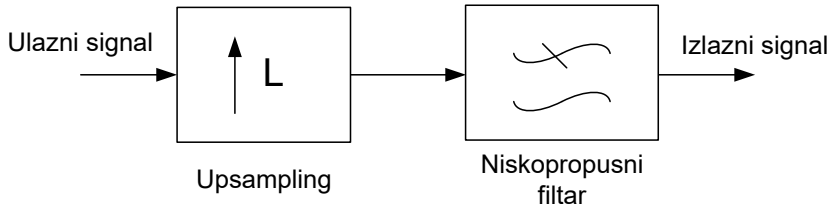
Gde M predstavlja faktor decimacije, odnosno broj faza kod polifazne filter banke. N_COEFF predstavlja broj koeficijenata po fazi, odnosno broj koeficijenata referentnog filtera podeljen sa brojem faza (N_COEFF/M). Ukoliko dati količnik nije ceo broj, potrebno je izvršiti zaokruživanje na više.

2 Interpolacija

Interpolacija je proces suprotan decimaciji odnosno predstavlja proces povećanja frekvencije odabiranja ulaznog signala za neki celobrojni fktor. Kao i faktor decimacije, i faktor interpolacije se definiše kao odnos frekvencije odabiranja ulaznog signala i frekvencije odabiranja signala nastalog interpolacijom. On se najčešće obeležava sa L i predstavlja sledeći količnik:

$$L = \frac{f_{so}}{f_{si}}$$

Kao i decimacija, i interpolacija se u praksi sastoji iz 2 faze: ubacivanja odbiraka u signal (engl. upsampling) i filtriranja niskopropusnim filtrom (Slika 2).



Slika 2 Interpolacija signala za faktor L

Proces ubacivanja odbiraka najčešće se realizuje ubacivanjem L-1 nula odbiraka između svaka 2 uzastopna ulazna odbirka (engl. zero-stuffing) . Filtriranje je neophodno jer unošenje nula odbiraka u ulazni signal dovodi do pojave neželjenih spektralnih slika u izlaznom signalu koje se nalaze na umnošku originalne frekvencije odabiranja. Filtar može biti FIR ili IIR tipa, ali se u praksi češće koriste FIR filtri pošto omogućavaju jednostavniju implementaciju.

Najočigledniji razlog upotrebe interpolacije je da se poveća frekvencija odabiranja na izlazu jednog sistema, da bi se taj izlaz mogao proslediti sistemu koji radi na višoj frekvenciji odabiranja.

S obzirom da je interpolacija bazirana na dodavanju nula odbiraka, interpolacija je moguća samo sa faktorom interpolacije koji je ceo broj. Ne postoji ograničenje što se ulaznih signala tiče. S obzirom da je u pitanju proces povećanja frekvencije odabiranja, bilo koji ulazni sistem ne može ugroziti Nikvistov kriterijum.

Kao i kod decimacije, ukoliko faktor interpolacije nije prost broj, interpolacija se može izvršiti u nekoliko uzastopnih faza. Npr. ukoliko je faktor interpolacije 16, interpolacija se može izvršiti na sledeći način:

- jedna faza: L = 16
- dve faze: L = 4 i L = 4 ili L = 8 i L = 2
- tri faze: L = 2, L = 2 i L = 4
- četiri faze: L = 2, L = 2, L = 2 i L = 2

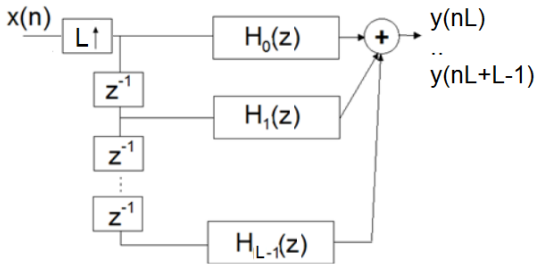
Kao što je pomenuto, kod interpolacije sa faktorom L, filtriranju signala niskopropusnim filterom prethodi umetanje L-1 nula u ulazni signal nakon svakog odbirka, kako bi se povećao broj odbiraka. Ukoliko se signal filtrira koristeći filter dužine N_COEFF, prilikom računanja svakog izlaznog odbirka vrši se N_COEFF množenja i N_COEFF-1 sabiranja. Umajući u vidu da samo svako L-to množeje ima rezultat različit od nule, dok sva ostala množenja predstavljaju množenje sa umetnutom nulom, ovakav pristup je neefikasan (samo 1/L operacija je potrebno).

Da bi se omogućilo efikasnije računanje interpolacije, neophodno je izmeniti niskopropusni filter tako da se računanje vrši samo za ne nulte vrednosti. Kao i kod decimacije ovo je moguće izvršiti razlaganjem filtera, odnosno korišćenjem polifaznog filtriranja.

Nakon razlaganja koeficijenata filtera *h(n)* na isti način kako je to urađeno kod decimacije dobija se matrica sa koeficijentima:

$$\begin{bmatrix} h(0) & .. & h(\frac{N_{COEFF}}{L}-L) \\ .. & .. & .. \\ h(L-1) & .. & h(\frac{N_{COEFF}}{L}-1) \end{bmatrix}$$

Polazeći od sturkture sa slike 2a, ukoliko se umesto odbacivanja odbiraka nakon filteriranja uvede umetanje odbiraka sa vrednosti 0 pre filtriranja dobija se struktura prikazana na slici 3.



Slika 2 - Polifazna implementacija interpolacije

Na osnovu strukture može se za ključiti da za jedan ulazni odbirak x(n) na izlazu se dobijaja L izlazih odbiraka. S obzirom da nakon svakog nenultog odbirka x(n) sledi L-1 nula, zaključuje se da će rezultat y(nL) biti jednak rezultatu filtriranja ulaznog signala sa filterom H_0, y(nL+1) rezultatu filtraranja istog signala filterom H_1, itd. U opštem slučaju ovo možemo zapisati kao:

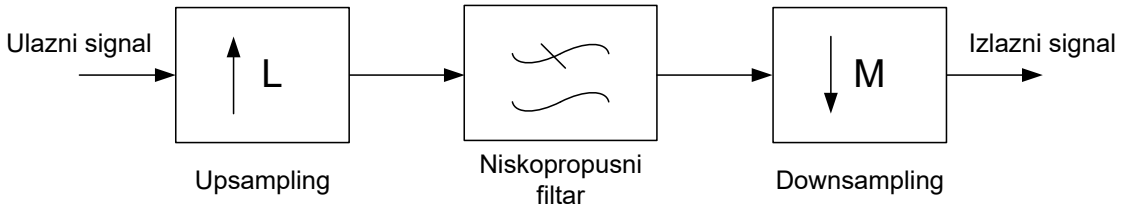
$$y(nL+k) = \sum_{m=0}^{N_{COEFF}/L} h_k(m) \cdot x(n-m)$$

Implemntacija polifazne interpolacije u programskom jeziku C može se realizovati tako što se izvrši izračunavanje novih koeficijenata na isti način kao što je to urađeno kod decimacije. Potom se u okviru funkcije za interpolaciju, prolazi kroz čitav ulazni niz odbiraka. Za svaki odbirak *x[i]* potrebno je pozvati funkciju za filtriranje L puta, sa različitim koeficijentima *h[k]*, gde *k* ima vrednosti *1..L-1*. Rezultati filtriranja se upisuju redom u izlazni niz na pozicije *y[i*L+k].*

3 Konverzija frekvencije odabiranja sa racionalnim faktorom

U slučaju kada je odnos izlazne i ulazne frekvencije odabiranja racionalan broj izražen kao količnik L/M, konverzija frekvencije odabiranja realizuje se kombinacijom interpolacije za faktor L i decimacije za faktor M (Slika 3). Naravno, ovde nije potrebno 2 puta filtrirati signal, već je dovoljno

koristiti jedan filtar čija granična frekvencija odgovara minimumu graničnih frekvencija interpolacionog i decimacionog filtra, u zavisnosti od toga koji od 2 faktora (L ili M) je manji.



Slika 3 Konverzija frekvencije odabiranja za racionalni faktor

Konverzija sa racionalnim faktorom se najčešće koristi prilikom povezivanja sistema koji rade sa različitim frekvencijama odabiranja. Ukoliko je količnik između frekvencija odabiranja ceo broj ili recipročna vrednost celog broja, tada su decimacija ili intepolacija dovoljne. Ukoliko je količnik racionalan broj, tada je potrebna kombinacija decimacije i interpolacije da bi se postigla željena promena frekvencije odabiranja.

Najtipičniji primer korišćenja konverzije sa racionalnim faktorom je konverzija frekvencije odabiranja od 48 kHz (koju koristi profesionalna audio oprema) na frekvenciju odabiranja od 44.1 kHz (koju koristi standardni audio CD). Stoga, transfer muzike sa profesionalne trake na CD zahteva promenu frekvencije odabiranja za faktor:

$$\frac{44100}{48000} = \frac{441}{480} = \frac{147}{160}$$

Da bi se ova konverzija implementirala, potrebno je uraditi interpolaciju za faktor L = 147 i zatim decimaciju za faktor M = 160.

Zadaci

Zadatak 1

- Data je implemntacija funkcije `downsample(Int16 *input, Int16 *output, int M, int N)` koja vrši decimaciju ulaznog signala dužine N za faktor M odbacivanjem odbiraka bez filtriranja.
- Primeniti funkciju na ulazni signal dat u projektu sa faktorom decimacije M = 4
- Prikazati vremenski oblik i amplitudski spektar ulaznog i decimiranog signala. Koristiti veličinu FFT-a 1024.

Zadatak 2

- Data je funkcija `decimate(Int16 *input, Int16 *output, int M, int N)` koja vrši decimaciju ulaznog signala dužine N za faktor M filtriranjem ulaznog signala niskopropusnim filtrom i odbacivanjem odbiraka.

- Primeniti funkciju na ulazni signal dat u projektu sa faktorom decimacije M = 4
- Prikazati vremenski oblik i amplitudski spektar decimiranog signala. Koristiti veličinu FFT-a 1024.
-

Zadatak 3

- Implementirati funkciju `upsample(Int16 *input, Int16 *output, int L, int N)` koja vrši interpolaciju ulaznog signala dužine N za faktor M ubacivanjem nula-odbiraka bez filtriranja.
- Primeniti funkciju na ulazni signal dat u projektu sa faktorom interpolacije L = 4
- Prikazati vremenski oblik i amplitudski spektar ulaznog i interpoliranog signala. Koristiti veličinu FFT-a 1024.

Zadatak 4

- Implementirati funkciju `interpolate(Int16 *input, Int16 *output, int L, int N)` koja vrši interpolaciju ulaznog signala dužine N za faktor M ubacivanjem nula-odbiraka i filtriranjem izlaznog signala niskopropusnim filtrom. Koeficijenti i implementacija filtra dati su u projektu.
- Primeniti funkciju na ulazni signal dat u projektu sa faktorom interpolacije L = 4
- Prikazati vremenski oblik i amplitudski spektar interpoliranog signala. Koristiti veličinu FFT-a 1024.

Zadatak 5

- Implementirati funkciju `resample(Int16 *input, Int16 *output, int L, int M, int N)` koja vrši konverziju frekvencije odabiranja ulaznog signala dužine N za faktor L/M kombinacijom interpolacije za faktor L i decimacije za faktor M.
- Primeniti funkciju na ulazni signal dat u projektu sa faktorom konverzije 3/4 i 4/3.
- Prikazati vremenski oblik i amplitudski spektar izlaznog signala. Koristiti veličinu FFT-a 1024.

Zadatak 6

- Data je funkcija `decimate_poly(Int16 *input, Int16 *output, int M, int N)` koja vrši decimaciju ulaznog signala dužine N za faktor M koristeći polifaznu filter banku.
- Primeniti funkciju na ulazni signal dat u projektu sa faktorom decimacije M = 4
- Prikazati vremenski oblik i amplitudski spektar decimiranog signala. Koristiti veličinu FFT-a 1024.

Zadatak 7

- Implementirati funkciju `interpolate_poly(Int16 *input, Int16 *output, int L, int N)` koja vrši interpolaciju ulaznog signala dužine N za faktor M koristeći polifaznu filter banku. Izračunati koeficijente filtara na osnovu datih koeficijenata u zadatku 4.
- Primeniti funkciju na ulazni signal dat u projektu sa faktorom interpolacije L = 4

- Prikazati vremenski oblik i amplitudski spektar interpoliranog signala. Koristiti veličinu FFT-a 1024.